AWR1xxx Radar Interface Control Document

Revision 0.95

Dec 27, 2017

Contents

List of Figures

List of Tables

Abbreviations

Abbreviation	Description
ADAS	Advanced Driver Assist System
AE	Asynchronous Event
AFE	Analog Front End
APLL	Analog Phase Locked Loop
AWR	Auto Radar
BPM	Binary Phase Modulation
BSS	BIST Sub system
CLPC	Closed Loop Power Control
DCC	Dual Clock Comparator
DFE	Digital Front End filtering chain
DSS	DSP Sub system
ESM	Error Signaling Module
MSS	Master Sub system
OLPC	Open Loop Power Control
SPI	Serial Peripheral Interface

1. Introduction

1.1 Scope

The AWR1243 and AWR1642 products are highly-integrated 77GHz CMOS automotive radar devices. The devices integrate all of the RF and Analog functionality, including VCO, PLL, PA, LNA, Mixer and ADC for multiple TX/RX channels into a single chip. The AWR1243 is an RF transceiver device and it includes 4 receiver channels and 3 transmit channels in a single chip. The AWR1243 also supports multi-chip cascading. The AWR1642 is a radar-on-a-chip device, which includes 4 receive channels and 2 transmit channels and additionally an integrated DSP for radar signal processing.

Both devices include a built-in BIST (Built-in Self-Test) processor, which is responsible to configure the RF/Analog and digital front-end in real-time, as well as to periodically schedule calibration and functional safety monitoring. This enables the mm-Wave front-end to be self-contained and capable of adapting itself to handle temperature and ageing effects, and to enable significant ease-of-use from an external host perspective.

This document contains the Interface Control Specification for communications on the serial interface (SPI) between the Radar device and the external host processor. The same protocol is used in AWR16xx when the messages are sent to Radar Control subsystem (BIST subsystem) from the MCU subsystem (Master subsystem).

1.2 Intended Audience

The intended audience for this document is firmware, host software, and validation engineers needing to understand the format and contents of all communications between the Radar device and the host processor.

2. AWR1xxx Communications Overview

2.1 Communication Link Description

The AWR12xx radar device communicates with the external host processor using the SPI interface. The radar device is configured and controlled from the external host processor by sending commands to AWR12xx device over SPI.

The AWR16xx radar device is configured and controlled using the internal MCU (Master subsystem) and it communicates with an external ECU using the CAN interface.

This document only talks about the communication protocol between radar device and external host processor using SPI in AWR12xx. In AWR16xx, the same protocol is used to communicate between the BIST subsystem and Master subsystem.

2.2 Communication Link configuration

2.2.1 SPI

This interface is synchronous. The interface includes four signals (SPICCLK, SPICS, and Data In and Data Out) and supports clock rates up to 40 MHz. The AWR12xx radar device is always the SPI slave and the external host processor will be the SPI master.

2.2.2 Mailbox

This interface includes a SRAM and an interrupt line from Master subsystem to BIST subsystem. A reverse channel which includes a different SRAM and a different interrupt line from the BIST subsystem to Master subsystem is used for responses which originate from BIST subsystem.

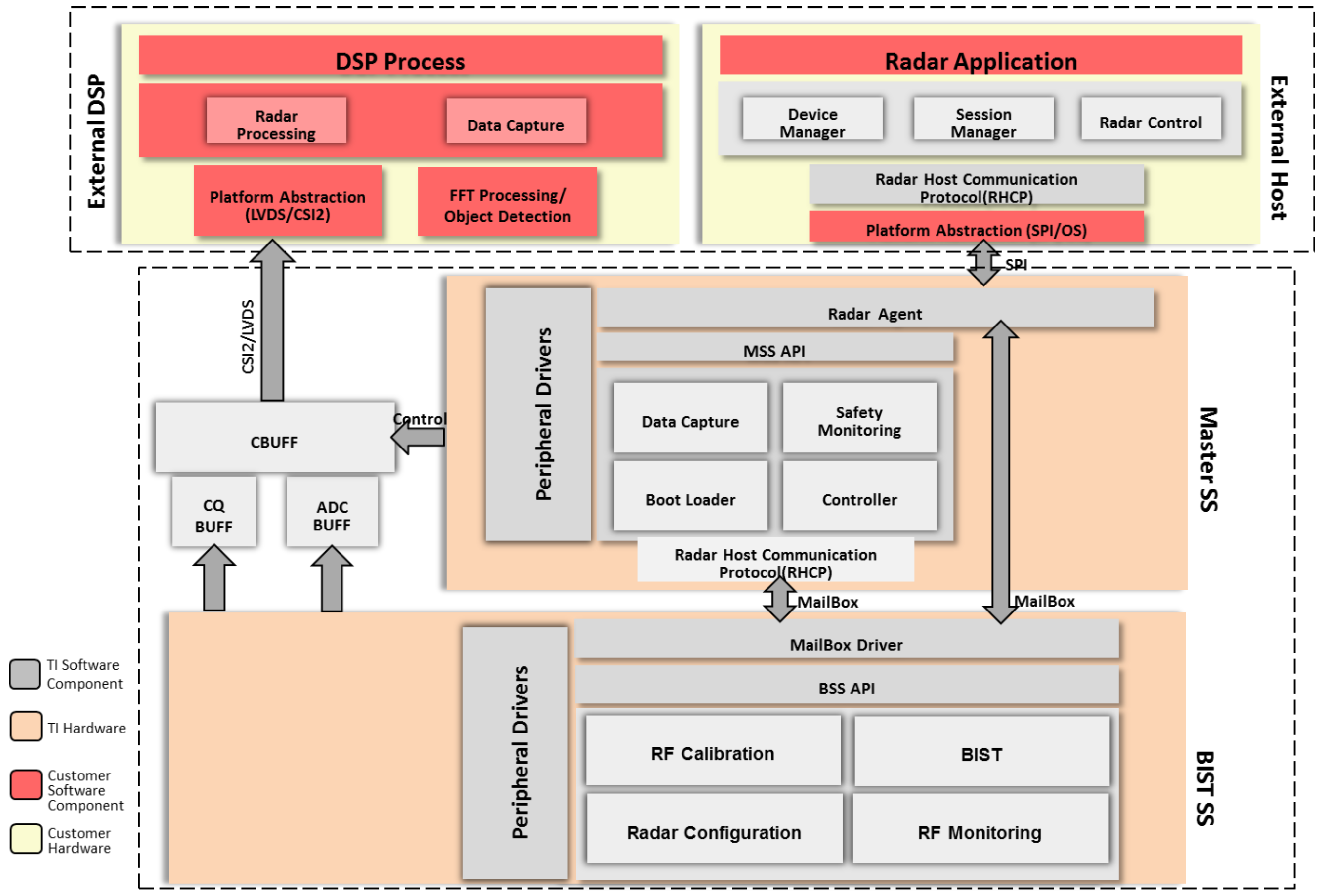


Figure 2.1 – AWR12xx Software Architecture

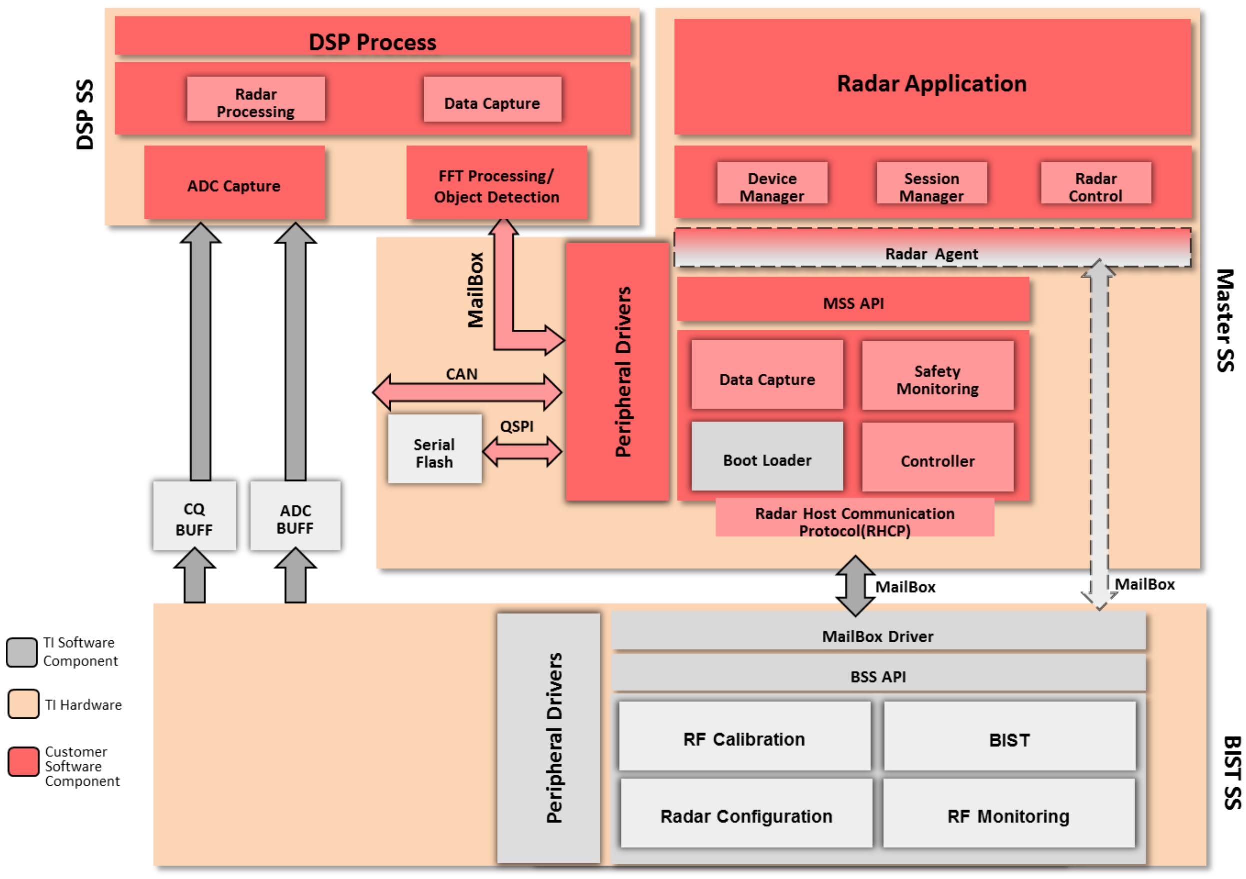


Figure 2.2 – AWR16xx Software Architecture

2.3 Radar Message Structure

Each message is sent in a message envelope, which starts with four special bytes called a sync pattern. Next, the message envelope contains the actual message and a CRC converted to a stream of bytes. FIGURE 2.3 defines the general form of radar messages. All communication messages between external host processor and the radar device will follow this message format. Each message consists of a 12-byte message header, variable length message data followed by a variable length CRC.

NOTE: The CRC and all the fields in the message headers and message data that are larger than one byte are sent in little-endian byte order i.e. the least significant byte is sent first.

A message envelope contains only one message.

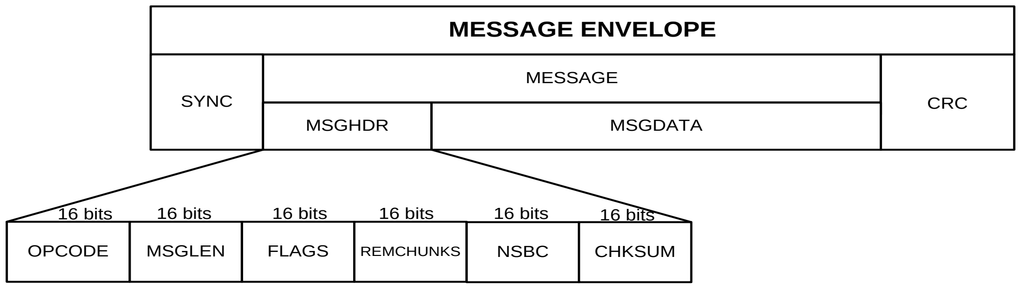


Figure 2.3 – Radar Message Structure

2.3.1 SYNC

SYNC is a unique 4 byte pattern which marks the start of the message. It can take one of the following 3 values, in memory all the bytes are stored in little endian format (least significant byte first).

Table 2.1 – Possible SYNC values and their usage

SYNC word value	**Description**
0x43211234	Messages from master to slave indicating a new command
0x87655678	Messages from external host to device indicating the host is now ready to receive a message from the device This pattern is defined as CNYS in this document.
0xABCDDCBA	Messages from slave to master

2.3.2 MSGHDR

FIGURE 2.4 defines the content of the message header. Each radar message must begin with this 12 byte message header in little endian format.

OPCODE (LS) (16 bits)	LENGTH (16 bits)	FLAGS (16 bits)	REMCHUNKS (16 bits)	NSBC (MS) (16 bits)

Figure 2.4 – Message Header Format

2.3.2.1 OPCODE

The OPCODE is unique for a given message type. FIGURE 2.5 defines the OPCODE format.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSGID										MSGTYPE		DIRECTION			

Figure 2.5 – OPCODE format

Table 2.2 – OPCODE field descriptions

Field	Description	
DIRECTION	Direction of command	
	0000	Invalid
	0001	Communication between Host to BSS
	0010	Communication between BSS to Host
	0011	Communication between Host to DSS
	0100	Communication between DSS to Host
	0101	Communication between Host to Master
	0110	Communication between Master to Host
	0111	Communication between BSS to Master
	1000	Communication between Master to BSS
	1001	Communication between BSS to DSS
	1010	Communication between DSS to BSS
	1011	Communication between Master to DSS
	1100	Communication between DSS to Master
	1101	Reserved
	1110	Reserved
	1111	Reserved
MSGTYPE	Message type	
	00	COMMAND
	01	RESPONSE (ACK or ERROR)
	10	NACK
	11	ASYNC
MSGID	Message ID	
	0x00	AWR_ERROR_MSG
	0x01	RESERVED

	0x02	RESERVED
	0x03	RESERVED
	0x04	AWR_RF_STATIC_CONF_SET_MSG
	0x05	RESERVED
	0x06	AWR_RF_INIT_MSG
	0x07	RESERVED
	0x08	AWR_RF_DYNAMIC_CONF_SET_MSG
	0x09	AWR_RF_DYNAMIC_CONF_GET_MSG
	0x0A	AWR_RF_FRAME_TRIG_MSG
	0x0B	RESERVED
	0x0C	AWR_RF_ADVANCED_FEATURES_CONF_SET_MSG
	0x0D	RESERVED
	0x0E	AWR_RF_MONITORING_CONF_SET_MSG
	0x0F	RESERVED
	0x10	RESERVED
	0x11	AWR_RF_STATUS_GET_MSG
	0x12	RESERVED
	0x13	AWR_RF_MONITORING_REPORT_GET_MSG
	0x14	RESERVED
	0x15	RESERVED
	0x16	AWR_RF_MISC_CONF_SET_MSG
	0x17	AWR_RF_MISC_CONF_GET_MSG
	0x18	RESERVED
	0x19	RESERVED
	0x80	AWR_RF_ASYNC_EVENT_MSG1
	0x81	AWR_RF_ASYNC_EVENT_MSG2
	0x200	AWR_DEV_RFPOWERUP_MSG
	0x201	RESERVED
	0x202	AWR_DEV_CONF_SET_MSG
	0x203	AWR_DEV_CONF_GET_MSG
	0x204	AWR_DEV_FILE_DOWNLOAD_MSG
	0x205	RESERVED
	0x206	AWR_DEV_FRAME_CONFIG_APPLY_MSG
	0x207	AWR_DEV_STATUS_GET_MSG
	0x208	AWR_DEV_MONITORING_CONF_SET_MSG

	0x209	RESERVED
	0x20A	RESERVED
	0x20B	AWR_DEV_MONITORING_REPORT_GET_MSG
	0x20C	AWR_DEV_INTERNAL_CONF_SET_MSG
	0x20D	RESERVED
	0x280	AWR_DEV_ASYNC_EVENT_MSG

2.3.2.2 LENGTH

The length field contains the length of the message in bytes including the message header, message data and CRC. Note that length field does not include the length of the sync field. The minimum length of the message is 12 bytes and maximum is 252 bytes. The message length minus CRC length must also be a multiple of 4 bytes.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				LEN											

Figure 2.6 – MSGLEN format

Table 2.3 – MSGLEN field descriptions

Field	**Description**
LEN	Message length in bytes (It includes message header, message data and CRC)

2.3.2.3 FLAGS

The FLAGS is used to control the communication between the radar device and external host

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEQNUM				CRCLEN		CRCREQ		PROTOCOL VERSION				ACKREQ		RETRY	

Figure 2.7 – FLAGS format

Table 2.4 – FLAGS field descriptions

Field	**Description**
RETRY	RETRY Value 00 New message

Field	Description
	11 Retransmitted message 01 and 10 Reserved
ACKREQ	Acknowledgement Request type 00 Acknowledgement is requested for the current message 11 Acknowledgement is not requested for the current message 01 and 10 Reserved
PROTOCOL VERSION	Version number of the protocol that is used to communicate with the device (4 bits)
CRCREQ	CRC request type 00 CRC is appended to the message 11 CRC is not appended to the message 01 and 10 Reserved
CRCLEN	Length of CRC appended to the message 00 16-bit CRC 01 32-bit CRC 10 64-bit CRC 11 Reserved
SEQNUM	4 bit sequence number of the message. Sequence number is reset to 0 after a device boot and each new message has the incremented sequence number. Whenever the same message is retransmitted, the sequence number is not incremented.

NOTE: It is recommended to always append CRC to the message to prevent any message integrity issues

2.3.2.4 REMCHUNKS

If the message length is larger than 256 bytes, then it is split into multiple chunks of sizes less than 256 bytes. When this field is non-zero, this field indicates the number of remaining chunks that are to be expected.

2.3.2.5 NSBC

The message may contain several configuration sub blocks with structure as defined in Figure 2.3. The NSBC field indicates the total number sub blocks inside the message data.

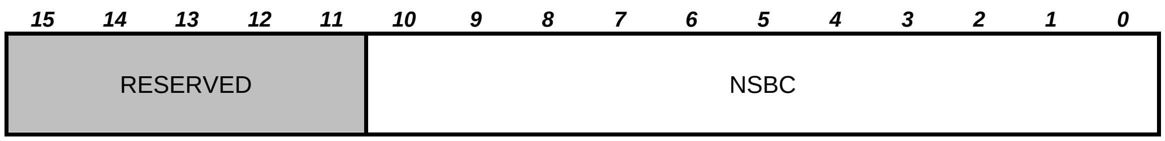


Figure 2.8 – NSBC format

Table 2.5 – NSBC field descriptions

Field	**Description**
NSBC	Number of sub blocks in the message

2.3.2.6 CHKSUM

The message header is protected by a 16-bit checksum to enable the receiver to check the integrity of the message header. The checksum is computed on MSGHDR only (MSGID, MSGLEN, FLAGS, REMCHUNKS and NSBC fields). Note that SYNC field is not included in checksum calculation.

Checksum is 16-bit one's complement of the one's complement sum of all 16-bit words in the message header (Ref. https://tools.ietf.org/html/rfc1071).

For e.g., suppose the message header contents looks like this

OPCODE	0x0281
MSGLEN	0x0800
FLAGS	0x040C
REMCHUNKS	0x0000
NSBC	0x0001
CHKSUM	0xF171

The receiver will compute the checksum as follows 0x0281 + 0x0800 = 0x0A81
Then, 0x0A81 + 0x040C = 0x0E8D
Then, 0x0E8D + 0x0000 = 0x0E8D
Then, 0x0E8D + 0x0001 = 0x0E8E
Ones complement of 0x0E8E is 0xF171 which matches with the received checksum.

2.3.3 MSGDATA

The message data contains the actual message specific data for the message. The message data contains sub blocks with structure as defined in Figure 2.9. More than one sub block can be appended in the MSGDATA to reduce the overall communication latency. The total number of sub blocks in MSGDATA is indicated in the NSBC field in the MSGHDR.

All data fields are aligned so that their offset in message is a multiple of the field size in bytes. For e.g. a 32 bit field in the message will be aligned to a 4 byte boundary and a 16 bit field will be aligned to a 2 byte boundary. This makes it possible to create a structure definition for the message for easy data access in most environments.

Any reserved (currently unused) fields in the messages should be always set as 0 when sent and ignored when received. This way those fields may be taken to use in later interface versions without modifying all old software.

All data structure in sub-blocks assumed to be in little endian format. For big endian Host system byte swap is required to match with defined protocol.

MSGDATA		
SBLKID (16 bits)	SBLKLEN (16 bits)	SBLKDATA (Variable length)

Figure 2.9 – Message Sub block structure

2.3.3.1 SBLKID

Unique ID of the sub block

2.3.3.2 SBLKLEN

Length of the sub block in bytes

2.3.3.3 SBLKDATA

Data corresponding to the sub block

2.3.4 CRC

This is a CRC which is appended to the message data to protect the integrity of the message. The CRC is computed on all the bytes in the MSGHDR and MSGDATA. Note that SYNC is not included in CRC calculation.

3 different types of CRCs can be used – 16 bit, 32 bit or 64 bit. The choice of the CRC type is indicated in the FLAGS field in the MSGHDR.

The polynomials used for each type of CRC calculation are

Table 2.6 – CRC types and their polynomials

CRC type	Polynomial	Remarks
16 bit	$x^{16} + x^{12} + x^5 + 1$	16-bit CRC-CCITT
32 bit	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5$	CRC-32 (used in Ethernet)

	$+ x^4 + x^2 + x + 1$	
64 bit	$x^{64} + x^4 + x^3 + x + 1$	CRC-64-ISO (HDLC)

3. Message Processing

3.1 Communication protocol

When requested by the message transmitter, all correctly formatted radar messages are acknowledged by the receiver. This request for an acknowledgement is specified in FLAGS field of the MSGHDR (message header) field (see Section 2.3.2.3, MSGHDR). A correctly formatted message is one that is formatted properly with a SYNC, MSGHDR, MSGDATA and CRC and that passes the CRC test when received. If an incorrectly formatted message is received, the radar device responds with a NACK message (MSGTYPE field in the MSGHDR set to NACK response). If a correctly formatted message is received, and after processing the message no errors are encountered, the radar device responds with an ACK response. In case of errors on a correctly formatted message, the radar device responds with an ERROR response.

The ACK response is a radar message which contains SYNC, MSGHDR, MSGDATA and CRC. In case the MSGTYPE was COMMAND_GET the MSGDATA for ACK response will contain the parameter values read by the radar device.

The NACK response is a radar message with only SYNC, MSGHDR and CRC. It does not contain MSGDATA.

For most commands the radar device prepares the acknowledgments and response packets immediately on reception. In certain cases, higher priority events in the system delay the execution of external communication function. The response time to command is a function of:

- Speed of the selected communication channel
- Although typical radar command/response occurs within a few hundreds of microseconds, it is recommended that host software wait up to 1 millisecond for response or acknowledgment before timing out on nonresponse.

The radar communication protocol is defined as follows

1. The host sends a message to the radar device requesting an acknowledgement. Host sets a timeout period of 1 ms for a response from the radar device.
2. The radar device checks the CHKSUM field for Message header validity and checks the MSGDATA field for correctness and also computes the CRC of the message and compares it with the received CRC.
    - If the computed CHKSUM does not match the received CHKSUM, the radar device does not send any response. The transmitter will timeout and eventually resend the command again with RETRY flag set
    - If the CRC matches and all parameters are valid/correct, the radar device sends an ACK to the host
    - If the CRC matches, but any parameter in the message is invalid/incorrect, then the radar device sends an ERROR response to the host
    - If the CRC does not match, the radar device sends a NACK response to the host

3. On reception of the ACK, the host can send the next command to the radar device.
4. If the host receives a NACK from the radar device within the timeout period, it sends the message again without the RETRY flag set.
5. If the host does not receive any response from the radar device within the timeout period then it sends the same command with the RETRY flag set.

3.2 Communication Sequence

3.2.1 Command/Response Sequence (Host)

1. Host prepares the message as defined by protocol in Section 2.3
2. Host writes the message to the communication channel and starts Retry Timer (~1 ms)
3. Host then waits for HOST IRQ high Interrupt
    a. If IRQ is received, go to Step 4
    b. If Retry Time expires, Enable Retry Flag and go to Step 2
4. Host writes CNYS (SYNC word = 0x5678 0x8765) and Dymmy bytes (0xFFFF 0xFFFF 0xFFFF 0xFFFF 0xFFFF 0xFFFF) on communication channel
5. Host waits for low on Host IRQ line
    a. If Host IRQ line is low, go to Step 6
    b. If Retry Time expires, Flag Error
6. Host reads the header from communication channel
7. Host checks the validity of header (verify checksum)
    a. If header is valid, parse the header and go to Step 8
    b. If header is invalid, ignore the header and go to Step 3
8. Host reads the payload from communication channel
9. Host checks the validity of the message (verify CRC)
    a. If message is valid, process the message
    b. If message is invalid, go to Step 2

3.2.2 Flow Diagram (Host) – Command/Response

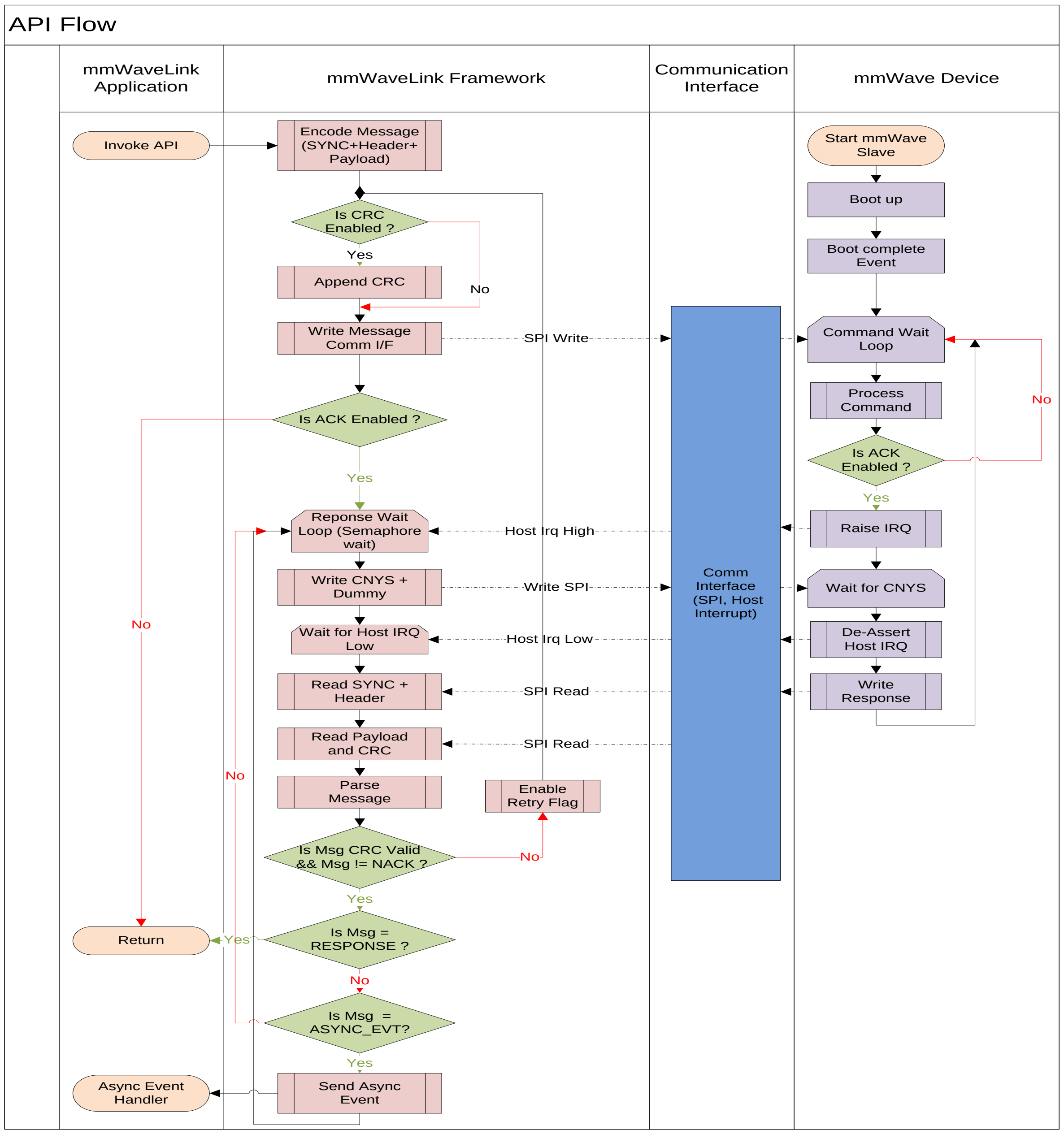


Figure 3.1 – Flow Diagram (API)

3.2.3 Flow Diagram (Host) – Bootup/ Asynchronous Event

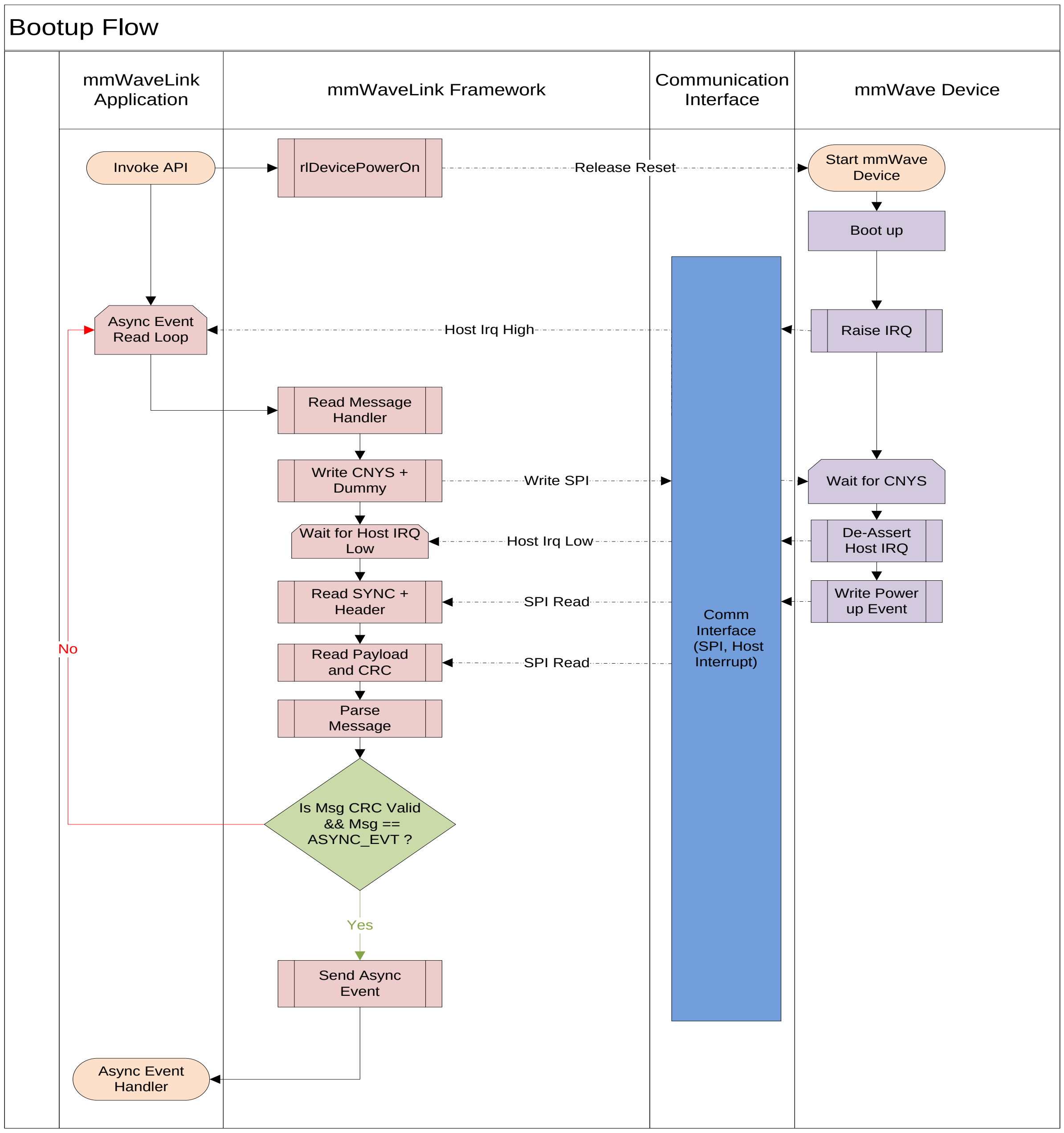


Figure 3.2 – Flow Diagram (Asynchronous Events)

3.2.4 SPI Message Sequence – Command/Response

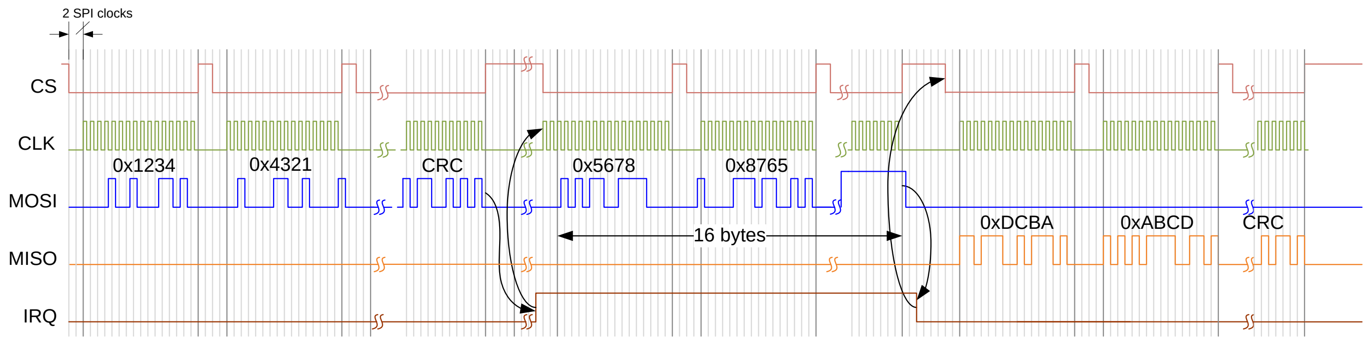


Figure 3.2 – SPI Message Sequence

NOTE:

1. Host should ensure that there is a delay of at least 2 SPI clocks between CS going low and start of SPI clock
2. Host should ensure that CS is toggled for every 16 bits of transfer via SPI
3. There should be a delay of at least 2 SPI Clocks between consecutive CS
4. SPI needs to be operated at Mode 0 (Phase 0, Polarity 0)
5. SPI word length should be 16 bit (Half word)

4. Radar Interface Messages Descriptions

This section describes all the radar interface messages that are used in communication with the radar transceiver.

4.1 Summary of all messages and their associated sub-blocks

Table 4.1 – Summary of all Radar messages and their associated sub blocks

Radar Messages	Associated sub-blocks
AWR_ACK_MSG	NA
AWR_NACK_MSG	NA
AWR_ERROR_MSG	AWR_RESP_ERROR_SB
AWR_RF_STATIC_CONF_SET_MSG	AWR_CHAN_CONF_SET_SB AWR_ADCOUT_CONF_SET_SB AWR_LOWPOWERMODE_CONF_SET_SB AWR_DYNAMICPOWERSAVE_CONF_SET_SB AWR_HIGHSPEEDINTFCLK_CONF_SET_SB AWR_RF_DEVICE_CFG_SB AWR_RF_MISC_CTL_SB AWR_CAL_MON_FREQUENCY_LIMITS_SB AWR_RF_INIT_CALIBRATION_CONF_SB AWR_CAL_MON_FREQUENCY_TX_POWER_LIMITS_SB AWR_CAL_DATA_RESTORE_SB
AWR_RF_STATIC_CONF_GET_MSG	AWR_CAL_DATA_SAVE_SB
AWR_RF_INIT_MSG	AWR_RF_INIT_SB
AWR_RF_DYNAMIC_CONF_SET_MSG	AWR_PROFILE_CONF_SET_SB AWR_CHIRP_CONF_SET_SB AWR_FRAME_CONF_SET_SB AWR_CONT_STREAMING_MODE_CONF_SET_SB AWR_CONT_STREAMING_MODE_EN_SB AWR_ADVANCED_FRAME_CONF_SB AWR_PERCHIRPPHASESHIFT_CONF_SB AWR_PROG_FILT_COEFF_RAM_SET_SB AWR_PROG_FILT_CONF_SET_SB AWR_CALIB_MON_TIME_UNIT_CONF_SB AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB AWR_INTER_RX_GAIN_PHASE_CONTROL_SB

	AWR_RX_GAIN_TEMPLUT_SET_SB AWR_TX_GAIN_TEMPLUT_SET_SB AWR_LOOPBACK_BURST_CONF_SET_SB AWR_DYN_CHIRP_CONF_SET_SB AWR_DYN_PERCHIRP_PHASESHIFTER_CONF_SET_SB AWR_DYN_CHIRP_ENABLE_SB AWR_INTERCHIRP_BLOCKCONTROLS_SB
AWR_RF_FRAME_TRIG_MSG	AWR_FRAMESTARTSTOP_CONF_SB
AWR_RF_ADVANCED_ FEATURES_CONF_SET_MSG	AWR_BPM_COMMON_CONF_SET_SB AWR_BPM_CHIRP_CONF_SET_SB
AWR_RF_MONITORING_CONF_ SET_MSG	AWR_MONITOR_RF_DIG_LATENTFAULT_CONF_SB AWR_MONITOR_RF_DIG_PERIODIC_CONF_SB AWR_MONITOR_ANALOG_ENABLES_CONF_SB AWR_MONITOR_TEMPERATURE_SONF_SB AWR_MONITOR_RX_GAIN_PHASE_CONF_SB AWR_MONITOR_RX_NOISE_FIGURE_CONF_SB AWR_MONITOR_RX_IFSTAGE_CONF_SB AWR_MONITOR_TX0_POWER_CONF_SB AWR_MONITOR_TX1_POWER_CONF_SB AWR_MONITOR_TX2_POWER_CONF_SB AWR_MONITOR_TX0_BALLBREAK_CONF_SB AWR_MONITOR_TX1_BALLBREAK_CONF_SB AWR_MONITOR_TX2_BALLBREAK_CONF_SB AWR_MONITOR_TX_GAIN_PHASE_MISMATCH_CONF_SB AWR_MONITOR_TX0_BPM_CONF_SB AWR_MONITOR_TX1_BPM_CONF_SB AWR_MONITOR_TX2_BPM_CONF_SB AWR_MONITOR_SYNTHESIZER_FREQUENCY_CONF_SB AWR_MONITOR_EXTERNAL_ANALOG_SIGNALS_CONF_SB AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_CONF_ SB AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_CONF_ SB AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_CONF_ SB AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_CONF_S B AWR_MONITOR_PMCLKLO_INTERNAL_ANALOG_SIGNALS_ CONF_SB AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_ CONF_SB

	AWR_MONITOR_PLL_CONTROL_VOLTAGE_SIGNALS_CONF_SB AWR_MONITOR_DUAL_CLOCK_COMP_CONF_SB AWR_MONITOR_RX_SATURATION_DETECTOR_CONF_SB AWR_MONITOR_SIG_IMG_MONITOR_CONF_SB AWR_MONITOR_RX_MIXER_IN_POWER_CONF_SB AWR_ANALOG_FAULT_INJECTION_CONF_SB
AWR_RF_MONITORING_REPORT_GET_MSG	AWR_RF_DFE_STATISTICS_REPORT_GET_SB
AWR_RF_STATUS_GET_MSG	AWR_RF_VERSION_GET_SB AWR_RF_CPUFAULT_STATUS_GET_SB AWR_RF_ESMFAULT_STATUS_GET_SB AWR_RF_BOOTUPBIST_STATUS_GET_SB
AWR_RF_MISC_CONF_SET_MSG	AWR_RF_TEST_SOURCE_CONFIG_SET_SB AWR_RF_TEST_SOURCE_ENABLE_SET_SB AWR_RF_LDO_BYPASS_SB AWR_RF_PALOOPBACK_CFG_SB AWR_RF_PSLOOPBACK_CFG_SB AWR_RF_IFLOOPBACK_CFG_SB AWR_RF_GPADC_CFG_SET_SB
AWR_RF_MISC_CONF_GET_MSG	AWR_RF_TEMPERATURE_GET_SB
AWR_RF_ASYNC_EVENT_MSG1	AWR_AE_RF_CPUFAULT_SB AWR_AE_RF_ESMFAULT_SB AWR_AE_RF_INITCALIBSTATUS_SB AWR_AE_RF_FRAME_TRIGGER_RDY_SB AWR_AE_RF_GPADC_RESULT_DATA_SB AWR_FRAME_END_AE_SB AWR_ANALOGFAULT_AE_SB AWR_CAL_MON_TIMING_FAIL_REPORT_AE_SB AWR_RUN_TIME_CALIBRATION_SUMMARY_REPORT_AE_SB AWR_MONITOR_RF_DIG_LATENTFAULT_REPORT_AE_SB AWR_MONITOR_REPORT_HEADER_AE_SB AWR_MONITOR_RF_DIG_PERIODIC_REPORT_AE_SB AWR_MONITOR_TEMPERATURE_REPORT_AE_SB AWR_MONITOR_RX_GAIN_PHASE_REPORT_AE_SB AWR_MONITOR_RX_NOISE_FIGURE_REPORT_AE_SB AWR_MONITOR_RX_IFSTAGE_REPORT_AE_SB AWR_MONITOR_TX0_POWER_REPORT_AE_SB AWR_MONITOR_TX1_POWER_REPORT_AE_SB

	AWR_MONITOR_TX2_POWER_REPORT_AE_SB AWR_MONITOR_TX0_BALLBREAK_REPORT_AE_SB AWR_MONITOR_TX1_BALLBREAK_REPORT_AE_SB
AWR_RF_ASYNC_EVENT_MSG2	AWR_MONITOR_TX2_BALLBREAK_REPORT_AE_SB AWR_MONITOR_TX_GAIN_PHASE_MISMATCH_REPORT_AE_SB AWR_MONITOR_TX0_BPM_REPORT_AE_SB AWR_MONITOR_TX1_BPM_REPORT_AE_SB AWR_MONITOR_TX2_BPM_REPORT_AE_SB AWR_MONITOR_SYNTHESIZER_FREQUENCY_REPORT_AE_SB AWR_MONITOR_EXTERNAL_ANALOG_SIGNALSREPORT_AE_SB AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_PMCLKLO_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_PLL_CONTROL_VOLTAGE_REPORT_AE_SB AWR_MONITOR_DUAL_CLOCK_COMP_REPORT_AE_SB AWR_MONITOR_RX_MIXER_IN_POWER_REPORT_AE_SB
AWR_DEV_RFPOWERUP_MSG	AWR_DEV_RFPOWERUP_SB
AWR_DEV_CONF_SET_MSG	AWR_DEV_MCUCLOCK_CONF_SET_SB AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB AWR_DEV_RX_DATA_PATH_CONF_SET_SB AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB AWR_DEV_RX_DATA_PATH_CLK_SET_SB AWR_DEV_LVDS_CFG_SET_SB AWR_DEV_RX_CONTSTREAMING_MODE_CONF_ SET_SB AWR_DEV_CSI2_CFG_SET_SB AWR_DEV_PMICCLOCK_CONF_SET_SB AWR_MSS_LATENTFAULT_TEST_CONF_SB AWR_MSS_PERIODICTESTS_CONF_SB AWR_DEV_TESTPATTERN_GEN_SET_SB

	AWR_DEV_CONFIGURATION_SET_SB
AWR_DEV_CONF_GET_MSG	AWR_DEV_MCUCLOCK_GET_SB AWR_DEV_RX_DATA_FORMAT_CONF_GET_SB AWR_DEV_RX_DATA_PATH_CONF_GET_SB AWR_DEV_RX_DATA_PATH_LANEEN_GET_SB AWR_DEV_RX_DATA_PATH_CLK_GET_SB AWR_DEV_LVDS_CFG_GET_SB AWR_DEV_RX_CONTSTREAMING_MODE_CONF_GET_SB AWR_DEV_CSI2_CFG_GET_SB AWR_DEV_PMICCLOCK_CONF_GET_SB AWR_MSS_LATENTFAULT_TEST_CONF_GET_SB AWR_MSS_PERIODICCONF_GET_SB AWR_DEV_TESTPATTEN_GEN_GET_SB
AWR_DEV_FILE_DOWNLOAD_MSG	AWR_DEV_FILE_DOWNLOAD_SB
AWR_DEV_FRAME_CONFIG_APPLY_MSG	AWR_DEV_FRAME_CONFIG_APPLY_SB AWR_DEV_ADV_FRAME_CONFIG_APPLY_SB
AWR_DEV_STATUS_GET_MSG	AWR_MSSVERSION_GET_SB AWR_MSSCPUFAULT_STATUS_GET_SB AWR_MSSESMFAULT_STATUS_GET_SB
AWR_DEV_ASYNC_EVENT_MSG	AWR_AE_DEV_MSSPOWERUPDONE_SB AWR_AE_DEV_RFPOWERUPDONE_SB AWR_AE_MSS_CPUFAULT_SB AWR_AE_MSS_ESMFAULT_SB AWR_AE_MSS_BOOTERRORSTATUS_SB AWR_AE_MSS_LATENTFAULT_TESTREPORT_SB AWR_AE_MSS_PERIODICTEST_STATUS_SB AWR_AE_MSS_RFERROR_STATUS_SB AWR_AE_MSS_VMON_ERRORSTATUS_SB AWR_AE_MSS_ADC_DATA_SB

4.2 AWR_ACK_MSG

The AWR_ACK_MSG is sent by the radar transceiver on a successful reception of a command after its CRC check.

Field Name	**Number of bytes**	**Description**

SYNC	4	Value = 0xABCDDCBA
OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 01 15:6 / MSGID / Same as MSGID in the command
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
CRC	Variable	Based on CRCLEN field in FLAGS

4.3 AWR_NACK_MSG

The AWR_NACK_MSG is sent by the radar transceiver if the CRC check of the command fails.

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0xABCDDCBA
OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 10 15:6 / MSGID / Same as MSGID in the command
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
CRC	Variable	Based on CRCLEN field in FLAGS

4.4 AWR_ERROR_MSG

The AWR_RF_ERROR_MSG is sent by the radar transceiver on finding errors in the command send by host.

Field Name	Number of bytes	Description
SYNC	4	Value = 0xABCDDCBA
OPCODE	2	Bits / Variable name / Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 01 15:6 MSGID 0x00
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_RESP_ERROR_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.5 AWR_RF_STATIC_CONF_SET_MSG

Field Name	Number of bytes	Description
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x04
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6

MSGDATA	Variable	Supported sub blocks AWR_CHAN_CONF_SET_SB AWR_ADCOUT_CONF_SET_SB AWR_LOWPOWERMODE_CONF_ SET_SB AWR_DYNAMICPOWERSAVE_CONF_SET_SB AWR_HIGHSPEEDINTFCLK_CONF_SET_SB AWR_RF_DEVICE_CFG_SB AWR_RF_RADAR_MISC_CTL_SB AWR_CAL_MON_FREQUENCY_LIMITS_SB AWR_RF_INIT_CALIBRATION_CONF_SB AWR_CAL_MON_FREQUENCY_TX_POWER_LIMITS_SB AWR_CAL_DATA_RESTORE_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.6 AWR_RF_STATIC_CONF_GET_MSG

Field Name	Number of bytes	Description
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x05
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_CAL_DATA_SAVE_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.7 AWR_RF_INIT_MSG

Field Name	Number of bytes	Description

SYNC	4	Value = 0x43211234
OPCODE	2	Bits Variable name Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x06
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_RF_INIT_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.8 AWR_RF_DYNAMIC_CONF_SET_MSG

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0x43211234
OPCODE	2	Bits Variable name Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x08
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6

MSGDATA	Variable	Supported sub blocks AWR_PROFILE_CONF_SET_SB AWR_CHIRP_CONF_SET_SB AWR_FRAME_CONF_SET_SB AWR_CONT_STREAMING_MODE_CONF_SET_SB AWR_CONT_STREAMING_MODE_EN_SB AWR_ADVANCED_FRAME_CONF_SB AWR_PHASESHIFT_CONF_SB AWR_PROG_FILT_COEFF_RAM_SET_SB AWR_PROG_FILT_CONF_SET_SB AWR_CALIB_MON_TIME_UNIT_CONF_SB AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB AWR_INTER_RX_GAIN_PHASE_CONTROL_SB AWR_RX_GAIN_TEMPLUT_SET_SB AWR_TX_GAIN_TEMPLUT_SET_SB AWR_LOOPBACK_BURST_CONF_SET_SB AWR_DYN_CHIRP_CONF_SET_SB AWR_DYN_PERCHIRP_PHASESHIFTER_CONF_SB AWR_DYN_CHIRP_ENABLE_SB AWR_INTERCHIRP_BLOCKCONTROLS_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.9 AWR_RF_FRAME_TRIG_MSG

Field Name	Number of bytes	Description
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x0A
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6

MSGDATA	Variable	Supported sub blocks AWR_FRAMESTARTSTOP_CONF_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.10 AWR_RF_ADVANCED_FEATURES_CONF_SET_MSG

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0x43211234
OPCODE	2	Bits Variable name Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x0C
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_BPM_COMMON_CONF_SET_SB AWR_BPM_CHIRP_CONF_SET_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.11 AWR_RF_MONITORING_CONF_SET_MSG

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0x43211234
OPCODE	2	Bits Variable name Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x0E
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3

REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_MONITOR_RF_DIG_LATENTFAULT_CONF_SB AWR_MONITOR_RF_DIG_PERIODIC_CONF_SB AWR_MONITOR_ANALOG_ENABLES_CONF_SB AWR_MONITOR_TEMPERATURE_SONF_SB AWR_MONITOR_RX_GAIN_PHASE_CONF_SB AWR_MONITOR_RX_NOISE_FIGURE_CONF_SB AWR_MONITOR_RX_IFSTAGE_CONF_SB AWR_MONITOR_TX0_POWER_CONF_SB AWR_MONITOR_TX1_POWER_CONF_SB AWR_MONITOR_TX2_POWER_CONF_SB AWR_MONITOR_TX0_BALLBREAK_CONF_SB AWR_MONITOR_TX1_BALLBREAK_CONF_SB AWR_MONITOR_TX2_BALLBREAK_CONF_SB AWR_MONITOR_TX_GAIN_PHASE_MISMATCH_CONF_SB AWR_MONITOR_TX0_BPM_CONF_SB AWR_MONITOR_TX1_BPM_CONF_SB AWR_MONITOR_TX2_BPM_CONF_SB AWR_MONITOR_SYNTHESIZER_FREQUENCY_CONF_SB AWR_MONITOR_EXTERNAL_ANALOG_SIGNALS_CONF_SB AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_CONF_SB AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_CONF_SB AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_CONF_SB AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_CONF_SB AWR_MONITOR_PMCLKLO_INTERNAL_ANALOG_SIGNALS_CONF_SB AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_CONF_SB AWR_MONITOR_PLL_CONTROL_VOLTAGE_SIGNALS_CONF_SB AWR_MONITOR_DUAL_CLOCK_COMP_CONF_SB AWR_MONITOR_RX_SATURATION_DETECTOR_CONF_SB AWR_MONITOR_SIG_IMG_MONITOR_CONF_SB AWR_MONITOR_RX_MIXER_IN_POWER_CONF_SB AWR_ANALOG_FAULT_INJECTION_CONF_SB

CRC	Variable	Based on CRCLEN field in FLAGS

4.12 AWR_RF_STATUS_GET_MSG

Field Name	Number of bytes	Description
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x11
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_RF_VERSION_GET_SB AWR_RF_CPUFAULT_STATUS_GET_SB AWR_RF_ESMFAULT_STATUS_GET_SB AWR_RF_BOOTUPBIST_STATUS_GET_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.13 AWR_RF_MONITORING_REPORT_GET_MSG

Field Name	Number of bytes	Description
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x13
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0

NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_RF_DFE_STATISTICS_REPORT_GET_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.14 AWR_RF_MISC_CONF_SET_MSG

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0x43211234
OPCODE	2	Bits Variable name Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x16
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_RF_TEST_SOURCE_CONFIG_SET_SB AWR_RF_TEST_SOURCE_ENABLE_SET_SB AWR_RF_LDO_BYPASS_SB AWR_RF_PALOOPBACK_CFG_SB AWR_RF_PSLOOPBACK_CFG_SB AWR_RF_IFLOOPBACK_CFG_SB AWR_RF_GPADC_CFG_SET_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.15 AWR_RF_MISC_CONF_GET_MSG

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0x43211234

OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 00 15:6 / MSGID / 0x17
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_RF_TEMPERATURE_GET_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.16 AWR_RF_ASYNC_EVENT_MSG1

The AWR_RF_ASYNC_EVENT_MSG1 is sent by the radar transceiver to the host. This message indicates that specific events have occurred within the device.

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0xABCDDCBA
OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 11 15:6 / MSGID / 0x80
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_AE_RF_CPUFAULT_SB AWR_AE_RF_ESMFAULT_SB AWR_AE_RF_INITCALIBSTATUS_SB AWR_AE_RF_FRAME_TRIGGER_RDY_SB AWR_AE_RF_GPADC_RESULT_DATA_SB

		AWR_FRAME_END_AE_SB AWR_ANALOGFAULT_AE_SB AWR_CAL_MON_TIMING_FAIL_REPORT_AE_SB AWR_RUN_TIME_CALIBRATION_SUMMARY_REPORT_AE_SB AWR_MONITOR_RF_DIG_LATENTFAULT_REPORT_AE_SB AWR_MONITOR_REPORT_HEADER_AE_SB AWR_MONITOR_RF_DIG_PERIODIC_REPORT_AE_SB AWR_MONITOR_TEMPERATURE_REPORT_AE_SB AWR_MONITOR_RX_GAIN_PHASE_REPORT_AE_SB AWR_MONITOR_RX_NOISE_FIGURE_REPORT_AE_SB AWR_MONITOR_RX_IFSTAGE_REPORT_AE_SB AWR_MONITOR_TX0_POWER_REPORT_AE_SB AWR_MONITOR_TX1_POWER_REPORT_AE_SB AWR_MONITOR_TX2_POWER_REPORT_AE_SB AWR_MONITOR_TX0_BALLBREAK_REPORT_AE_SB AWR_MONITOR_TX1_BALLBREAK_REPORT_AE_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.17 AWR_RF_ASYNC_EVENT_MSG2

The AWR_RF_ASYNC_EVENT_MSG2 is sent by the radar transceiver to the host. This message indicates that specific events have occurred within the device.

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0xABCDDCBA
OPCODE	2	Bits — Variable name — Value 3:0 — DIRECTION — See Table 2.2 5:4 — MSGTYPE — 11 15:6 — MSGID — 0x81
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_MONITOR_TX2_BALLBREAK_REPORT_AE_SB AWR_MONITOR_TX_GAIN_PHASE_MISMATCH_REPORT_AE_

		SB AWR_MONITOR_TX0_BPM_REPORT_AE_SB AWR_MONITOR_TX1_BPM_REPORT_AE_SB AWR_MONITOR_TX2_BPM_REPORT_AE_SB AWR_MONITOR_SYNTHESIZER_FREQUENCY_REPORT_AE_SB AWR_MONITOR_EXTERNAL_ANALOG_SIGNALSREPORT_AE_SB AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_PMCLKLO_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB AWR_MONITOR_PLL_CONTROL_VOLTAGE_REPORT_AE_SB AWR_MONITOR_DUAL_CLOCK_COMP_REPORT_AE_SB AWR_MONITOR_RX_MIXER_IN_POWER_REPORT_AE_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.18 AWR_DEV_RFPOWERUP_MSG

The AWR_DEV_RFPOWERUP_MSG is sent by the host to the MSS. This message indicates that BSS can now be powered up.

Field Name	Number of bytes	Description
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 DIRECTION See Table 2.2 5:4 MSGTYPE 00 15:6 MSGID 0x200
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3

REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_DEV_RFPOWERUP_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.19 AWR_DEV_CONF_SET_MSG

The AWR_DEV_CONF_SET_MSG is sent by the host to the radar transceiver. This message indicates that specific events have occurred within the device.

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 00 15:6 / MSGID / 0x202
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6

MSGDATA	Variable	Supported sub blocks AWR_DEV_MCUCLOCK_CONF_SET_SB AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB AWR_DEV_RX_DATA_PATH_CONF_SET_SB AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB AWR_DEV_RX_DATA_PATH_CLK_SET_SB AWR_DEV_LVDS_CFG_SET_SB AWR_DEV_RX_CONTSTREAMING_MODE_CONF_SET_SB AWR_DEV_CSI2_CFG_SET_SB AWR_DEV_PMICCLOCK_CONF_SET_SB AWR_MSS_LATENTFAULT_TEST_CONF_SB AWR_MSS_PERIODICTESTS_CONF_SB AWR_DEV_TESTPATTERN_GEN_SET_SB AWR_DEV_CONFIGURATION_SET_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.20 AWR_DEV_CONF_GET_MSG

The AWR_DEV_CONF_SET_MSG is sent by the host to the radar transceiver. This message indicates that specific events have occurred within the device.

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 00 15:6 / MSGID / 0x203
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6

MSGDATA	Variable	Supported sub blocks AWR_DEV_MCUCLOCK_GET_SB AWR_DEV_RX_DATA_FORMAT_CONF_GET_SB AWR_DEV_RX_DATA_PATH_CONF_GET_SB AWR_DEV_RX_DATA_PATH_LANEEN_GET_SB AWR_DEV_RX_DATA_PATH_CLK_GET_SB AWR_DEV_LVDS_CFG_GET_SB AWR_DEV_RX_CONTSTREAMING_MODE_CONF_GET_SB AWR_DEV_CSI2_CFG_GET_SB AWR_DEV_PMICCLOCK_CONF_GET_SB AWR_MSS_LATENTFAULT_TEST_CONF_GET_SB AWR_MSS_PERIODICCONF_GET_SB AWR_DEV_TESTPATTEN_GEN_GET_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.21 AWR_DEV_FILE_DOWNLOAD_MSG

The AWR_DEV_FILE_DOWNLOAD_MSG is sent by the host to MSS. This message sends a file to be written into the device.

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 00 15:6 / MSGID / 0x204
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_DEV_FILE_DOWNLOAD_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.22 AWR_DEV_FRAME_CONFIG_APPLY_MSG

The AWR_DEV_FRAME_CONFIG_APPLY_MSG is sent by the host to MSS. This message indicates to MSS to apply all the regular framing mode configurations related to ADC buffer and CBUFF.

Field Name	Number of bytes	Description
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 00 15:6 / MSGID / 0x206
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_DEV_FRAME_CONFIG_APPLY_SB AWR_DEV_ADV_FRAME_CONFIG_APPLY_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.23 AWR_DEV_STATUS_GET_MSG

The AWR_DEV_STATUS_GET_MSG is sent by the host to MSS to get some status information from the device.

Field Name	Number of bytes	Description
SYNC	4	Value = 0x43211234
OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 00 15:6 / MSGID / 0x207
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3

REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_MSSVERSION_GET_SB AWR_MSSCPUFAULT_STATUS_GET_SB AWR_MSSESMFAULT_STATUS_GET_SB
CRC	Variable	Based on CRCLEN field in FLAGS

4.24 AWR_DEV_ASYNC_EVENT_MSG

The AWR_DEV_ASYNC_EVENT_MSG is sent by the radar transceiver to the host. This message indicates that specific events have occurred within the device.

Field Name	**Number of bytes**	**Description**
SYNC	4	Value = 0xABCDDCBA
OPCODE	2	Bits / Variable name / Value 3:0 / DIRECTION / See Table 2.2 5:4 / MSGTYPE / 11 15:6 / MSGID / 0x280
MSGLEN	2	Length of the message in bytes (do not include sync length)
FLAGS	2	See Section 2.3.2.3
REMCHUNKS	2	Value = 0
NSBC	2	Number of sub blocks contained in the message
CHKSUM	2	See Section 2.3.2.6
MSGDATA	Variable	Supported sub blocks AWR_AE_DEV_MSSPOWERUPDONE_SB AWR_AE_DEV_RFPOWERUPDONE_SB AWR_AE_MSS_CPUFAULT_SB AWR_AE_MSS_ESMFAULT_SB AWR_AE_MSS_BOOTERRORSTATUS_SB AWR_AE_MSS_LATENTFAULT_TESTREPORT_SB AWR_AE_MSS_PERIODICTEST_STATUS_SB AWR_AE_MSS_RFERROR_STATUS_SB AWR_AE_MSS_VMON_ERRORSTATUS_SB AWR_AE_MSS_ADC_DATA_SB

CRC	Variable	Based on CRCLEN field in FLAGS

5. Radar Functional APIs

This section describes all the radar interface sub blocks that are used in messages for communicating with the radar transceiver. Some of the sub blocks are status responses from the radar device.

5.1 Sub block related to AWR_ERROR_MSG

5.1.1 Sub block 0x0000 – AWR_RESP_ERROR_SB

This sub block contains the error response for an API command.

Table 5.1 describes the contents of this sub block.

Table 5.1 – AWR_RESP_ERROR_SB contents

Field Name	**Number of bytes**	**Description**	
SBLKID	2	Value = 0x0000	
SBLKLEN	2	Value = 8	
API_RESP	2	0x0001	ERROR_CMD: Incorrect MSGID
		0x0002	ERROR_CMD: No Sub block found in the MSG
		0x0003	ERROR_CMD: Incorrect Sub block ID
		0x0004	ERROR_CMD: Incorrect Sub block Length
		0x0005	ERROR_CMD: Incorrect Sub block data
		0x0006	ERROR_PROC: Error in processing the command
		0x0007	ERROR_FILECRCMISMATCH: File CRC mismatched
		0x0008	ERROR_FILETYPEMISMATCH: File type mismatched w.r.t. magic number
		0x0009 – 0xFFFF	See Section 6 for details on error codes from each API
API_RESP_ERROR_SBC_ID	2	0x0000 – 0xFFFF	Sub-Block ID in which Error Occurred for Sub block related Errors

5.2 Sub blocks related to AWR_RF_STATIC_CONF_SET_MSG

5.2.1 *Sub block 0x0080 – AWR_CHAN_CONF_SET_SB*

This sub block contains static device configurations (applicable for the given power cycle) – how many RX and TX channels are needed for operation. It also defines static configurations related to whether the sensor uses a single AWR1xxx or multiple AWR1xxx chips to realize a larger antenna array (multiple is applicable only in AWR12xx).

Table 5.2 describes the contents of this sub block.

Table 5.2 – AWR_CHAN_CONF_SET_SB contents

Field Name	**Number of bytes**	**Description**	
SBLKID	2	Value = 0x0080	
SBLKLEN	2	Value = 12	
RX_CHAN_EN	2	b0	RX_CHAN0_EN 0 Disable RX Channel 0 1 Enable RX Channel 0
		b1	RX_CHAN1_EN 0 Disable RX Channel 1 1 Enable RX Channel 1
		b2	RX_CHAN2_EN 0 Disable RX Channel 2 1 Enable RX Channel 2
		b3	RX_CHAN3_EN 0 Disable RX Channel 3 1 Enable RX Channel 3
		b15:4	RESERVED 0b000000000000
TX_CHAN_EN	2	b0	TX_CHAN0_EN 0 Disable TX Channel 0 1 Enable TX Channel 0
		b1	TX_CHAN1_EN 0 Disable TX Channel 1 1 Enable TX Channel 1

		b2	TX_CHAN2_EN 0 Disable TX Channel 2 1 Enable TX Channel 2
		b15:3	RESERVED 0b0000000000000
CASCADING_CFG	2	This field specifies the cascading configuration. Value Description 0x0000 SINGLECHIP: Single AWR1xxx sensor application 0x0001 MULTICHIP_MASTER: Multiple AWR12xx sensor application. This AWR12xx is the master chip and generates LO and conveys to other AWR12xx's in the sensor. This is applicable only in AWR12xx. 0x0002 MULTICHIP_SLAVE: Multiple AWR12x sensor application. This AWR12xx is a slave chip and uses LO conveyed to it by the master AWR12xx in the sensor. This is applicable only in AWR12xx. MULTICHIP_MASTER and MULTICHIP_SLAVE are in general referred to as MULTICHIP applications, where larger antenna array sizes are possible in comparison with SINGLECHIP cases.	
RESERVED	2	0x0000	

5.2.2 *Sub block 0x0082 – AWR_ADCOUT_CONF_SET_SB*

This sub block contains static device configurations (applicable for the given power cycle) – regarding the data format of the ADC output (including the digital filtering).

Table 5.3 describes the contents of this sub block.

Table 5.3 – AWR_ADCOUT_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0082
SBLKLEN	2	Value = 12

NUM_ADC_BITS	1	b1:0	00 12 bits 01 14 bits 10 16 bits Other RESERVED These bits also define the CSI2 data type format. For 12 bit RAW12 format is used, for 14 bit RAW14 and for 16 bit RAW8 format is used
		b7:2	RESERVED 0b000000
FULL_SCALE_ REDUCTION_ FACTOR	1	b7:0	Number of bits to reduce ADC full scale by Valid range: 0 to (16 – Number of ADC bits) For e.g. for 12 bit ADC output, this field can take values 0, 1, 2 or 3 For 14 bit ADC output, this field can take values 0, 1 or 2 For 16 bit ADC output, this field can take only value 0 Example: If the user desires 12 bit ADC output, then the digital front end (DFE) chain drops 4 LSBs before placing the data in ADC buffer (DFE output is 16 bits wide). If the user sets FULL_SCALE_REDUCTION_FACTOR as 1, then the DFE will drop only 3 LSBs but still restricting the data in ADC buffer to be within +/- 2^12. This allows wider ADC swings in smaller signal conditions.
ADC_OUT_FMT	2	b1:0	00 Real 01 Complex 1x (image band filtered out) 10 Complex 2x (image band visible) 11 Pseudo Real
		b15:2	RESERVED 0b00000000000000
RESERVED	2	0x0000	
RESERVED	2	0x0000	

5.2.3 *Sub block 0x0083 – AWR_LOWPOWERMODE_CONF_SET_SB*

This sub block contains static device configurations (applicable for this power cycle) – Sigma Delta ADC root sampling clock rate (reducing rate to half to save power in small IF bandwidth applications).

Table 5.4 describes the contents of this sub block.

Table 5.4 – AWR_LOWPOWERMODE_CONF_SET_SB contents

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x0083	
SBLKLEN	2	Value = 8	
RESERVED	2	0x0000	
LP_ADC_MODE	2	0x00	Regular ADC mode
		0x01	Low power ADC mode

5.2.4 *Sub block 0x0084 – AWR_DYNAMICPOWERSAVE_CONF_SET_SB*

This sub block defines static device configuration – whether to enable dynamic power saving during inter-chirp IDLE times by turning off various circuits e.g. TX, RX, LO Distribution blocks.

Table 5.5 describes the contents of this sub block.

Table 5.5 – AWR_DYNAMICPOWERSAVE_CONF_SET_SB contents

Field Name	Number of bytes	Description		
SBLKID	2	Value = 0x0084		
SBLKLEN	2	Value = 8		
BLOCK_CFG	2	b0	Enable power save by switching off TX during inter-chirp IDLE period	
			0	Disable
			1	Enable
		b1	Enable power save by switching off RX during inter-chirp IDLE period	
			0	Disable
			1	Enable
		b2	Enable power save by switching off LO Distribution blocks during inter-chirp IDLE period	
			0	Disable
			1	Enable
		b15:3	RESERVED 0b0000000000000	
RESERVED	2	0x0000		

5.2.5 *Sub block 0x0085 – AWR_HIGHSPEEDINTFCLK_CONF_SET_SB*

This sub block contains static device configurations (applicable for the given power cycle) – regarding high speed interface clock rates which are related to sending the ADC data from AWR device to the host in either LVDS or CSI2 format.

Table 5.6 describes the contents of this sub block.

Table 5.6 – AWR_HIGHSPEEDINTFCLK_CONF_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0085
SBLKLEN	2	Value = 8
HSICLKRATECODE	2	This field indicates the high speed interface input clock rate, needed by the LVDS or CSI2 module. It should be *N* times the final serial data rate, where *N* = 2 in DDR mode and *N* = 1 in SDR mode. Bit 15:5 = Reserved (all 0). Bit 3:0 are to be set based on desired rate as follows: (see table below) For example, for 900 Mbps output rate with DDR, choose Bit3:0=0b1101, and for 450 Mbps output rate with SDR, choose Bit3:0=0b0110.
RESERVED	2	0x0000

	Bit 1:0 = 0b00	Bit 1:0 = 0b01	Bit 1:0 = 0b10	Bit 1:0 = 0b11
Bit 3:2 = 0b00	Reserved	800MHz	400MHz	200MHz
Bit 3:2 = 0b01	Reserved	900MHz	450MHz	225MHz
Bit 3:2 = 0b10	Reserved	1200MHz	600MHz	300MHz
Bit 3:2 = 0b11	Reserved	1800MHz	Reserved	Reserved

5.2.6 *Sub block 0x0086 – AWR_RF_DEVICE_CFG_SB*

This sub block configures the direction of async event from BSS. Typically async events are sent to MSS. With this API, the user can configure the destination of async event.

Table 5.7 describes the contents of this sub block.

Table 5.7 – AWR_RF_DEVICE_CFG_SB contents

Field Name	**Number of bytes**	**Description**	
SBLKID	2	Value = 0x0086	
SBLKLEN	2	Value = 16	
RF_AE_DIRECTION	4	b1:b0	ASYNC_EVENT_DIR 00 BSS to MSS 01 BSS to HOST 10 BSS to DSS 11 RESERVED The ASYNC_EVENT_DIR controls the direction for following ASYNC_EVENTS 1. CPU_FAULT 2. ESM_FAULT All other ASYNC_EVENTs are sent to the subsystem which issues the API
		b3:b2	MONITORING_ASYNC_EVENT_DIR 00 BSS to MSS 01 BSS to HOST 10 BSS to DSS 11 RESERVED
		b31:b4	RESERVED 0x0000000
AE_CONTROL	1	b0	FRAME_START_ASYNC_EVENT_DIS 0 Frame Start async event enable 1 Frame Start async event disable
		b1	FRAME_STOP_ASYNC_EVENT_DIS 0 Frame Stop async event enable 1 Frame Stop async event disable
		b7:2	RESERVED 0b000000
RESERVED	2	0x0000	
BSS_DIG_CTRL	1	b0	WDT_DISABLE 0 Keep watchdog disabled 1 Enable watch dog

		b7:1	RESERVED 0b0000000
ASYNC_EVENT_CRC_CONFIG	1	Value 0 1 2	Description 16 bit CRC for BSS async events 32 bit CRC for BSS async events 64 bit CRC for BSS async events
RESERVED	3	0x000000	

5.2.7 *Sub block 0x0087 – AWR_RF_RADAR_MISC_CTL_SB*

This sub block controls miscellaneous global RF controls.

Table 5.8 describes the contents of this sub block.

Table 5.8 – AWR_RF_RADAR_MISC_CTL_SB contents

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x0087	
SBLKLEN	2	Value = 12	
RF_MISC_CTL	4	b0	PERCHIRP_PHASESHIFTER_EN 0 Per chirp phase shifter is disabled 1 Per chirp phase shifter is enabled This control is applicable only in AWR1243. For AWR1642, this is a RESERVED bit and should be set to 0.
		b31:b1	RESERVED 0b000_0000_0000_0000_0000_0000_0000_0000
RESERVED	4	0x00000000	

5.2.8 *Sub block 0x0088 – AWR_CAL_MON_FREQUENCY_LIMITS_SB*

This sub block sets the limits for RF frequency transmission.

Table 5.9 describes the contents of this sub block.

Table 5.9 – AWR_CAL_MON_FREQUENCY_LIMITS_SB contents

Field Name	Number of bytes	Description

SBLKID	2	Value = 0x0088
SBLKLEN	2	Value = 16
FREQ_LIMIT_LOW	2	The sensor's lower frequency limit for calibrations and monitoring is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 100 MHz Valid range: 760 to 810
FREQ_LIMIT_HIGH	2	The sensor's higher frequency limit for calibrations and monitoring is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 100 MHz Valid range: 760 to 810 NOTE: FREQ_LIMIT_HIGH should be strictly greater than FREQ_LIMIT_LOW Examples: For an LRR device deployed in the US, one might typically configure FREQ_LIMIT_LOW to 760 and FREQ_LIMIT_HIGH to 770.
RESERVED	8	RESERVED 0x0000_0000_0000_0000

5.2.9 *Sub block 0x0089 – AWR_RF_INIT_CALIBRATION_CONF_SB*

This sub block sets the limits for RF frequency transmission.

Table 5.9 describes the contents of this sub block.

Table 5.10 – AWR_RF_INIT_CALIBRATION_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0089
SBLKLEN	2	Value = 16

RF_INIT_CALIB_ ENABLE_MASK	4	Normally, upon receiving RF INIT message, the BSS performs all relevant initial calibrations. This step can be disabled by the host by setting the corresponding calibration bit in this field to 0x0. If disabled, the host needs to send the INJECT CALIB DATA message so that the BSS can operate using the calibration data thus injected. Internal/Debug use: Each of these calibrations can be selectively disabled by issuing this message before RF INIT message.

Bit	Definition
0	RESERVED
1	RESERVED
2	RESERVED
3	RESERVED
4	Enable LODIST calibration
5	Enable RX ADC DC offset calibration
6	Enable HPF cutoff calibration
7	Enable LPF cutoff calibration
8	Enable Peak detector calibration
9	Enable TX power calibration
10	Enable RX gain calibration
11	Enable TX Phase calibration
12	Enable RX IQMM calibration
31:13	RESERVED 0b000_0000_0000_0000_0000

ENABLE_SOFT_ REPORT	1	**Bit** **Definition** b0 ENABLE_RF_INIT_CALIB_ DATA_REPORT If this bit is 1, then the calibration results are reported through a RF_INIT_CALIB_DATA structure. This can be injected back (at a subsequent boot) using an INJECT_ CALIB_DATA message and save on calibration time. This feature is not enabled as of now. b1 ENABLE_RF_INIT_CALIB_DEBUG_DATA_REPORT [RESERVED, for debug only] If this bit is 1, calibration related RF_INIT_CALIB_DEBUG_DATA structure is sent to the host. This report contains details/debug information for each calibration. b7:b2 RESERVED
RESERVED	3	0x000000
RESERVED	4	0x00000000

5.2.10 *Sub block 0x008A – AWR_CAL_MON_FREQUENCY_TX_POWER_LIMITS_SB*

This sub block sets the limits for RF frequency transmission for each TX and also TX power limits.

Table 5.11 – AWR_CAL_MON_FREQUENCY_TX_POWER_LIMITS_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x008A
SBLKLEN	2	Value = 28
FREQ_LIMIT_LOW_ TX0	2	The sensor's lower frequency limit for calibrations and monitoring for TX0 is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 10 MHz Valid range: 7600 to 8100
FREQ_LIMIT_LOW_ TX1	2	The sensor's lower frequency limit for calibrations and monitoring for TX1 is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 10 MHz Valid range: 7600 to 8100

FREQ_LIMIT_LOW_TX2	2	The sensor's lower frequency limit for calibrations and monitoring for TX2 is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 10 MHz Valid range: 7600 to 8100
FREQ_LIMIT_HIGH_TX0	2	The sensor's higher frequency limit for calibrations and monitoring for TX0 is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 10 MHz Valid range: 7600 to 8100 NOTE: FREQ_LIMIT_HIGH_TXn should be strictly greater than FREQ_LIMIT_LOW_TXn
FREQ_LIMIT_HIGH_TX1	2	The sensor's higher frequency limit for calibrations and monitoring for TX1 is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 10 MHz Valid range: 7600 to 8100 NOTE: FREQ_LIMIT_HIGH_TXn should be strictly greater than FREQ_LIMIT_LOW_TXn
FREQ_LIMIT_HIGH_TX2	2	The sensor's higher frequency limit for calibrations and monitoring for TX2 is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 10 MHz Valid range: 7600 to 8100 NOTE: FREQ_LIMIT_HIGH_TXn should be strictly greater than FREQ_LIMIT_LOW_TXn
TX0_POWER_BACKOFF	1	TX0 output power back off 1LSB = 1dB Valid values: 0, 3, 6, 9
TX1_POWER_BACKOFF	1	TX1 output power back off 1LSB = 1dB Valid values: 0, 3, 6, 9
TX2_POWER_BACKOFF	1	TX2 output power back off 1LSB = 1dB Valid values: 0, 3, 6, 9
RESERVED	1	0x00
RESERVED	2	0x0000
RESERVED	2	0x0000
RESERVED	2	0x0000
RESERVED	2	0x0000

5.2.11 Sub block 0x008B – AWR_CAL_DATA_RESTORE_SB

This sub block restores the calibration data which was stored previously using the AWR_CAL_DATA_SAVE_SB command.

Table 5.12 – AWR_CAL_DATA_RESTORE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x008B
SBLKLEN	2	Value = 232
RESERVED	2	0x0000
CHUNK_ID	2	Index of the current chunk
CAL_DATA	224	Calibration data which was stored in non-volatile memory

5.3 Sub blocks related to AWR_RF_STATIC_CONF_GET_MSG

5.3.1 Sub block 0x00A0 – 0x00AA – RESERVED

5.3.2 Sub block 0x00AB – AWR_CAL_DATA_SAVE_SB

This sub block reads the calibration data from the device which can be injected later using the AWR_CAL_DATA_RESTORE_SB command.

Table 5.13 – AWR_CAL_DATA_SAVE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x00AB
SBLKLEN	2	Value = 8
RESERVED	2	0x0000
CHUNK_ID	2	Index of the requested chunk Valid values: 0 to NUM_CHUNKS – 1

Response to the above command will contain the calibration data which is formatted as shown below

Field Name	Number of bytes	Description

SBLKID	2	Value = 0x00AB
SBLKLEN	2	Value = 232
NUM_CHUNKS	2	Total number of calibration data chunks
CHUNK_ID	2	Current chunk number
CAL_DATA	224	Calibration data

5.4 Sub blocks related to AWR_RF_INIT_MSG

5.4.1 *Sub block 0x00C0 – AWR_RF_INIT_SB*

This sub block, needed to be initially issued, triggers one time calibrations such as those related to APLL and synthesizer. The BSS processor is woken up upon receiving this sub block, the RF analog and digital baseband sections are initialized.

Table 5.14 describes the content of this sub block.

Table 5.14 – AWR_RFINIT_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x00C0
SBLKLEN	2	Value = 4

NOTE: This sub block will be acknowledged immediately but an async event AWR_AE_RF_INITCALIBSTATUS_SB from BSS will indicate that the RF initialization is complete. No commands shall be sent to BSS till the async event is received.

5.5 Sub blocks related to AWR_RF_DYNAMIC_CONF_SET_MSG

5.5.1 *Sub block 0x0100 – AWR_PROFILE_CONF_SET_SB*

This sub block contains FMCW radar chirp profiles or properties (FMCW slope, chirp duration, TX power etc.). Since the device supports multiple profiles, each profile is defined in this sub block. Internal RF and analog calibrations may be triggered upon receiving this sub block and ASYNC_EVENT response sent once completed.

NOTE: This API can be issued dynamically to change profile parameters. Few parameters which cannot be changed are

1. PF_NUM_ADC_SAMPLES
2. PF_DIGITAL_OUTPUT_SAMPLING_RATE
3. Programmable filter coefficients in xWR1642

Table 5.15 describes the contents of this sub block.

Table 5.15 – AWR_PROFILE_CONF_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0100
SBLKLEN	2	Value = 48
PF_INDX	2	The profile index for which the rest of the fields are applicable for
PF_VCO_SELECT	1	Bit — Description b0 — FORCE_VCO_SEL 0 – Use internal VCO selection 1 – Forced external VCO selection b1 — VCO_SEL 0 – VCO1 (76 – 78 GHz) 1 – VCO2 (77 – 81 GHz) Note: There is an overlap region of 77-78 GHz in which any of the VCOs can be used, for other regions use only the VCO which can work in that region. For e.g. for 76-77 GHz use only VCO1 and for 78-81GHz use only VCO2, for 77-78 GHz, any VCO can be used. Also note that users should not inter-mix chirps from different VCOs within the same frame. b7:2 — RESERVED 0b000000

PF_CALLUT_ UPDATE	1	Bit	Description
		b0	RETAIN_TXCAL_LUT 0 – Update TX calibration LUT 1 – Do not update TX calibration LUT
		b1	RETAIN_RXCAL_LUT 0 – Update RX calibration LUT and update RX IQMM correction 1 – Do not update TX calibration LUT
		b7:2	RESERVED
		If PF_TX_OUTPUT_POWER_BACKOFF is changed then set RETAIN_TXCAL_LUT to 0, else set it to 1 and if PF_RX_GAIN or if sweep bandwidth is changed, then set RETAIN_RXCAL_LUT to 0 else set them to 1	
PF_FREQ_START_ CONST	4	Start frequency for each profile 1 LSB = $3.6e9/2^{26}$ Hz ≈ 53.644 Hz Valid range: 0x5471C71B to 0x5A000000	
PF_IDLE_TIME_ CONST	4	Idle time for each profile 1 LSB = 10 ns Valid range: 0 to 524287	
PF_ADC_START_ TIME_CONST	4	Time of starting of ADC capture relative to the knee of the ramp 1 LSB = 10 ns Valid range: 0 to 4095	
PF_RAMP_END_ TIME	4	End of ramp time relative to the knee of the ramp 1 LSB = 10 ns Valid range: 0 to 500000 Ensure that the total frequency sweep is either within 76-78 GHz or 77-81 GHz	
PF_TX_OUTPUT_ POWER_BACKOFF	4	b7:0	TX0 output power back off
		b15:8	TX1 output power back off
		b23:16	TX2 output power back off
		b31:24	RESERVED 0x00
		This field defines how much the transmit power should be reduced from the maximum. 1 LSB = 1 dB Valid values: [0, 1, 2, 3, 4, 5, 6, 12, 18, 24, 30]	
PF_TX_PHASE_	4	b1:0	RESERVED (set it to 0b00)

SHIFTER		b7:2	TX0 phase shift value 1 LSB = $360^{\circ} / 2^{6} \approx 5.625^{\circ}$
		b9:8	RESERVED (set it to 0b00)
		b15:10	TX1 phase shift value 1 LSB = $360^{\circ} / 2^{6} \approx 5.625^{\circ}$
		b17:16	RESERVED (set it to 0b00)
		b23:18	TX2 phase shift value 1 LSB = $360^{\circ} / 2^{6} \approx 5.625^{\circ}$
		b31:24	RESERVED 0x00
		This field defines the additional phase shift to be introduced on each transmitter output.	
PF_FREQ_SLOPE_ CONST	2	Frequency slope for each profile is encoded in 2 bytes (16 bit signed number) 1 LSB = $3.6e9 \times 900 / 2^{26} \approx 48.279 \text{ kHz}/\mu\text{s}$ Valid range: -2072 to 2072	
PF_TX_START_ TIME	2	Time of start of transmitter relative to the knee of the ramp 1 LSB = 10 ns Valid range: -4096 to 4095 Positive numbers refer to start of TX after knee of the ramp and negative numbers refer to start of TX before the knee of the ramp	
PF_NUM_ADC_ SAMPLES	2	Number of ADC samples to capture in a chirp for each RX Valid range: 2 to MAX_NUM_SAMPLES, Where MAX_NUM_SAMPLES is such that all the enabled RX channels' data fits into 16 kB memory, with each sample consuming 2 bytes for real ADC output case and 4 bytes for complex 1x and complex 2x ADC output cases. For example: 4 RX, Complex ADC output → MAX_NUM_SAMPLES = 1024 4 RX, Real ADC output → MAX_NUM_SAMPLES = 2048 2 RX, Complex ADC output → MAX_NUM_SAMPLES = 2048 2 RX, Real ADC output → MAX_NUM_SAMPLES = 4096	
PF_DIGITAL_ OUTPUT_ SAMPLING_ RATE	2	ADC Sampling rate for each profile is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 1 ksps Valid range 2000 to 37500	

PF_HPF1_ CORNER_FREQ	1	HPF1 corner frequency for each profile is encoded in 1 byte Value HPF1 corner frequency definition 0x00 175 kHz 0x01 235 kHz 0x02 350 kHz 0x03 700 kHz
PF_HPF2_ CORNER_FREQ	1	HPF2 corner frequency for each profile is encoded in 1 byte Value HPF2 corner frequency definition 0x00 350 kHz 0x01 700 kHz 0x02 1.4 MHz 0x03 2.8 MHz
RESERVED	2	0x0000
PF_RX_GAIN	2	(see bit table below)
RESERVED	2	0x0000

PF_RX_GAIN bit definitions:

Bit	Definition
5:0	RX_GAIN This field defines RX gain for each profile. 1 LSB = 1 dB Valid values: all even values from 24 to 52
7:6	RF_GAIN_TARGET **Value** **RF gain target** 00 30 dB 01 34 dB 10 RESERVED 11 26 dB
15:8	RESERVED 0x00

5.5.2 *Sub block 0x0101 – AWR_CHIRP_CONF_SET_SB*

This sub block contains chirp to chirp variations on top of the chirp profiles defined in the AWR_PROFILE_CONF_SET_SB. E.g. which profile is to be used for each chirp in a frame, and small dithers in FMCW start frequency and idle time for each chirp are possible to be defined here.

Table 5.16 describes the contents of this sub block.

Table 5.16 – AWR_CHIRP_CONF_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0101
SBLKLEN	2	Value = 24
CHIRP_START_INDX	2	Valid range 0 to 511
CHIRP_END_INDX	2	Valid range CHIRP_START_INDX to 511
PROFILE_INDX	2	Valid range 0 to 3
RESERVED	2	0x0000
CHIRP_FREQ_ START_VAR	4	1 LSB = $3.6e9/2^{26} \approx 53.644 \text{ Hz}$ Valid range: 0 to 8388607
CHIRP_FREQ_ SLOPE_VAR	2	1 LSB = $3.6e9 \times 900/2^{26} \approx 48.279 \text{ kHz}$ Valid range: 0 to 63
CHIRP_IDLE_TIME_ VAR	2	Idle time of each chirp is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 10 ns Valid range: 0 to 4095
CHIRP_ADC_ START_TIME_VAR	2	ADC start time of each chirp is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 10 ns Valid range: 0 to 4095
CHIRP_TX_EN	2	TX enable selection Bit Definition b0 TX0 Enable b1 TX1 Enable b2 TX2 Enable b15:3 RESERVED 0b0_0000_0000_0000 Note: Maximum of only 2 TX can be turned in a chirp

5.5.3 *Sub block 0x0102 – AWR_FRAME_CONF_SET_SB*

This sub block defines a frame, i.e. a sequence of chirps to be transmitted subsequently, the no. of frames to be transmitted, frame periodicity and how to trigger them.

Table 5.17 describes the contents of this sub block.

Table 5.17 – AWR_FRAME_CONF_SET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x0102
SBLKLEN	2	Value = 28
RESERVED	2	May use to indicate Frame mode or Continuous chirping mode of operation.
CHIRP_START_INDX	2	Valid range 0 to 511
CHIRP_END_INDX	2	Valid range CHIRP_START_INDX to 511
NUM_LOOPS	2	Number of times to repeat from CHIRP_START_INDX to CHIRP_END_INDX in each frame Valid range 1 to 255
NUM_FRAMES	2	Number of frames to transmit This field is ignored and internally assumed as 1 if this AWR1xxx is configured as MULTICHIP_SLAVE in AWR_CHAN_CONF_SB. 16 bit unsigned number Valid range: 0 to 65535 (0 for infinite frames)
RESERVED	2	0x0000
FRAME_ PERIODICITY	4	1 LSB = 5 ns Typical range 1 ms to 1000 ms This is the frame repetition period.
TRIGGER_SELECT	2	0x0001 SWTRIGGER (Software API based triggering): Frame is triggered upon receiving AWR_FRAMESTARTSTOP_CONF_SB. There could be several tens of micro seconds uncertainty in triggering. This mode is not applicable if this AWR1xx is configured as MULTICHIP_SLAVE in AWR_CHAN_CONF_SB. 0x0002 HWTRIGGER (Hardware SYNC_IN based triggering): Each frame is triggered by rising edge of pulse in SYNC_IN pin, after receiving AWR_FRAMESTARTSTOP_CONF_SB (this is to prevent spurious transmission). W.r.t. the SYNC_IN pulse, the actual transmission has 5ns uncertainty in SINGLECHIP and only a 300 ps uncertainty (due to tight inter-chip synchronization needed) in MULTICHIP sensor applications as defined in AWR_CHAN_CONF_SB.
RESERVED	2	0x0000

FRAME_TRIGGER_ DELAY	4	Optional time delay from the SYNC_IN trigger to the occurrence of frame chirps. Applicable only in SINGLECHIP sensor applications, as defined in AWR_CHAN_CONF_SB. It is recommended only for staggering the transmission of multiple radar sensors around the car for interference avoidance, if needed. Typical range is 0 to few tens of micro seconds. Units: 1 LSB = 5 ns

More parameters may be added in near future.

5.5.4 *Sub block 0x0103 – AWR_CONT_STREAMING_MODE_CONF_SET_SB*

This sub block contains configuration needed to enable continuous streaming mode from the device.

Table 5.18 describes the contents of this sub block.

Table 5.18 – AWR_CONT_STREAMING_MODE_CONF_SET_SB contents

Field Name	**Number of bytes**	**Description**	
SBLKID	2	Value = 0x0103	
SBLKLEN	2	Value = 24	
PF_FREQ_START_ CONST	4	Frequency start for each profile is encoded in 4 bytes (32 bit unsigned number) 1 LSB = $3.6e9/2^{26} \approx 53.644$ Hz Valid range: 0 to 0x7FFFFFFF	
PF_TX_OUTPUT_ POWER_ BACKOFF	4	b7:0	TX0 output power back off
		b15:8	TX1 output power back off
		b23:16	TX2 output power back off
		b31:24	RESERVED 0x00
		This field defines how much the transmit power should be reduced from the maximum. 1 LSB = 1 dB Valid values: [0,1,2,3,4,5,6,12,18,24,30]	
PF_TX_PHASE_ SHIFTER	4	b1:0	RESERVED (set it to 0b00)
		b7:2	TX0 phase shift value 1 LSB = $360^{\circ}/2^{6} \approx 5.625^{\circ}$
		b9:8	RESERVED (set it to 0b00)

		b15:10	TX1 phase shift value 1 LSB = $360^{\circ} / 2^{6} \approx 5.625^{\circ}$
		b17:16	RESERVED (set it to 0b00)
		b23:18	TX2 phase shift value 1 LSB = $360^{\circ} / 2^{6} \approx 5.625^{\circ}$
		b31:24	RESERVED 0x00
		This field defines the additional phase shift to be introduced on each transmitter output.	
PF_DIGITAL_ OUTPUT_ SAMPLING_RATE	2	ADC Sampling rate for each profile is encoded in 2 bytes (16 bit unsigned number) 1 LSB = 1 ksps Valid range 2000 to 37500	
PF_HPF1_ CORNER_FREQ	1	HPF1 corner frequency for each profile is encoded in 1 byte Value HPF1 corner frequency definition 0x00 175 kHz 0x01 235 kHz 0x02 350 kHz 0x03 700 kHz	
PF_HPF2_ CORNER_FREQ	1	HPF2 corner frequency for each profile is encoded in 1 byte Value HPF2 corner frequency definition 0x00 350 kHz 0x01 700 kHz 0x02 1.4 MHz 0x03 2.8 MHz	

PF_RX_GAIN	1	This field defines RX gain for continuous streaming mode. **Bit** | **Definition** 5:0 | RX_GAIN This field defines RX gain for each profile. 1 LSB = 1 dB Valid values: all even values from 24 to 52 7:6 | RF_GAIN_TARGET **Value** **RF gain target** 00 30 dB 01 34 dB 10 RESERVED 11 26 dB
VCO_SELECT	1	Bit Description b0 FORCE_VCO_SEL 0 – Use internal VCO selection 1 – Forced external VCO selection b1 VCO_SEL 0 – VCO1 (76 – 78 GHz) 1 – VCO2 (77 – 81 GHz) b7:2 RESERVED 0b00_0000
RESERVED	2	0x00

5.5.5 *Sub block 0x0104 – AWR_CONT_STREAMING_MODE_EN_SB*

This sub block contains configuration needed to enable continuous streaming mode from the device.

Table 5.19 describes the contents of this sub block.

Table 5.19 – AWR_CONT_STREAMING_MODE_EN_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0104
SBLKLEN	2	Value = 8

CONT_ STREAMING_EN	2	Value / Definition 0x0000 Disable continuous streaming mode 0x0001 Enable continuous streaming mode
RESERVED	2	0x0000

5.5.6 *Sub block 0x0105 – AWR_ADVANCED_FRAME_CONF_SB*

This sub block contains advanced frame configuration options.

Table 5.20 describes the contents of this sub block.

Table 5.20 – AWR_ADVANCED_FRAME_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0105
SBLKLEN	2	Value = 152
NUM_SUBFRAMES	1	Number of sub frames enabled in this frame Valid range: 1 to 4
FORCE_SINGLE_ PROFILE	1	Value / Definition 0x0 The profile index set in Chirp Config API message governs which profile is used when that chirp is transmitted 0x1 The profile index indicated in Chirp Config message is ignored and all the chirps in each sub frame use a single profile as indicated by that sub frame’s profile index set in this message.
LOOPBACK_CFG	1	Bit / Definition b0 LOOPBACK_CFG_EN 0 – Disable 1 – Enable b2:1 SUB_FRAME_ID Sub frame ID for which the loopback configuration applies b7:3 RESERVED
RESERVED	1	0x00

SF1_PROFILE_INDX	2	This is applicable only if FORCE_SINGLE_PROFILE is set to 1. Please refer to that field for description. Valid range: 0 to 3 Not applicable for loop-back sub-frame
SF1_CHIRP_ START_INDX	2	Start index of the first chirp for the first burst in sub frame 1 Valid range: 0 to 511 Not applicable for loop-back sub-frame
SF1_NUM_UNIQUE_ CHIRPS_PER_ BURST	2	Number of unique chirps per burst Valid range: 1 to 512 Not applicable for loop-back sub-frame
SF1_NUM_LOOPS_ PER_BURST	2	No. of times to loop through the unique chirps in each burst, without gaps, using HW. Valid range: 1 to 255
SF1_BURST_ PERIOD	4	BURST_PERIOD >= (NUM_LOOPS_PER_BURST)*(Sum total of all unique chirp times per burst) + *InterBurstBlankTime*, where *InterBurstBlankTime* is primarily for sensor calibration / monitoring, thermal control, and some minimum time needed for triggering next burst. InterBurstBlankTime $\geq$ 100 us With loopback enabled, InterBurstBlankTime $\geq$ 350 us NOTE: Across bursts, if the value (Sum total of all unique chirp times per burst), is not a constant, then the actual available blank time can vary and needs to be accounted for. 1 LSB = 5 ns Valid range: 100us to 1s
SF1_CHIRP_ START_INDX_ OFFSET	2	The chirp start index for each burst is determined as the chirp start index of the previous burst plus SFx_START_INDX_OFFSET * BURST_INDX CHIRP_START_INDX = SFx_CHIRP_START_INDX + (SFx_CHIRP_START_INDX_OFFSET * BURST_INDEX) Valid range: 0 to 511 A value of 0 can be used to repeat the same set of unique chirps across bursts. Non-zero values allow spanning a larger number of unique chirps (across bursts). Not applicable for loop-back sub-frame
SF1_NUM_BURSTS	2	Number of bursts constituting this sub frame Valid range: 1 to 512 Not applicable for loop-back sub-frame

SF1_NUM_OUTER_ LOOPS	2	Number of times to loop over the set of above defined bursts, for this sub frame. Valid range: 1 to 64 Not applicable for loop-back sub-frame
RESERVED	2	0x0000

SF1_PERIOD	4	PERIOD >= Sum total time of all bursts + *InterSubFrameBlankTime*, Where, Sum total time of all bursts = Num Outer Loops * Num Bursts * Burst Period. *InterSubFrameBlankTime* is primarily for sensor calibration / monitoring, thermal control, transferring out any safety monitoring data if requested, hardware reconfiguration for next sub frame, retriggering of next SF. *InterSubFrameBlankTime* ≥ 100 us With loopback enabled, InterSubFrameBlankTime ≥ 350 us Add 150 us to *InterSubFrameBlankTime* for test source configuration if test source is enabled. 1 LSB = 5 ns Valid range 100 us to 1 s
RESERVED	4	0x00000000
RESERVED	4	0x00000000
SF2_PROFILE_INDX	2	This is applicable only if FORCE_SINGLE_PROFILE is set to 1. Please refer to that field for description. Valid range: 0 to 3
SF2_CHIRP_ START_INDX	2	Start index of the first chirp for the first burst in sub frame 2 Valid range: 0 to 511
SF2_NUM_UNIQUE_ CHIRPS_PER_ BURST	2	Number of unique chirps per burst Valid range: 1 to 512
SF2_NUM_LOOPS_ PER_BURST	2	No. of times to loop through the unique chirps in each burst, without gaps, using HW. Valid range: 1 to 255
SF2_BURST_ PERIOD	4	BURST_PERIOD >= (NUM_LOOPS_PER_BURST)*(Sum total of all unique chirp times per burst) + *InterBurstBlankTime*, where *InterBurstBlankTime* is primarily for sensor calibration / monitoring, thermal control, and some minimum time needed for triggering next burst. InterBurstBlankTime ≥ 100 us With loopback enabled, InterBurstBlankTime ≥ 350 us NOTE: Across bursts, if the value (Sum total of all unique chirp times per burst), is not a constant, then the actual available blank time can vary and needs to be accounted for. 1 LSB = 5 ns

SF2_CHIRP_ START_INDX_ OFFSET	2	The chirp start index for each burst is determined as the chirp start index of the previous burst plus SFx_START_INDX_OFFSET * BURST_INDX CHIRP_START_INDX = SFx_CHIRP_START_INDX + (SFx_CHIRP_START_INDX_OFFSET * BURST_INDEX) Valid range: 0 to 511 A value of 0 can be used to repeat the same set of unique chirps across bursts. Non-zero values allow spanning a larger number of unique chirps (across bursts).
SF2_NUM_BURSTS	2	Number of bursts constituting this sub frame Valid range: 1 to 512
SF2_NUM_OUTER_ LOOPS	2	Number of times to loop over the set of above defined bursts, for this sub frame. Valid range: 1 to 64
RESERVED	2	0x0000
SF2_PERIOD	4	PERIOD >= Sum total time of all bursts + *InterSubFrameBlankTime*, Where, Sum total time of all bursts = Num Outer Loops * Num Bursts * Burst Period. *InterSubFrameBlankTime* is primarily for sensor calibration / monitoring, thermal control, transferring out any safety monitoring data if requested, hardware reconfiguration for next sub frame, retriggering of next SF. *InterSubFrameBlankTime* $\geq$ 100 us With loopback enabled, InterSubFrameBlankTime $\geq$ 350 us Add 150 us to *InterSubFrameBlankTime* for test source configuration if test source is enabled. 1 LSB = 5 ns Valid range: 100us to 1s
RESERVED	4	0x00000000
RESERVED	4	0x00000000
SF3_PROFILE_INDX	2	This is applicable only if FORCE_SINGLE_PROFILE is set to 1. Please refer to that field for description. Valid range: 0 to 3
SF3_CHIRP_ START_INDX	2	Start index of the first chirp in this sub frame Valid range: 0 to 511
SF3_NUM_UNIQUE_ CHIRPS_PER_ BURST	2	Number of unique chirps per burst Valid range: 1 to 512

SF3_NUM_LOOPS_ PER_BURST	2	No. of times to loop through the unique chirps in each burst, without gaps, using HW. Valid range: 1 to 255
SF3_BURST_ PERIOD	4	BURST_PERIOD >= (NUM_LOOPS_PER_BURST)*(Sum total of all unique chirp times per burst) + *InterBurstBlankTime*, where *InterBurstBlankTime* is primarily for sensor calibration / monitoring, thermal control, and some minimum time needed for triggering next burst. InterBurstBlankTime $\geq$ 100 us With loopback enabled, InterBurstBlankTime $\geq$ 350 us NOTE: Across bursts, if the value (Sum total of all unique chirp times per burst), is not a constant, then the actual available blank time can vary and needs to be accounted for. 1 LSB = 5 ns
SF3_CHIRP_ START_INDX_ OFFSET	2	The chirp start index for each burst is determined as the chirp start index of the previous burst plus SFx_START_INDX_OFFSET * BURST_INDX CHIRP_START_INDX = SFx_CHIRP_START_INDX + (SFx_CHIRP_START_INDX_OFFSET * BURST_INDEX) Valid range: 0 to 511 A value of 0 can be used to repeat the same set of unique chirps across bursts. Non-zero values allow spanning a larger number of unique chirps (across bursts).
SF3_NUM_BURSTS	2	Number of bursts constituting this sub frame Valid range: 1 to 512
SF3_NUM_OUTER_ LOOPS	2	Number of times to loop over the set of above defined bursts, for this sub frame. Valid range: 1 to 64
RESERVED	2	0x0000
SF3_PERIOD	4	PERIOD >= Sum total time of all bursts + *InterSubFrameBlankTime*, Where, Sum total time of all bursts = Num Outer Loops * Num Bursts * Burst Period. *InterSubFrameBlankTime* is primarily for sensor calibration / monitoring, thermal control, transferring out any safety monitoring data if requested, hardware reconfiguration for next sub frame, retriggering of next SF. *InterSubFrameBlankTime* $\geq$ 100 us With loopback enabled, InterSubFrameBlankTime $\geq$ 350 us Add 150 us to *InterSubFrameBlankTime* for test source configuration if test source is enabled. 1 LSB = 5 ns

RESERVED	4	0x00000000
RESERVED	4	0x00000000
SF4_PROFILE_INDX	2	This is applicable only if FORCE_SINGLE_PROFILE is set to 1. Please refer to that field for description. Valid range: 0 to 3
SF4_CHIRP_ START_INDX	2	Start index of the first chirp in this sub frame Valid range: 0 to 511
SF4_NUM_UNIQUE_ CHIRPS_PER_ BURST	2	Number of unique chirps per burst Valid range: 1 to 512
SF4_NUM_LOOPS_ PER_BURST	2	No. of times to loop through the unique chirps in each burst, without gaps, using HW. Valid range: 1 to 255
SF4_BURST_ PERIOD	4	BURST_PERIOD >= (NUM_LOOPS_PER_BURST)*(Sum total of all unique chirp times per burst) + *InterBurstBlankTime*, where *InterBurstBlankTime* is primarily for sensor calibration / monitoring, thermal control, and some minimum time needed for triggering next burst. InterBurstBlankTime $\geq$ 100 us With loopback enabled, InterBurstBlankTime $\geq$ 350 us NOTE: Across bursts, if the value (Sum total of all unique chirp times per burst), is not a constant, then the actual available blank time can vary and needs to be accounted for. 1 LSB = 5 ns Valid range: 100us to 1s
SF4_CHIRP_ START_INDX_ OFFSET	2	The chirp start index for each burst is determined as the chirp start index of the previous burst plus SFx_START_INDX_OFFSET * BURST_INDX CHIRP_START_INDX = SFx_CHIRP_START_INDX + (SFx_CHIRP_START_INDX_OFFSET * BURST_INDEX) Valid range: 0 to 511 A value of 0 can be used to repeat the same set of unique chirps across bursts. Non-zero values allow spanning a larger number of unique chirps (across bursts).
SF4_NUM_BURSTS	2	Number of bursts constituting this sub frame Valid range: 1 to 512
SF4_NUM_ OUTER_LOOPS	2	Number of times to loop over the set of above defined bursts, for this sub frame. Valid range: 1 to 64
RESERVED	2	0x0000

SF4_PERIOD	4	SF_PERIOD >= Sum total time of all bursts + *InterSubFrameBlankTime*, Where, Sum total time of all bursts = Num Outer Loops * Num Bursts * Burst Period. *InterSubFrameBlankTime* is primarily for sensor calibration / monitoring, thermal control, transferring out any safety monitoring data if requested, hardware reconfiguration for next sub frame, retriggering of next SF. *InterSubFrameBlankTime* ≥ 100 us With loopback enabled, InterSubFrameBlankTime ≥ 350 us Add 150 us to *InterSubFrameBlankTime* for test source configuration if test source is enabled. 1 LSB = 5 ns Valid range: 100us to 1s
RESERVED	4	0x00000000
RESERVED	4	0x00000000
NUM_FRAMES	2	Number of frames to transmit (1 frame = all enabled sub frames). If set to 0, frames are transmitted endlessly till Frame Stop message is received. Valid range: 0 to 65535
TRIGGER_SELECT	2	0x0001 SWTRIGGER (Software API based triggering): Frame is triggered upon receiving AWR_FRAMESTARTSTOP_CONF_SB. There could be several tens of micro seconds uncertainty in triggering. This mode is not applicable if this AWR1xx is configured as MULTICHIP_SLAVE in AWR_CHAN_CONF_SB. 0x0002 HWTRIGGER (Hardware SYNC_IN based triggering): Each frame is triggered by rising edge of pulse in SYNC_IN pin, after receiving AWR_FRAMESTARTSTOP_CONF_SB (this is to prevent spurious transmission). W.r.t. the SYNC_IN pulse, the actual transmission has 5ns uncertainty in SINGLECHIP and only a 300 ps uncertainty (due to tight inter-chip synchronization needed) in MULTICHIP sensor applications as defined in AWR_CHAN_CONF_SB.

FRAME_TRIGGER_ DELAY	4	Optional time delay from the SYNC_IN trigger to the occurrence of frame chirps. Applicable only in SINGLECHIP sensor applications, as defined in AWR_CHAN_CONF_SB. It is recommended only for staggering the transmission of multiple radar sensors around the car for interference avoidance, if needed. Typical range is 0 to few tens of micro seconds. Units: 1 LSB = 5 ns
RESERVED	4	0x00000000
RESERVED	4	0x00000000

5.5.7 *Sub block 0x0106 – AWR_PERCHIRPPHASESHIFT_CONF_SB*

This sub block defines static phase shift configurations per chirp in each of the TXs. The API is applicable only in AWR1243P.

Table 5.21 describes the contents of this sub block.

Table 5.21 – AWR_PERCHIRPPHASESHIFT_CONF_SB contents

Field Name	**Number of bytes**	**Description**	
SBLKID	2	Value = 0x0106	
SBLKLEN	2	Value = 12	
CHIRP_START_ INDX	2	b15:0	Start index of the chirp for configuring the phase shifter Valid range 0 to 511
CHIRP_END_INDX	2	b15:0	End index of the chirp for configuring the phase shifter Valid range 0 to 511
TX0_PHASE_ SHIFTER	1	TX0 phase shift value Bits — TX0 phase shift definition b1:0 — RESERVED (set it to 0b00) b7:2 — TX0 phase shift value 1 LSB = $360^\circ / 2^6 = 5.625^\circ$ Valid range: 0 to 63	
TX1_PHASE_ SHIFTER	1	TX1 phase shift value Bits — TX1 phase shift definition b1:0 — RESERVED (set it to 0b00) b7:2 — TX1 phase shift value 1 LSB = $360^\circ / 2^6 = 5.625^\circ$ Valid range: 0 to 63	

TX2_PHASE_ SHIFTER	1	TX2 phase shift value Bits — TX2 phase shift definition b1:0 — RESERVED (set it to 0b00) b7:2 — TX2 phase shift value 1 LSB = $360^{\circ} / 2^{6} = 5.625^{\circ}$ Valid range: 0 to 63
RESERVED	1	0x00

5.5.8 *Sub block 0x0107 – AWR_PROG_FILT_COEFF_RAM_SET_SB*

This sub block can be used to program the coefficients for the external programmable filter. The API is applicable only in AWR1642.

Note that the programmable filter is applicable in Complex 1X and Real-only output modes for sampling rates under 6.25 Msps (Complex 1X) and under 12.5 Msps (Real). This is to allow for a trade-off between digital filter chain setting time and close-in anti-alias attenuation. A real-coefficient FIR with up to 26 taps (16-bit coefficients) is supported in the Complex 1X output mode, and a real-coefficient FIR with up to 20 taps (16-bit coefficients) in supported in the Real output mode

NOTE: This API should be issued before AWR_PROFILE_CONF_SET_SB.

Table 5.22 describes the contents of this sub block.

Table 5.22 – AWR_PROG_FILT_COEFF_RAM_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0107
SBLKLEN	2	Value = 212
COEFF_ARRAY	208	The array of coefficients for the programmable filter, across all profiles, to be stored in the coefficient RAMS. Each tap is a 16-bit signed <1.15, s> number. The exact set of taps to be used for a given profile can be specified through AWR_PROG_FILT_CONF_SB Note: All the filter taps across profiles are to be provided in one shot. There is a HW constraint that each profile's filter taps should start at an even address.

5.5.9 Sub block 0x0108 – AWR_PROG_FILT_CONF_SET_SB

This sub block can be used to configure the coefficients for the external programmable filter and associate them to a certain profile. The API is applicable only in AWR1642. This API should be issued before AWR_PROFILE_CONF_SET_SB.

Table 5.23 describes the contents of this sub block.

Table 5.23 – AWR_PROG_FILT_CONF_SET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x0108
SBLKLEN	2	Value = 8
PROFILE_INDX	1	This field indicates the profile Index for which this configuration applies.
PROG_FILT_ COEFF_ START_INDEX	1	The index of the first coefficient of the programmable filter taps corresponding to this profile in the coefficient RAM programmed using AWR_PROG_FILT_COEFF_SET_SB
PROG_FILT_ LENGTH	1	The length (number of taps) of the filter corresponding to this profile. Together with the previous field, this determines the set of coefficients picked up from the coefficient RAM to form the filter taps for this profile. NOTE: This has to be an even number. For odd-length filters, a 0 (zero) tap needs to be appended at the end to make the length even. This is a HW constraint.
PROG_FILT_FREQ_ SHIFT_FACTOR	1	Relevant only for the Complex 1x output mode with the programmable filter. Determines the magnitude of the frequency shift do be done before filtering using the real-coefficient programmable filter. 1 LSB = 0.01*Fs shift, where Fs is the output sampling rate, specified as PF_DIGITAL_OUTPUT_SAMPLING_ RATE in AWR_PROFILE_CONF_SET_SB

NOTE: PROG_FILT_COEFF_START_INDEX should be an even number

5.5.10 Sub block 0x0109 – AWR_CALIB_MON_TIME_UNIT_CONF_SB

This API sub block is used to set calibration and monitoring time unit.

Table 5.24 – AWR_CALIB_MON_TIME_UNIT_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x0109
SBLKLEN	2	Value = 12
CALIB_MON_TIME_ UNIT	2	Defines the basic time unit, in terms of which calibration and/or monitoring periodicities are to be defined. If any monitoring functions are desired and enabled, the monitoring infrastructure automatically inherits this time unit as the period over which the various monitors are cyclically executed; so this should be set to the desired FTTI. For calibrations, a separate CALIB_PERIODICITY can be specified, as a multiple of this time unit, in AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB Note: Even though calibrations many not be desired every time unit, every time unit shall be made long enough to include active chirping time, time required for all enabled calibrations and monitoring functions. 1 LSB = Duration of one frame. Recommendation: TBD
NUM_OF_ CASCADED_DEV	1	Applicable only in cascaded mode. In non-cascaded mode set this to 1
DEVICE_ID	1	Applicable only in cascaded mode. In non-cascaded mode set this to 0
RESERVED	4	0x0000_0000

5.5.11 *Sub block 0x010A – AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB*

This API is used to trigger one time calibrations instantaneously or schedule periodic run time time calibrations which will be scheduled by the firmware while framing during any available idle slot of 200us.

Table 5.25 – AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x010A

SBLKLEN	2	Value = 24
ONE_TIME_CALIB_ ENABLE_MASK	4	Upon receiving this trigger message, one time calibration of various RF/analog aspects are triggered if the corresponding bits in this field are set to 1. The response is in the form of an asynchronous event sent to the host. The calibrations, if enabled, are performed after the completion of any ongoing calibration cycle, and the calibration results take effect from the frame that begins after the asynchronous event response is sent from the BSS. APLL and SYNTH calibrations are done always internally irrespective of bits are enabled or not, the time required for these calibrations must be allocated. **Bit** / **Definition** 0 RESERVED 1 RESERVED 2 RESERVED 3 RESERVED 4 LODIST_CALIBRATION_EN 5 RESERVED 6 RESERVED 7 RESERVED 8 PD_CALIBRATION_EN 9 TX_POWER_CALIBRATION_EN 10 RX_GAIN_CALIBRATION_EN 11 RESERVED 12 RESERVED 31:13 RESERVED 0b

Nested table in ONE_TIME_CALIB_ENABLE_MASK:

Bit	Definition
0	RESERVED
1	RESERVED
2	RESERVED
3	RESERVED
4	LODIST_CALIBRATION_EN
5	RESERVED
6	RESERVED
7	RESERVED
8	PD_CALIBRATION_EN
9	TX_POWER_CALIBRATION_EN
10	RX_GAIN_CALIBRATION_EN
11	RESERVED
12	RESERVED
31:13	RESERVED 0b

PERIODIC_CALIB_ ENABLE_MASK	4	Automatic periodic triggering of calibrations of various RF/analog aspects can be set up by the host issuing this message with corresponding bits in this field set to 1. **Bit** / **Definition** 0 / RESERVED 1 / RESERVED 2 / RESERVED 3 / RESERVED 4 / LODIST_CALIBRATION_EN 5 / RESERVED 6 / RESERVED 7 / RESERVED 8 / PD_CALIBRATION_EN 9 / TX_POWER_CALIBRATION_EN 10 / RX_GAIN_CALIBRATION_EN 11 / RESERVED 12 / RESERVED 31:13 / RESERVED APLL and SYNTH calibrations are done always internally irrespective of bits are enabled or not, the time required for these calibrations must be allocated.
CALIBRATION_ PERIODICITY	4	This field is applicable only for those calibrations which are enabled to be done periodically in the PERIODIC_CALIB_ENABLE_MASK field. This field indicates the desired periodicity of calibrations. If this field is set to N, the results of the first calibration (based on ONE_TIME_CALIB_ENABLE_MASK) are applicable for the first N CALIB_MON_TIME_UNITs. The results of the next calibration are applicable for the next N CALIB_MON_TIME_UNITs, and so on. Recommendation: Set CALIBRATION_PERIODICITY such that frequency of calibrations is greater than or equal to 1 second. 1 LSB = 1 CALIB_MON_TIME_UNIT, as specified in AWR_CALIB_MON_TIME_UNIT_CONF_SB. If the user does not wish to receive calibration reports when periodic calibrations are not enabled, then the user should set CALIBRATION_PERIODICITY to 0

ENABLE_CAL_ REPORT	1	<table><tr><th>Bit</th><th>Definition</th></tr><tr><td>b0</td><td>ENABLE_SUMMARY_REPORT 0 Summary reports are disabled 1 Summary reports are enabled</td></tr><tr><td>b7:1</td><td>RESERVED</td></tr></table>
RESERVED	1	0x00
TX_POWER_CAL_ MODE	1	<table><tr><th>Bit</th><th>Definition</th></tr><tr><td>b0</td><td>TX_POWER_CAL_MODE 0 Update TX gain setting from LUT and do a closed loop calibration (OLPC + CLPC) 1 Update TX gain settings from LUT only (OLPC only) OLPC: Open Loop Power Control. In this mode the TX stage codes are set based on a coarse measurement and a LUT generated for every temperature and the stage codes are picked from the LUT CLPC: Closed Loop Power Control. In this mode the TX stage codes are picked from the coarse LUT as generated in OLPC step. Later the TX power is measured and the TX stage codes are corrected to achieve the desired TX power accuracy.</td></tr><tr><td>b7:1</td><td>RESERVED</td></tr></table>
RESERVED	1	0x00
RESERVED	4	0x00000000

5.5.12 *Sub block 0x010B – AWR_INTER_RX_GAIN_PHASE_CONTROL_SB*

This API can be used to induce different gain/phase offsets on the different RXs, for inter-RX mismatch compensation.

Table 5.26 – AWR_INTER_RX_GAIN_PHASE_CONTROL_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x010B

SBLKLEN	2	Value = 28
PROFILE_INDX	1	This field indicates the profile Index for which this configuration applies.
RESERVED	3	0x000000
DIGITAL_GAIN	4	One byte per RX (8-bit signed number) **Bits** **Assignment** b7:0 RX0 digital gain b15:8 RX1 digital gain b23:16 RX2 digital gain b31:24 RX3 digital gain 1 LSB = 0.1 dB Valid Range: -120 to 119
DIGITAL_PHASE_SHIFT	8	Two bytes per RX **Bits** **Assignment** b15:0 RX0 digital phase shift b31:16 RX1 digital phase shift b47:32 RX2 digital phase shift b63:48 RX3 digital phase shift 1 LSB = $360^\circ / 2^{16} \approx 0.0055^\circ$ Valid Range: 0 to 65535 NOTE: This field is NOT applicable when ADC_OUT_FMT is 00 (real output)
RESERVED	8	0x00000000

5.5.13 Sub block 0x010C – AWR_RX_GAIN_TEMPLUT_SET_SB

This API can be used to overwrite the RX gain temperature LUT used in firmware. This API should be issued after profile configuration API.

Table 5.27 – AWR_RX_GAIN_TEMPLUT_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x010C
SBLKLEN	2	Value = 28

PROFILE_INDX	1	This field indicates the profile Index for which this configuration applies.
RESERVED	1	0x00
RX_GAIN_CODE	19	Byte0: RX gain code for temperature < -30 deg C Byte1: RX gain code for temperature [-30, -20) deg C Byte2: RX gain code for temperature [-20, -10) deg C Byte3: RX gain code for temperature [-10, 0) deg C Byte4: RX gain code for temperature [0, 10) deg C Byte5: RX gain code for temperature [10, 20) deg C Byte6: RX gain code for temperature [20, 30) deg C Byte7: RX gain code for temperature [30, 40) deg C Byte8: RX gain code for temperature [40, 50) deg C Byte9: RX gain code for temperature [50, 60) deg C Byte10: RX gain code for temperature [60, 70) deg C Byte11: RX gain code for temperature [70, 80) deg C Byte12: RX gain code for temperature [80, 90) deg C Byte13: RX gain code for temperature [90, 100) deg C Byte14: RX gain code for temperature [100, 110) deg C Byte15: RX gain code for temperature [110, 120) deg C Byte16: RX gain code for temperature [120, 130) deg C Byte17: RX gain code for temperature [130, 140) deg C Byte18: RX gain code for temperature ≥ 140 deg C Each byte is encoded as follows

Bits	Definition
4:0	IF_GAIN_CODE IF gain is IF_GAIN_CODE * 2 – 6 dB Valid values: 0 to 17 1 LSB = 2 dB
7:5	RF_GAIN_CODE

Value	RF Gain
0	Maximum RF gain
1	Maximum RF gain – 2dB
2	Maximum RF gain – 4dB
3	Maximum RF gain – 6dB
4	Maximum RF gain – 8dB

RESERVED	1	0x00
RESERVED	2	0x0000

5.5.14 Sub block 0x010D – AWR_TX_GAIN_TEMPLUT_SET_SB

This API can be used to overwrite the TX gain temperature LUT used in firmware. This API should be issued after profile configuration API.

Table 5.28 – AWR_TX_GAIN_TEMPLUT_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x010D
SBLKLEN	2	Value = 68
PROFILE_INDX	1	This field indicates the profile Index for which this configuration applies
RESERVED	1	0x00
TX0_GAIN_CODE	19	Byte0: TX0 gain code for temperature < -30 deg C Byte1: TX0 gain code for temperature [-30, -20) deg C Byte2: TX0 gain code for temperature [-20, -10) deg C Byte3: TX0 gain code for temperature [-10, 0) deg C Byte4: TX0 gain code for temperature [0, 10) deg C Byte5: TX0 gain code for temperature [10, 20) deg C Byte6: TX0 gain code for temperature [20, 30) deg C Byte7: TX0 gain code for temperature [30, 40) deg C Byte8: TX0 gain code for temperature [40, 50) deg C Byte9: TX0 gain code for temperature [50, 60) deg C Byte10: TX0 gain code for temperature [60, 70) deg C Byte11: TX0 gain code for temperature [70, 80) deg C Byte12: TX0 gain code for temperature [80, 90) deg C Byte13: TX0 gain code for temperature [90, 100) deg C Byte14: TX0 gain code for temperature [100, 110) deg C Byte15: TX0 gain code for temperature [110, 120) deg C Byte16: TX0 gain code for temperature [120, 130) deg C Byte17: TX0 gain code for temperature [130, 140) deg C Byte18: TX0 gain code for temperature ≥ 140 deg C Each byte is encoded as follows **Bits** / **Definition** 5:0 / STG_CODE Higher values for higher gain 7:6 / RESERVED
RESERVED	1	0x00

TX1_GAIN_CODE	19	Byte0: TX1 gain code for temperature $<$-30 deg C Byte1: TX1 gain code for temperature [-30, -20) deg C Byte2: TX1 gain code for temperature [-20, -10) deg C Byte3: TX1 gain code for temperature [-10, 0) deg C Byte4: TX1 gain code for temperature [0, 10) deg C Byte5: TX1 gain code for temperature [10, 20) deg C Byte6: TX1 gain code for temperature [20, 30) deg C Byte7: TX1 gain code for temperature [30, 40) deg C Byte8: TX1 gain code for temperature [40, 50) deg C Byte9: TX1 gain code for temperature [50, 60) deg C Byte10: TX1 gain code for temperature [60, 70) deg C Byte11: TX1 gain code for temperature [70, 80) deg C Byte12: TX1 gain code for temperature [80, 90) deg C Byte13: TX1 gain code for temperature [90, 100) deg C Byte14: TX1 gain code for temperature [100, 110) deg C Byte15: TX1 gain code for temperature [110, 120) deg C Byte16: TX1 gain code for temperature [120, 130) deg C Byte17: TX1 gain code for temperature [130, 140) deg C Byte18: TX1 gain code for temperature $\geq$140 deg C
RESERVED	1	0x00
TX2_GAIN_CODE	19	Byte0: TX2 gain code for temperature $<$-30 deg C Byte1: TX2 gain code for temperature [-30, -20) deg C Byte2: TX2 gain code for temperature [-20, -10) deg C Byte3: TX2 gain code for temperature [-10, 0) deg C Byte4: TX2 gain code for temperature [0, 10) deg C Byte5: TX2 gain code for temperature [10, 20) deg C Byte6: TX2 gain code for temperature [20, 30) deg C Byte7: TX2 gain code for temperature [30, 40) deg C Byte8: TX2 gain code for temperature [40, 50) deg C Byte9: TX2 gain code for temperature [50, 60) deg C Byte10: TX2 gain code for temperature [60, 70) deg C Byte11: TX2 gain code for temperature [70, 80) deg C Byte12: TX2 gain code for temperature [80, 90) deg C Byte13: TX2 gain code for temperature [90, 100) deg C Byte14: TX2 gain code for temperature [100, 110) deg C Byte15: TX2 gain code for temperature [110, 120) deg C Byte16: TX2 gain code for temperature [120, 130) deg C Byte17: TX2 gain code for temperature [130, 140) deg C Byte18: TX2 gain code for temperature $\geq$140 deg C

RESERVED	1	0x00
RESERVED	2	0x0000

5.5.15 Sub block 0x010E – AWR_LOOPBACK_BURST_CONF_SET_SB

This API can be used to introduce loopback chirps within the functional frames. This loopback chirps will be introduced only if advanced frame configuration is used where user can define which sub-frame contains loopback chirps. The following loopback configuration will apply to one burst and user can program up to 16 different loopback configurations in 16 different bursts of a given sub-frame. User has to ensure that the corresponding sub-frame is defined in AWR_ADVANCED_FRAME_CONF_SB and sufficient time is given to allow the loopback bursts to be transmitted.

NOTE 1:	If user desires to enable loopback chirps within functional frames, then this API should be issued after AWR_PROFILE_CONF_SET_SB
NOTE 2:	Only profile based phase shifter is supported in loopback configuration. Per-chirp phase shifter if enabled will not be reflected in loopback chirps.

Table 5.29 – AWR_LOOPBACK_BURST_CONF_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x010E
SBLKLEN	2	Value = 48
LOOPBACK_SEL	1	**Value** / **Definition**: 0 – No loopback; 1 – IF loopback; 2 – PS loopback; 3 – PA loopback; Others – RESERVED
BASE_PROFILE_ INDX	1	Base profile used for loopback chirps Valid values 0 to 3
BURST_INDX	1	Indicates the index of the burst in the loopback sub-frame for which this configuration applies Valid values 0 to 15
RESERVED	1	0x00

FREQ_CONST	4	Start frequency for loopback. The start frequency configured here should be within profile's sweep bandwidth. 1 LSB = $3.6e9/2^{26}$ Hz ≈ 53.644 Hz Valid range: 0x5471C71B to 0x5A000000
SLOPE_CONST	2	Frequency slope for loopback burst (32 bit signed number) 1 LSB = $3.6e9 \times 900/2^{26} \approx 48.279$ kHz/μs Valid range: -2072 to 2072
RESERVED	2	0x0000
TX_BACKOFF	4	**Bits** / **Definition** b7:0 — TX0 back off, 1 LSB = 1 dB b15:8 — TX1 back off, 1 LSB = 1 dB b23:16 — TX2 back off, 1 LSB = 1 dB b31:24 — RESERVED
RX_GAIN	2	**Bit** / **Definition** 5:0 — RX_GAIN This field defines RX gain for each profile. 1 LSB = 1 dB Valid values: all even values from 24 to 52 7:6 — RF_GAIN_TARGET **Value** **RF gain target** 00 30 dB 01 34 dB 10 RESERVED 11 26 dB 15:8 — RESERVED
TX_ENABLE	1	**Bit** / **Definition** b0 — TX0 b1 — TX1 b2 — TX2 b7:3 — RESERVED

RESERVED	1	0x00
BPM_CONFIG	2	**Bit** **Definition** b0 CONST_BPM_VAL_TX0_OFF Value of Binary Phase Shift value for TX0, during idle time b1 CONST_BPM_VAL_TX0_ON Value of Binary Phase Shift value for TX0, during chirp b2 CONST_BPM_VAL_TX1_OFF For TX1 b3 CONST_BPM_VAL_TX1_ON For TX1 b4 CONST_BPM_VAL_TX2_OFF For TX2 b5 CONST_BPM_VAL_TX2_ON For TX2 b15:6 RESERVED
DIGITAL_ CORRECTION_ DISABLE	2	**Bits** **Digital corrections** b0 IQMM correction disable (only for PS and PA loopback, for IF loopback IQMM is disabled by firmware) 0 – Enable, 1- Disable b1 Inter-RX Gain and Phase correction disable 0 – Enable, 1– Disable b15:2 RESERVED

IF_LOOPBACK_ FREQ	1	Value — IF Loopback frequency value 0 — 180 kHz 1 — 240 kHz 2 — 360 kHz 3 — 720 kHz 4 — 1 MHz 5 — 2 MHz 6 — 2.5 MHz 7 — 3 MHz 8 — 4.02 MHz 9 — 5 MHz 10 — 6 MHz 11 — 8.03 MHz 12 — 9 MHz 13 — 10 MHz 255-13 — RESERVED
IF_LOOPBACK_ MAG	1	1 LSB = 10 mV Valid range: 1 to 63

PS1_PGA_GAIN_ INDEX	1	**Value**	**PGA gain value**
		0	PGA is OFF
		1	-22 dB
		2	-16 dB
		3	-15 dB
		4	-14 dB
		5	-13 dB
		6	-12 dB
		7	-11 dB
		8	-10 dB
		9	-9 dB
		10	-8 dB
		11	-7 dB
		12	-6 dB
		13	-5 dB
		14	-4 dB
		15	-3 dB
		16	-2 dB
		17	-1 dB
		18	0 dB
		19	1 dB
		20	2 dB
		21	3 dB
		22	4 dB
		23	5 dB
		24	6 dB
		25	7 dB
		26	8 dB
		27	9 dB
		255-28	RESERVED

PS2_PGA_GAIN_INDEX	1	**Value** **PGA gain value** 0 PGA is OFF 1 -22 dB 2 -16 dB 3 -15 dB 4 -14 dB 5 -13 dB 6 -12 dB 7 -11 dB 8 -10 dB 9 -9 dB 10 -8 dB 11 -7 dB 12 -6 dB 13 -5 dB 14 -4 dB 15 -3 dB 16 -2 dB 17 -1 dB 18 0 dB 19 1 dB 20 2 dB 21 3 dB 22 4 dB 23 5 dB 24 6 dB 25 7 dB 26 8 dB 27 9 dB 255-28 RESERVED

PS_LOOPBACK_ FREQ	4	Phase shifter loop back frequency in kHz 1 LSB = 1 kHz **Bits** **Definition** b15:0 TX0 Loopback Frequency b31:16 TX1 Loopback Frequency
RESERVED	4	RESERVED
PA_LOOPBACK_ FREQ	2	This value is a 100MHz divider which sets the loopback frequency For e.g. for a 1 MHz frequency, set this to 100 For a 2 MHz frequency, set this to 50 Note: To ensure no leakage of signal power, user has to ensure that 100MHz/LOOPBACK_FREQ is an integer multiple of bin width For e.g. if user choses 25Msps sampling rate and 2048 samples/chirp, then LOOPBACK_FREQ of 64 (=1.5625MHz) will ensure no leakage
RESERVED	2	0x0000
RESERVED	2	0x0000
RESERVED	2	0x0000

5.5.16 Sub block 0x010F – AWR_DYN_CHIRP_CONF_SET_SB

This API can be used to dynamically change the chirp configuration while frames are on-going. The configuration will be stored in software and the new configuration will be applied after receiving the AWR_DYN_CHIRP_ENABLE_SB API.

Table 5.30 – AWR_DYN_CHIRP_CONF_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x010F
SBLKLEN	2	Value = 200
RESERVED	1	0x00
CHIRP_SEGMENT_ SELECT	1	Valid range 0 to 31. Indicates the segment of the chirp RAM that the 16 chirp definitions in this sub block map to
RESERVED	2	0x0000

		Bits	Definition
CHIRP1_R1	4	3:0	PROFILE_INDX Valid range 0 to 3
		7:4	RESERVED
		13:8	FREQ_SLOPE_VAR 1 LSB = $3.6e9 \times 900 / 2^{26} \approx 48.279 \text{ kHz}$ Valid range: 0 to 63
		15:14	RESERVED
		18:16	TX_ENABLE Bit Definition b0 TX0 Enable b1 TX1 Enable b2 TX2 Enable
		23:19	RESERVED
		29:24	BPM_CONSTANT_BITS Bit Definition b0 CONST_BPM_VAL_TX0_OFF Value of Binary Phase Shift value for TX0, when during idle time b1 CONST_BPM_VAL_TX0_ON Value of Binary Phase Shift value for TX0, during chirp b2 CONST_BPM_VAL_TX1_OFF For TX1 b3 CONST_BPM_VAL_TX1_ON For TX1 b4 CONST_BPM_VAL_TX2_OFF For TX2 b5 CONST_BPM_VAL_TX2_ON For TX2
		31:30	RESERVED

CHIRP1_R2	4	**Bits**	**Definition**
		b22:0	FREQ_START_VAR 1 LSB = $3.6e9/2^{26} \approx 53.644\ \text{Hz}$ Valid range: 0 to 8388607
		b31:23	RESERVED
CHIRP1_R3	4	**Bits**	**Definition**
		b11:0	IDLE_TIME_VAR 1 LSB = 10 ns Valid range: 0 to 4095
		b15:12	RESERVED
		b27:16	ADC_START_TIME_VAR 1 LSB = 10 ns Valid range: 0 to 4095
		b31:28	RESERVED
CHIRP2_R1	4	See description for CHIRP1_R1	
CHIRP2_R2	4	See description for CHIRP1_R2	
CHIRP2_R3	4	See description for CHIRP1_R3	
…	…	…	
CHIRP16_R1	4	See description for CHIRP1_R1	
CHIRP16_R2	4	See description for CHIRP1_R2	
CHIRP16_R3	4	See description for CHIRP1_R3	

5.5.17 *Sub block 0x0110 – AWR_DYN_PERCHIRP_PHASESHIFTER_CONF_SET_SB*

This API can be used to dynamically change the per-chirp phase shifter configuration (applicable only in AWR1243P) while frames are on-going. The configuration will be stored in software and the new configuration will be applied after receiving the AWR_DYN_CHIRP_ENABLE_SB API.

Table 5.31 – AWR_DYN_PERCHIRP_PHASESHIFTER_CONF_SET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x0110
SBLKLEN	2	Value = 56
RESERVED	1	0x00

CHIRP_SEGMENT_ SELECT	1	Valid range 0 to 31. Indicates the segment of the chirp RAM that the 16 chirp definitions in this sub block map to
CHIRP1_TX0_ PHASE_ SHIFTER	1	TX0 phase shift value Bits TX0 phase shift definition b1:0 RESERVED (set it to 0b00) b7:2 TX0 phase shift value 1 LSB = $360^{\circ} / 2^{6} = 5.625^{\circ}$ Valid range: 0 to 63
CHIRP1_TX1_ PHASE_ SHIFTER	1	TX1 phase shift value Bits TX1 phase shift definition b1:0 RESERVED (set it to 0b00) b7:2 TX1 phase shift value 1 LSB = $360^{\circ} / 2^{6} = 5.625^{\circ}$ Valid range: 0 to 63
CHIRP1_TX2_ PHASE_ SHIFTER	1	TX2 phase shift value Bits TX2 phase shift definition b1:0 RESERVED (set it to 0b00) b7:2 TX2 phase shift value 1 LSB = $360^{\circ} / 2^{6} = 5.625^{\circ}$ Valid range: 0 to 63
CHIRP2_TX0_ PHASE_ SHIFTER	1	See description for CHIRP1_TX0_ PHASE_ SHIFTER
CHIRP2_TX1_ PHASE_ SHIFTER	1	See description for CHIRP2_TX0_ PHASE_ SHIFTER
CHIRP2_TX2_ PHASE_ SHIFTER	1	See description for CHIRP3_TX0_ PHASE_ SHIFTER
…	…	…
CHIRP16_TX0_ PHASE_ SHIFTER	1	See description for CHIRP1_TX0_ PHASE_ SHIFTER
CHIRP16_TX1_ PHASE_ SHIFTER	1	See description for CHIRP2_TX0_ PHASE_ SHIFTER
CHIRP16_TX2_ PHASE_ SHIFTER	1	See description for CHIRP3_TX0_ PHASE_ SHIFTER
RESERVED	2	0x0000

5.5.18 *Sub block 0x0111 – AWR_DYN_CHIRP_ENABLE_SB*

This API can be used to trigger the copy of chirp configuration from software to hardware. The copy will be performed at the end of the ongoing frame.

Table 5.32 – AWR_DYN_CHIRP_ENABLE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0111
SBLKLEN	2	Value = 8
RESERVED	4	0x00000000

NOTE: HW reconfiguration time (as shown in the figure below) is around 200 us. User has to ensure that AWR_DYN_CHIRP_ENABLE_SB API is issued at least 200 us before the start of the next frame

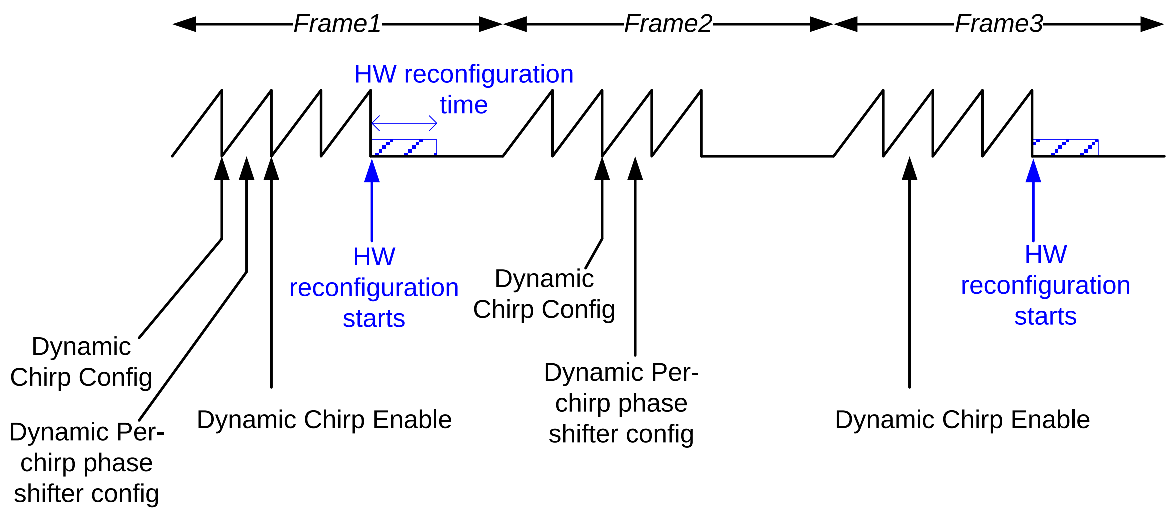

Frame1
Frame2
Frame3
HW reconfiguration time
HW reconfiguration starts
Dynamic Chirp Config
Dynamic Chirp Config
Dynamic Per-chirp phase shifter config
Dynamic Chirp Enable
Dynamic Per-chirp phase shifter config
Dynamic Chirp Enable
HW reconfiguration starts

Figure 5.1 – Dynamic chirp configuration use case timing diagram

5.5.19 *Sub block 0x0112 – AWR_INTERCHIRP_BLOCKCONTROLS_SB*

This API can be used to program the inter-chip turn on and turn off times or various RF blocks.

Table 5.33 – AWR_INTERCHIRP_BLOCKCONTROLS_SB contents

Field Name	**Number of bytes**	**Description**

SBLKID	2	Value = 0x0112
SBLKLEN	2	Value = 44
RX02_RF_TURN_ OFF_TIME	2	Time to wait after ramp end before turning off RX0 and RX2 RF stages. 1 LSB = 10 ns Valid range: -1024 to 1023
RX13_RF_TURN_ OFF_TIME	2	Time to wait after ramp end before turning off RX1 and RX3 RF stages. 1 LSB = 10 ns Valid range: -1024 to 1023
RX02_BB_TURN_ OFF_TIME	2	Time to wait after ramp end before turning off RX0 and RX2 baseband stages. 1 LSB = 10 ns Valid range: -1024 to 1023
RX13_BB_TURN_ OFF_TIME	2	Time to wait after ramp end before turning off RX1 and RX3 baseband stages. 1 LSB = 10 ns Valid range: -1024 to 1023
RX02_RF_PRE_ ENABLE_TIME	2	Time before TX Start Time when RX0 and RX2 RF stages are to be put in fast-charge state. 1 LSB = 10 ns Valid range: -1024 to 1023
RX13_RF_PRE_ ENABLE_TIME	2	Time before TX Start Time when RX1 and RX3 RF stages are to be put in fast-charge state. 1 LSB = 10 ns Valid range: -1024 to 1023
RX02_BB_PRE_ ENABLE_TIME	2	Time before TX Start Time when RX1 and RX3 baseband stages are to be put in fast-charge state. 1 LSB = 10 ns Valid range: -1024 to 1023
RX13_BB_PRE_ ENABLE_TIME	2	Time before TX Start Time when RX2 and RX4 baseband stages are to be put in fast-charge state. 1 LSB = 10 ns Valid range: -1024 to 1023
RX02_RF_TURN_ ON_TIME	2	Time before TX Start Time when RX1 and RX3 RF stages are to be enabled. 1 LSB = 10 ns Valid range: -1024 to 1023

RX13_RF_TURN_ ON_TIME	2	Time before TX Start Time when RX2 and RX4 RF stages are to be enabled. 1 LSB = 10 ns Valid range: -1024 to 1023
RX02_BB_TURN_ ON_TIME	2	Time before TX Start Time when RX1 and RX3 baseband stages are to be enabled. 1 LSB = 10 ns Valid range: -1024 to 1023
RX13_BB_TURN_ ON_TIME	2	Time before TX Start Time when RX2 and RX4 baseband stages are to be enabled. 1 LSB = 10 ns Valid range: -1024 to 1023
RX_LO_CHAIN_ TURN_OFF_TIME	2	Time to wait after ramp end before turning off RX LO chain. 1 LSB = 10 ns Valid range: -1024 to 1023
TX_LO_CHAIN_ TURN_OFF_TIME	2	Time to wait after ramp end before turning off TX LO chain. 1 LSB = 10 ns Valid range: -1024 to 1023
RX_LO_CHAIN_ TURN_ON_TIME	2	Time before TX Start Time when the RX LO chain is to be enabled. 1 LSB = 10 ns Valid range: -1024 to 1023
TX_LO_CHAIN_ TURN_ON_TIME	2	Time before TX Start Time when the TX LO chain is to be enabled. 1 LSB = 10 ns Valid range: -1024 to 1023
RESERVED	4	0x00000000
RESERVED	4	0x00000000

NOTE: The minimum inter-chirp time should be greater than maximum of the following

1. abs(RX02_RF_TURN_OFF_TIME) + max(abs(RX02_RF_PRE_ENABLE_TIME), abs(RX02_RF_TURN_ON_TIME))
2. abs(RX13_RF_TURN_OFF_TIME) + max(abs(RX13_RF_PRE_ENABLE_TIME), abs(RX13_RF_TURN_ON_TIME))
3. abs(RX02_BB_TURN_OFF_TIME) + max(abs(RX02_BB_PRE_ENABLE_TIME), abs(RX02_BB_TURN_ON_TIME))
4. abs(RX13_BB_TURN_OFF_TIME) + max(abs(RX13_BB_PRE_ENABLE_TIME),

abs(RX13_BB_TURN_ON_TIME)

5. abs(RX_LO_TURN_OFF_TIME) + abs(RX_LO_TURN_ON_TIME)
6. abs(TX_LO_TURN_OFF_TIME) + abs(TX_LO_TURN_ON_TIME)

5.6 Sub blocks related to AWR_RF_DYNAMIC_CONF_GET_SB

5.6.1 Sub block 0x0120 – AWR_PROFILE_CONF_GET_SB

This sub block reads the parameters of a given profile. The profile details are available as part of the acknowledgement. The structure is same as AWR_PROFILE_CONF_SET_SB

Table 5.34 describes the contents of this sub block.

Table 5.34 – AWR_PROFILE_CONF_GET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0120
SBLKLEN	2	Value = 8
PROFILE_INDX	2	Valid range 0 to 3 Index of the profile which is to be read
RESERVED	2	0x0000

5.6.2 Sub block 0x0121 – AWR_CHIRP_CONF_GET_SB

This sub block reads the parameters of a given chirp. The profile details are available as part of the acknowledgement. The structure is same as AWR_CHIRP_CONF_SET_SB

Table 5.35 describes the contents of this sub block.

Table 5.35 – AWR_CHIRP_CONF_GET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0121
SBLKLEN	2	Value = 8
CHIRP_START_ INDX	2	Valid range 0 to 511 Starting index of the chirp which is to be read
CHIRP_END_INDX	2	Valid range 0 to 511 Ending index of the chirp which is to be read

5.6.3 Sub block 0x0122 – AWR_FRAME_CONF_GET_SB

This sub block reads the parameters of the configured frame. The profile details are available as part of the acknowledgement. The structure is same as AWR_FRAME_CONF_SET_SB

Table 5.36 describes the contents of this sub block.

Table 5.36 – AWR_CHIRP_CONF_GET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0122
SBLKLEN	2	Value = 4

5.6.4 Sub block 0x0123 – RESERVED

5.6.5 Sub block 0x0124 – RESERVED

5.6.6 Sub block 0x0125 – AWR_ADV_FRAME_CONF_GET_SB

This sub block reads the parameters of the configured frame. The profile details are available as part of the acknowledgement. The structure is same as AWR_ADVANCED_FRAME_CONF_SET_SB

Table 5.37 describes the contents of this sub block.

Table 5.37 – AWR_CHIRP_CONF_GET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x0125
SBLKLEN	2	Value = 4

5.6.7 Sub block 0x0126 – RESERVED

5.6.8 Sub block 0x0127 – RESERVED

5.6.9 Sub block 0x0128 – RESERVED

5.6.10 Sub block 0x0129 – RESERVED

5.6.11 Sub block 0x012A – RESERVED

5.6.12 Sub block 0x012B – RESERVED

5.6.13 Sub block 0x012C – AWR_RX_GAIN_TEMPLUT_GET_SB

This API is issued to read the temperature based RX gain LUT used by the firmware. This API should be issued after the profile configuration API. The acknowledgement packet sent in response to this API will contain the LUT. The structure is same as AWR_RX_GAIN_LUT_SET_SB.

Table 5.38 – AWR_RX_GAIN_TEMPLUT_GET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x012C
SBLKLEN	2	Value = 8
PROFILE_INDX	1	Profile index for which the RX gain LUT is desired
RESERVED	3	0x000000

5.6.14 Sub block 0x012D – AWR_TX_GAIN_TEMPLUT_GET_SB

This API is issued to read the temperature based TX gain LUT used by the firmware. This API should be issued after the profile configuration API. The acknowledgement packet sent in response to this API will contain the LUT. The structure is same as AWR_TX_GAIN_LUT_SET_SB.

Table 5.39 – AWR_RX_GAIN_TEMPLUT_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x012D
SBLKLEN	2	Value = 8
PROFILE_INDX	1	Profile index for which the TX gain LUT is desired
RESERVED	3	0x000000

5.7 Sub blocks related to AWR_FRAME_TRIG_MSG

5.7.1 Sub block 0x0140 – AWR_FRAMESTARTSTOP_CONF_SB

This sub block starts or stops transmission of frames.

Table 5.40 describes the contents of this sub block.

Table 5.40 – AWR_FRAMESTARTSTOP_CONF_SB contents

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x0140	
SBLKLEN	2	Value = 8	
START_STOP_CMD	2	0x0000	Stop the transmission of frames after the current frame is over
		0x0001	Trigger a frame in software triggered mode. In hardware SYNC_IN triggered mode, this command allows subsequent SYNC_IN trigger to be honored
RESERVED	2	0x0000	

5.8 Sub blocks related to AWR_RF_ADVANCED_FEATURES_CONF_SET_MSG

5.8.1 Sub block 0x0180 – AWR_BPM_COMMON_CONF_SET_SB

This API sub block defines static configurations related to BPM (Binary Phase Modulation) feature in each of the TXs. E.g. the source of the BPM pattern (one constant value for each chirp as defined, or intra-chirp pseudo random BPM pattern as found by a programmable LFSR or a programmable sequence inside each chirp), are defined here.

Table 5.41 describes the contents of this sub block.

Table 5.41 – AWR_BPM_COMMON_CONF_SET_SB contents

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x0180	
SBLKLEN	2	Value = 20	
BPM_MODE_CFG	2	b1:0	BPM_SRC_SEL (select source of BPM pattern) 00 CHIRP_CONFIG_BPM (refer to AWR_BPM_CHIRP_CONF_SB) 01 RESERVED 10 RESERVED 11 RESERVED
		b15:2	Reserved
RESERVED	2	b15:0	0x0000
RESERVED	2	b15:0	0x0000
RESERVED	2	b15:0	0x0000
RESERVED	4	b31:0	0x00000000
RESERVED	4	b31:0	0x00000000

5.8.2 *Sub block 0x0181 – AWR_BPM_CHIRP_CONF_SET_SB*

This sub block defines static configurations related to BPM (Binary Phase Modulation) feature in each of the TXs.

Table 5.42 describes the contents of this sub block.

Table 5.42 – AWR_BPM_CHIRP_CONF_SET_SB contents

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x0181	
SBLKLEN	2	Value = 12	
CHIRP_START_INDX	2	b15:0	Start index of the chirp for configuring the constant BPM Valid range 0 to 511
CHIRP_END_INDX	2	b15:0	End index of the chirp for configuring the constant BPM Valid range 0 to 511
CONST_BPM_VAL	2	b0	CONST_BPM_VAL_TX0_TXOFF Value of Binary Phase Shift value for TX0, when during idle time

		b1	CONST_BPM_VAL_TX0_TXON Value of Binary Phase Shift value for TX0, during chirp
		b2	CONST_BPM_VAL_TX1_TXOFF For TX1
		b3	CONST_BPM_VAL_TX1_TXON For TX1
		b4	CONST_BPM_VAL_TX2_TXOFF For TX2
		b5	CONST_BPM_VAL_TX2_TXON For TX2
		b15:6	Reserved
RESERVED	2	0x0000	

5.9 Sub blocks related to AWR_RF_STATUS_GET_MSG

5.9.1 Sub block 0x0220 – AWR_RF_VERSION_GET_SB

This sub block reads RF HW and FW versions. The information returned by the device will be in the format as given in AWR_RFVERSION_SB.

TABLE 5.43 describes the contents of the request sub block

Table 5.43 – AWR_RF_VERSION_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x0220
SBLKLEN	2	Value = 4

Response to AWR_RFVERSION_GET_SB

AWR_RFVERSION_SB sub block is sent by the radar device in response to AWR_RFVERSION_GET_SB. Note that SBLKID for both AWR_RFVERSION_GET_SB and AWR_RFVERSION_SB are same.

Table 5.44 describes the contents of the response sub block.

Table 5.44 – AWR_RF_VERSION_SB contents

Field Name	Number of bytes	Description

SBLKID	2	Value = 0x0220
SBLKLEN	2	Value = 20
HW_VARIANT	1	HW variant number
HW_VERSION_MAJOR	1	HW version major number
HW_VERSION_MINOR	1	HW version minor number
BSS_FW_VERSION_MAJOR	1	BSS FW version major number
BSS_FW_VERSION_MINOR	1	BSS FW version minor number
BSS_FW_VERSION_BUILD	1	BSS FW version build number
BSS_FW_VERSION_DEBUG	1	BSS FW version debug number
BSS_FW_VERSION_YEAR	1	Year of BSS FW version release
BSS_FW_VERSION_MONTH	1	Month of BSS FW version release
BSS_FW_VERSION_DAY	1	Day of BSS FW version release
BSS_FW_VERSION_PATCH_MAJOR	1	BSS FW version patch major number
BSS_FW_VERSION_PATCH_MINOR	1	BSS FW version patch minor number
BSS_FW_VERSION_PATCH_YEAR	1	Year of BSS FW patch release
BSS_FW_VERSION_PATCH_MONTH	1	Month of BSS FW patch release
BSS_FW_VERSION_PATCH_DAY	1	Day of BSS FW patch release
RESERVED	1	0x00

5.9.2 Sub block 0x0221 – AWR_RF_CPUFAULT_STATUS_GET_SB

This sub block provides the RF BSS CPU fault information.

Table 5.45 describes the content of this sub block.

Table 5.45 – AWR_RF_CPUFAULT_STATUS_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x0221
SBLKLEN	2	Value = 4

Response to AWR_RF_CPUFAULT_STATUS_GET_SB

AWR_RF_CPUFAULT_STATUS_SB is sent in response to AWR_RF_CPUFAULT_STATUS_GET_SB.

Table 5.46 describes the content of AWR_RF_CPUFAULT_STATUS_SB

Table 5.46 – AWR_RF_CPUFAULT_STATUS_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x0221
SBLKLEN	2	Value = 36
FAULT_TYPE	1	0 RF Processor Undefined Instruction Abort 1 RF Processor Instruction pre-fetch Abort 2 RF Processor Data Access Abort 3 RF Processor Firmware Fatal Error 0x4 – 0xFE Reserved 0xFF No fault
RESERVED	1	0x00
LINE_NUM	2	Valid only in case of FAULT type is 0x3, provides the firmware line number at which fatal error occurred.
FAULT_LR	4	The instruction PC address at which Fault occurred
FAULT_PREV_LR	4	The return address of the function from which fault function has been called (Call stack LR)
FAULT_SPSR	4	The CPSR register value at which fault occurred
FAULT_SP	4	The SP register value at which fault occurred
FAULT_CAUSE_ADDRESS	4	The address access at which Fault occurred (valid only for fault type 0x0 to 0x2)

FAULT_ERROR_ STATUS	2	The status of Error (Error Cause type – valid only for fault type 0x0 to 0x2) 0x000 BACKGROUND_ERR 0x001 ALIGNMENT_ERR 0x002 DEBUG_EVENT 0x00D PERMISSION_ERR 0x008 SYNCH_EXTER_ERR 0x406 ASYNCH_EXTER_ERR 0x409 SYNCH_ECC_ERR 0x408 ASYNCH_ECC_ERR
FAULT_ERROR_ SOURCE	1	The Source of the Error (Error Source type - valid only for fault type 0x0 to 0x2) 0x0 ERR_SOURCE_AXI_MASTER 0x1 ERR_SOURCE_ATCM 0x2 ERR_SOURCE_BTCM
FAULT_AXI_ ERROR_TYPE	1	The AXI Error type (Error Source type - valid only for fault type 0x0 to 0x2) 0x0 AXI_DECOD_ERR 0x1 AXI_SLAVE_ERR
FAULT_ACCESS_ TYPE	1	The Error Access type (Error Access type - valid only for fault type 0x0 to 0x2) 0x0 READ_ERR 0x1 WRITE_ERR
FAULT_RECOVERY _TYPE	1	The Error Recovery type (Error Recovery type - Valid only for fault type 0x0 to 0x2) 0x0 UNRECOVERY 0x1 RECOVERY
RESERVED	2	0x0000

5.9.3 *Sub block 0x0222 – AWR_RF_ESMFAULT_STATUS_GET_SB*

This sub block provides the information regarding additional RF sub system faults.

Table 5.47 describes the content of this sub block.

Table 5.47 – AWR_RF_ESMFAULT_STATUS_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x0222
SBLKLEN	2	Value = 4

The response to above request is given in the AWR_RF_ESMFAULT_STATUS_SB.

Table 5.48 describes the contents of AWR_RF_ESMFAULT_STATUS_SB.

Table 5.48 – AWR_RF_ESMFAULT_STATUS_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x0222
SBLKLEN	2	Value = 12

ESM_GROUP1_ ERRORS	4	**Bit** **Error Information** 0 ESM_G1_RAMPGEN_SB_ERR 1 RESERVED 2 ESM_G1_GPADC_RAM_SB_ERR 3 ESM_G1_VIM_RAM_SB_ERR 4 ESM_G1_DFE_SELFTEST_ERR 5 ESM_G1_VIM_SELFTEST_ERR 6 ESM_G1_B0TCM_SB_ERR 7 ESM_G1_B1TCM_SB_ERR 8 ESM_G1_CCMR4_SELFTEST_ERR 9 ESM_G1_ATCM_SB_ERR 10 ESM_G1_RAMPGEN_SELFTEST_ERR 11 ESM_G1_RAMPGEN_PAR_SELFTST_ERR 12 ESM_G1_SEQ_EXT_SELFTEST_ERR 13 ESM_G1_SEQ_EXT_SB_ERR 14 RESERVED 15 ESM_G1_AGC_RAM_SB_ERR 16 ESM_G1_B1TCM_PAR_CHK_ERR 17 ESM_G1_B0TCM_PAR_CHK_ERR 18 ESM_G1_ATCM_PAR_CHK_ERR 19 ESM_G1_MB_MSS2BSS_SB_ERR 20 ESM_G1_MB_BSS2MSS_SB_ERR 21-31 RESERVED
ESM_GROUP2_ ERRORS	4	**Bit** **Error Information** 0 ESM_G2_DFE_STC_ERR 1 ESM_G2_CR4_STC_ERR 2 ESM_G2_CCMR4_COMP_ERR 3 ESM_G2_B0TCM_DB_ERR 4 ESM_G2_B1TCM_DB_ERR 5 ESM_G2_ATCM_DB_ERR 6 ESM_G2_DCC_ERR 7 ESM_G2_SEQ_EXT_ERR 8 ESM_G2_SYNT_FREQ_MON_ERR 9 ESM_G2_DFE_PARITY_ERR

		10 ESM_G2_RAMPGEN_DB_ERR 11 ESM_G2_BUBBLE_CORRECTION_FAIL 12 ESM_G2_RAMPGEN_LOCSTEP_ERR 13 ESM_G2_RTI_RESET_ERR 14 ESM_G2_GPADC_RAM_DB_ERR 15 ESM_G2_VIM_COMP_ERR 16 ESM_G2_CR4_LIVE_LOCK_ERR 17 ESM_G2_WDT_NMI_ERR 18 ESM_G2_VIM_RAM_DB_ERR 19 ESM_G2_RAMPGEN_PAR_ERR 20 ESM_G2_SEQ_EXT_DB_ERR 21 ESM_G2_DMA_MPU_ERR 22 ESM_G2_AGC_RAM_DB_ERR 23 ESM_G2_CRC_COMP_ERR 24 ESM_G2_WAKEUP_STS_ERR 25 ESM_G2_SHORT_CIRCUIT_ERR 26 ESM_G2_B1TCM_PAR_ERR 27 ESM_G2_B0TCM_PAR_ERR 28 ESM_G2_ATCM_PAR_ERR 29 ESM_G2_MB_MSS2BSS_DB_ERR 30 ESM_G2_MB_BSS2MSS_DB_ERR 31 ESM_G2_CCC_ERR

5.9.4 Sub block 0x0223 – RESERVED

5.9.5 Sub block 0x0224 – AWR_RF_BOOTUPBIST_STATUS_GET_SB

This sub block provides the information regarding boot up self-test status.

Table 5.49 describes the content of this sub block.

Table 5.49 – AWR_RF_BOOTUPBIST_STATUS_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x224

SBLKLEN	2	Value = 4

The response of this sub block will be AWR_RF_BOOTUPBIST_STATUS_DATA_SB with content as shown in Table 5.50.

Table 5.50 – AWR_RF_BOOTUPBIST_STATUS_DATA_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x224
SBLKLEN	2	Value = 20
RF_POWERUP_ BIST_STATUS_ FLAGS	4	1 – PASS, 0 – FAIL Bit Status Information 0 ROM CRC check 1 CR4 and VIM lockstep test 2 RESERVED 3 VIM test 4 STC test of diagnostic 5 CR4 STC 6 CRC test 7 RAMPGEN memory ECC test 8 DFE Parity test 9 DFE memory ECC 10 RAMPGEN lockstep test 11 FRC lockstep test 12 DFE memory PBIST 13 RAMPGEN memory PBIST 14 PBIST test 15 WDT test 16 ESM test 17 DFE STC 18 RESERVED 19 ATCM, BTCM ECC test 20 ATCM, BTCM parity test 21 RESERVED 22 RESERVED 23 RESERVED

		24 FFT test 25 RTI test 26 PCR test 27-31 RESERVED
POWERUP_TIME	4	RF BIST SS power up time 1 LSB = 5ns
RESERVED	4	0x00000000
RESERVED	4	0x00000000

5.10 Sub blocks related to AWR_RF_MONITORING_REPORT_GET_MSG

5.10.1 Sub block 0x0260 – AWR_RF_DFE_STATISTICS_REPORT_GET_SB

Table 5.51 describes the content of this sub block.

Table 5.51 – AWR_RF_DFE_STATISTICS_REPORT_GET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x260
SBLKLEN	2	Value = 4

The response of this sub block will be AWR_RF_DFE_STATISTICS_REPORT_SB with content as shown in Table 5.52.

Table 5.52 – AWR_RF_DFE_STATISTICS_REPORT_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x260
SBLKLEN	2	Value = 196
PF0_RX0_ICH	2	Residual DC value in I chain for profile 0, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF0_RX0_QCH	2	Residual DC value in Q chain for profile 0, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF0_RX0_ISQ	2	RMS power in I chain for profile 0, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number

		1LSB = 1V^2/2^{15} referred to ADC input
PF0_RX0_QSQ	2	RMS power in Q chain for profile 0, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF0_RX0_IQCORR	4	Cross correlation between I and Q chains for profile 0, RX channel 0 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = 1V^2/2^{30} referred to ADC input
PF0_RX1_ICH	2	Residual DC value in I chain for profile 0, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF0_RX1_QCH	2	Residual DC value in Q chain for profile 0, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF0_RX1_ISQ	2	RMS power in I chain for profile0, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF0_RX1_QSQ	2	RMS power in Q chain for profile 0, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF0_RX1_IQCORR	4	Cross correlation between I and Q chains for profile 0, RX channel 1 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = 1V^2/2^{30} referred to ADC input
PF0_RX2_ICH	2	Residual DC value in I chain for profile 0, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF0_RX2_QCH	2	Residual DC value in Q chain for profile 0, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF0_RX2_ISQ	2	RMS power in I chain for profile 0, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF0_RX2_QSQ	2	RMS power in Q chain for profile 0, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF0_RX2_IQCORR	4	Cross correlation between I and Q chains for profile 0, RX channel 2 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = 1V^2/2^{30} referred to ADC input

PF0_RX3_ICH	2	Residual DC value in I chain for profile 0, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF0_RX3_QCH	2	Residual DC value in Q chain for profile 0, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF0_RX3_ISQ	2	RMS power in I chain for profile 0, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF0_RX3_QSQ	2	RMS power in Q chain for profile 0, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF0_RX3_IQCORR	4	Cross correlation between I and Q chains for profile 0, RX channel 3 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = 1V^2/2^{30} referred to ADC input
PF1_RX0_ICH	2	Residual DC value in I chain for profile 1, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF1_RX0_QCC	2	Residual DC value in Q chain for profile 1, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF1_RX0_ISQ	2	RMS power in I chain for profile 1, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF1_RX0_QSQ	2	RMS power in Q chain for profile 1, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF1_RX0_IQCORR	4	Cross correlation between I and Q chains for profile 1, RX channel 0 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = 1V^2/2^{30} referred to ADC input
PF1_RX1_ICH	2	Residual DC value in I chain for profile 1, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF1_RX1_QCH	2	Residual DC value in Q chain for profile 1, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF1_RX1_ISQ	2	RMS power in I chain for profile 1, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number

		1LSB = 1V^2/2^{15} referred to ADC input
PF1_RX1_QSQ	2	RMS power in Q chain for profile 1, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF1_RX1_IQCORR	4	Cross correlation between I and Q chains for profile 1, RX channel 1 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = 1V^2/2^{30} referred to ADC input
PF1_RX2_ICH	2	Residual DC value in I chain for profile 1, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF1_RX2_QCH	2	Residual DC value in Q chain for profile 1, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF1_RX2_ISQ	2	RMS power in I chain for profile 1, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF1_RX2_QSQ	2	RMS power in Q chain for profile 1, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF1_RX2_IQCORR	4	Cross correlation between I and Q chains for profile 1, RX channel 2 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = 1V^2/2^{30} referred to ADC input
PF1_RX3_ICH	2	Residual DC value in I chain for profile 1, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF1_RX3_QCH	2	Residual DC value in Q chain for profile 1, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = 1V/2^{15} referred to ADC input
PF1_RX3_ISQ	2	RMS power in I chain for profile 1, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF1_RX3_QSQ	2	RMS power in Q chain for profile 1, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = 1V^2/2^{15} referred to ADC input
PF1_RX3_IQCORR	4	Cross correlation between I and Q chains for profile 1, RX channel 3 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = 1V^2/2^{30} referred to ADC input

PF2_RX0_ICH	2	Residual DC value in I chain for profile 2, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF2_RX0_QCC	2	Residual DC value in Q chain for profile 2, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF2_RX0_ISQ	2	RMS power in I chain for profile 2, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF2_RX0_QSQ	2	RMS power in Q chain for profile 2, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF2_RX0_IQCORR	4	Cross correlation between I and Q chains for profile 2, RX channel 0 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = $1V^2/2^{30}$ referred to ADC input
PF2_RX1_ICH	2	Residual DC value in I chain for profile 2, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF2_RX1_QCH	2	Residual DC value in Q chain for profile 2, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF2_RX1_ISQ	2	RMS power in I chain for profile 2, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF2_RX1_QSQ	2	RMS power in Q chain for profile 2, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF2_RX1_IQCORR	4	Cross correlation between I and Q chains for profile 2, RX channel 1 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = $1V^2/2^{30}$ referred to ADC input
PF2_RX2_ICH	2	Residual DC value in I chain for profile 2, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF2_RX2_QCH	2	Residual DC value in Q chain for profile 2, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF2_RX2_ISQ	2	RMS power in I chain for profile 2, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number

		1LSB = $1V^2/2^{15}$ referred to ADC input
PF2_RX2_QSQ	2	RMS power in Q chain for profile 2, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF2_RX2_IQCORR	4	Cross correlation between I and Q chains for profile 2, RX channel 2 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = $1V^2/2^{30}$ referred to ADC input
PF2_RX3_ICH	2	Residual DC value in I chain for profile 2, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF2_RX3_QCH	2	Residual DC value in Q chain for profile 2, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF2_RX3_ISQ	2	RMS power in I chain for profile 2, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF2_RX3_QSQ	2	RMS power in Q chain for profile 2, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF2_RX3_IQCORR	4	Cross correlation between I and Q chains for profile 2, RX channel 3 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = $1V^2/2^{30}$ referred to ADC input
PF3_RX0_ICH	2	Residual DC value in I chain for profile 3, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF3_RX0_QCC	2	Residual DC value in Q chain for profile 3, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF3_RX0_ISQ	2	RMS power in I chain for profile 3, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF3_RX0_QSQ	2	RMS power in Q chain for profile 3, RX channel 0 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF3_RX0_IQCORR	4	Cross correlation between I and Q chains for profile 3, RX channel 0 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = $1V^2/2^{30}$ referred to ADC input

PF3_RX1_ICH	2	Residual DC value in I chain for profile 3, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF3_RX1_QCH	2	Residual DC value in Q chain for profile 3, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF3_RX1_ISQ	2	RMS power in I chain for profile 3, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF3_RX1_QSQ	2	RMS power in Q chain for profile 3, RX channel 1 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF3_RX1_IQCORR	4	Cross correlation between I and Q chains for profile 3, RX channel 1 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = $1V^2/2^{30}$ referred to ADC input
PF3_RX2_ICH	2	Residual DC value in I chain for profile 3, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF3_RX2_QCH	2	Residual DC value in Q chain for profile 3, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF3_RX2_ISQ	2	RMS power in I chain for profile 3, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF3_RX2_QSQ	2	RMS power in Q chain for profile 3, RX channel 2 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF3_RX2_IQCORR	4	Cross correlation between I and Q chains for profile 3, RX channel 2 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = $1V^2/2^{30}$ referred to ADC input
PF3_RX3_ICH	2	Residual DC value in I chain for profile 3, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF3_RX3_QCH	2	Residual DC value in Q chain for profile 3, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit signed number 1LSB = $1V/2^{15}$ referred to ADC input
PF3_RX3_ISQ	2	RMS power in I chain for profile 3, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number

		1LSB = $1V^2/2^{15}$ referred to ADC input
PF3_RX3_QSQ	2	RMS power in Q chain for profile 3, RX channel 3 (post DC and IQ mismatch correction) represented by a 16 bit unsigned number 1LSB = $1V^2/2^{15}$ referred to ADC input
PF3_RX3_IQCORR	4	Cross correlation between I and Q chains for profile 3, RX channel 3 (post DC and IQ mismatch correction) represented by a 32 bit signed number 1LSB = $1V^2/2^{30}$ referred to ADC input

5.11 Sub blocks related to AWR_RF_MISC_CONF_SET_MSG

5.11.1 Sub block 0x02C0 – RESERVED

5.11.2 Sub block 0x02C1 – RESERVED

5.11.3 Sub block 0x02C2 – AWR_RF_TEST_SOURCE_CONFIG_SET_SB

This sub block is used to configure the test source of BSS

Table 5.53 describes the content of this sub block.

Table 5.53 – AWR_RF_TEST_SOURCE_CONFIG_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x02C2
SBLKLEN	2	Value = 72
POSITION_VEC1	[2+2+2],	Relative position in Cartesian coordinate from radar to objects, [x, y, z] (all signed, though for y, only unsigned makes sense if forward looking: our radar is on y=0 plane). Object 0 [x,y,z] 1 LSB = 1 cm Valid Range: y: 0 to 32767 cm, x & z: +/-32767 cm
VELOCITY_VEC1	[2+2+2],	Relative velocity in Cartesian coordinate, similar to position vector (all signed) Object 0 1 LSB = 1 cm/s Valid Range = +/- 5000 (i.e. +/-180 kmph)

SIG_LEV_VEC1	[2]	Reflecting objects' signal level at ADC output, relative to ADC Full Scale 1 LSB = –0.1dBFS Valid range: 0 to 950 The same field may be used to emulate enable/disable each object by programming appropriate levels.
BOUNDARY_MIN_VEC1	[2+2+2],	Boundary minimum limit for each of x, y, z. When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Object 1 [x,y,z] 1 LSB = 1 cm Valid Range: x: 0 to 32767 cm, y & z: +/-32767 cm
BOUNDARY_MAX_VEC1	[2+2+2],	Boundary maximum limit for each of x, y, z. When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Object 1 [x,y,z] 1 LSB = 1 cm Valid Range: x: 0 to 32767 cm, y & z: +/-32767 cm
POSITION_VEC2	[2+2+2],	Relative position in Cartesian coordinate from radar to objects, [x, y, z] (all signed, though for y, only unsigned makes sense if forward looking: our radar is on y=0 plane). Object 1 [x,y,z] 1 LSB = 1 cm Valid Range: y: 0 to 32767 cm, x & z: +/-32767 cm
VELOCITY_VEC2	[2+2+2],	Relative velocity in Cartesian coordinate, similar to position vector (all signed) Object 1 1 LSB = 1 cm/s Valid Range = +/- 5000 (i.e. +/-180 kmph)
SIG_LEV_VEC2	[2]	Reflecting objects' signal level at ADC output, relative to ADC Full Scale 1 LSB = –0.1dBFS Valid range: 0 to 950 The same field may be used to emulate enable/disable each object by programming appropriate levels.

BOUNDARY_MIN_VEC2	[2+2+2],	Boundary minimum limit for each of x, y, z. When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Object 1 [x,y,z] 1 LSB = 1 cm Valid Range: x: 0 to 32767 cm, y & z: +/-32767 cm
BOUNDARY_MAX_VEC2	[2+2+2],	Boundary maximum limit for each of x, y, z. When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Object 1 [x,y,z] 1 LSB = 1 cm Valid Range: x: 0 to 32767 cm, y & z: +/-32767 cm
RX_ANT_POS_XZ	8	Receiver Antenna positions to be modeled. The radar is on y=0 plane. Only x and z coordinates to be provided. 1LSB = Wavelength / 8 Valid range = +/-15 wave lengths Byte 0: RX0 X coordinate (may be 0 as reference) Byte 1: RX0 Z (may be 0 as reference) Byte 2: RX1 X Byte 3: RX1 Z Byte 4: RX2 X Byte 5: RX2 Z Byte 6: RX3 X Byte 7: RX3 Z
RESERVED	6	RESERVED
RESERVED	2	Reserved for 4 bytes alignment

5.11.4 *Sub block 0x02C3 – AWR_RF_TEST_SOURCE_ENABLE_SET_SB*

This sub block is used to enable test source of BSS

Table 5.54 describes the content of this sub block.

Table 5.54 – AWR_RF_TEST_SOURCE_ENABLE_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x02C3

SBLKLEN	2	Value = 8	
TS_EN	2	b0	0: Disable (revert to normal functionality) 1: Enable (enter test source functionality)
		b15:1	RESERVED
RESERVED	2	0x0000	

5.11.5 *Sub block 0x02C4 – 0x02CB RESERVED*

5.11.6 *Sub block 0x02CC – AWR_RF_LDO_BYPASS_SB*

This sub block enables LDO bypass option within BSS.

CAUTION: Do not enable RF LDO bypass option when the PMIC is configured to supply 1.3V to VIN_13RF1 and VIN_13RF2 analog and RF power supply inputs. This may damage the device. Typically in TI EVMs, PMIC is configured to supply 1.3V to the RF supplies.

Table 5.55 describes the content of this sub block.

Table 5.55 – AWR_RF_LDO_BYPASS_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x02CC
SBLKLEN	2	Value = 8
RFLDOBYPASS_EN	2	0 RF LDO not bypassed 1 RF LDO bypassed
SUPPLY_ MONITOR_IRDROP	2	IR drop is the voltage drop from the PMIC output to the device pin. The user should program the voltage drop in percentage units which will be used for adjusting the thresholds for measuring the external supplies. Value Description 0 IR drop of 0% 1 IR drop of 3% 2 IR drop of 6% 3 IR drop of 9%

5.11.7 Sub block 0x02CD – AWR_RF_PALOOPBACK_CFG_SB

This sub block enables/disables PA loopback for all enabled profiles. This is used to debug both the TX and RX chains are working correctly.

Table 5.56 describes the content of this sub block.

NOTE: If monitoring is enabled with the loopback APIs (subblock 0x02CD, 0x02CE, 0x02CF), then loopback will not work after montoring is complete. To use loopback with monitoring, use AWR_ADVANCED_FRAME_CONF_SB with AWR_LOOPBACK_BURST_CONF_SB.

Table 5.56 – AWR_RF_PALOOPBACK_CFG_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x02CD
SBLKLEN	2	Value = 8
PA_LOOPBACK_FREQ	2	This value is a 100MHz divider which sets the loopback frequency For e.g. for a 1 MHz frequency, set this to 100 For a 2 MHz frequency, set this to 50 Note: To ensure no leakage of signal power, user has to ensure that 100MHz/LOOPBACK_FREQ is an integer multiple of bin width For e.g. if user choses 25Msps sampling rate and 2048 samples/chirp, then LOOPBACK_FREQ of 64 (=1.5625MHz) will ensure no leakage
PA_LOOPBACK_EN	1	0 PA loopback is not enabled 1 PA loopback is enabled
RESERVED	1	0x00

5.11.8 Sub block 0x02CE – AWR_RF_PSLOOPBACK_CFG_SB

This sub block enables/disables PS (phase shifter) loopback for all enabled profiles. This is used to debug the TX (before the PA) and RX chains.

Table 5.57 describes the content of this sub block.

NOTE: If monitoring is enabled with the loopback APIs (subblock 0x02CD, 0x02CE, 0x02CF), then loopback will not work after montoring is complete. To use loopback with monitoring, use AWR_ADVANCED_FRAME_CONF_SB with AWR_LOOPBACK_BURST_CONF_SB.

Table 5.57 – AWR_RF_PSLOOPBACK_CFG_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x02CE
SBLKLEN	2	Value = 12
PS_LOOPBACK_FREQ	2	Loop back frequency in kHz 1 LSB = 1kHz
RESERVED	2	0x0000
PS_LOOPBACK_EN	1	Value Definition 0 PS loopback is not enabled 1 PS loopback is enabled
PS_LOOPBACK_TXID	1	b0 TX0 is used for loopback b1 TX1 is used for loopback b7:2 RESERVED

PGA_GAIN_INDEX	1	Value PGA gain value 0 PGA is OFF 1 -22 dB 2 -16 dB 3 -14 dB 4 -11 dB 5 -9 dB 6 -6 dB 7 -5 dB 8 -4 dB 9 -3 dB 10 -2 dB 11 -1 dB 12 0 dB 13 1 dB 14 2 dB 15 3 dB 16 4 dB 17 5 dB 18 6 dB 19 7 dB 20 8 dB 21 9 dB 255-22 RESERVED
RESERVED	1	0x00

5.11.9 *Sub block 0x02CF – AWR_RF_IFLOOPBACK_CFG_SB*

This sub block enables/disables IF loopback for all enabled profiles. This is used to debug the RX IF chain.

Table 5.58 describes the content of this sub block.

NOTE: If monitoring is enabled with the loopback APIs (subblock 0x02CD, 0x02CE, 0x02CF), then loopback will not work after montoring is complete. To use loopback with monitoring, use AWR_ADVANCED_FRAME_CONF_SB with

	AWR_LOOPBACK_BURST_CONF_SB.

Table 5.58 – AWR_RF_IFLOOPBACK_CFG_SB contents

Field Name	**Number of bytes**	**Description**	
SBLKID	2	Value = 0x02CF	
SBLKLEN	2	Value = 8	
IF_LOOPBACK_FREQ	2	Value	IF Loopback frequency value
		0	180 kHz
		1	240 kHz
		2	360 kHz
		3	720 kHz
		4	1 MHz
		5	2 MHz
		6	2.5 MHz
		7	3 MHz
		8	4.017857 MHz
		9	5 MHz
		10	6 MHz
		11	8.035714 MHz
		12	9 MHz
		13	10 MHz
		65535-14	RESERVED
IF_LOOPBACK_EN	1	0	IF loopback is not enabled
		1	IF loopback is enabled
RESERVED	1	0x00	

5.11.10 Sub block 0x02D0 – AWR_RF_GPADC_CFG_SET_SB

This sub block enables the GPADC reads for external inputs (available only in AWR1642).

Table 5.59 describes the content of this sub block.

Table 5.59 – AWR_RF_GPADC_CFG_SET_SB contents

Field Name	**Number of bytes**	**Description**

SBLKID	2	Value = 0x02D0	
SBLKLEN	2	Value = 8	
GPADC_SENSOR_CFG	2	b0	ANATEST1 0 Disable 1 Enable
		b1	ANATEST2 0 Disable 1 Enable
		b2	ANATEST3 0 Disable 1 Enable
		b3	ANATEST4 0 Disable 1 Enable
		b4	IFORCE 0 Disable 1 Enable
		b5	VSENSE 0 Disable 1 Enable
		b6	IFORCEBUF 0 Disable 1 Enable
		b15:b7	RESERVED
ANATEST1_CFG	2	b7:0 b15:8	Number of samples to collect RESERVED
ANATEST2_CFG	2	b7:0 b15:8	Number of samples to collect RESERVED
ANATEST3_CFG	2	b7:0 b15:8	Number of samples to collect RESERVED
ANATEST4_CFG	2	b7:0 b15:8	Number of samples to collect RESERVED
IFORCE_CFG	2	b7:0 b15:8	Number of samples to collect RESERVED

VSENSE_CFG	2	b7:0 b15:8	Number of samples to collect RESERVED
IFORCEBUF_CFG	2	b7:0 b15:8	Number of samples to collect RESERVED
RESERVED	4	0x00000000	
RESERVED	4	0x00000000	
RESERVED	4	0x00000000	

The response to the AWR_RF_GPADC_CFG_SET_SB is an async event AWR_AE_RF_GPADC_RESULT_DATA_SB which contains the measured values for each of the enabled channels.

5.11.11 Sub block 0x02D1 – RESERVED

5.11.12 Sub block 0x02D2 – RESERVED

5.11.13 Sub block 0x02D3 – RESERVED

5.12 Sub blocks related to AWR_RF_MISC_CONF_GET_MSG

5.12.1 Sub block 0x02E0 to 0x2E9 – RESERVED

5.12.2 Sub block 0x02EA – AWR_RF_TEMPERATURE_GET_SB

This sub block provides the device temperature sensor information.

Table 5.60 describes the content of this sub block.

Table 5.60 – AWR_RF_TEMPERATURE_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x02EA
SBLKLEN	2	Value = 4

Response to AWR_RF_TEMPERATURE_GET_SB

AWR_RF_TEMPERATURE_DATA_SB sub block is sent by the radar device in response to AWR_RF_TEMPERATURE_GET_SB.

Table 5.61 – AWR_RF_TEMPERATURE_DATA_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x2EA
SBLKLEN	2	Value = 28
TIME	4	BSS local Time from device power up 1 LSB = 1 ms
TEMP_RX0_SENS	2	RX0 temperature sensor reading (signed value) 1 LSB = 1 degree C
TEMP_RX1_SENS	2	RX1 temperature sensor reading (signed value) 1 LSB = 1 degree C
TEMP_RX2_SENS	2	RX2 temperature sensor reading (signed value) 1 LSB = 1 degree C
TEMP_RX3_SENS	2	RX3 temperature sensor reading (signed value) 1 LSB = 1 degree C
TEMP_TX0_SENS	2	TX0 temperature sensor reading (signed value) 1 LSB = 1 degree C
TEMP_TX1_SENS	2	TX1 temperature sensor reading (signed value) 1 LSB = 1 degree C
TEMP_TX2_SENS	2	TX2 temperature sensor reading (signed value) 1 LSB = 1 degree C
TEMP_PM_SENS	2	PM temperature sensor reading (signed value) 1 LSB = 1 degree C
TEMP_DIG_SENS	2	Digital temperature sensor reading (signed value) 1 LSB = 1 degree C
RESERVED	2	0x0000

5.13 Sub blocks related to AWR_RF_ASYNC_EVENT_MSG1

5.13.1 Sub block 0x1000 – RESERVED

5.13.2 Sub block 0x1001 – RESERVED

5.13.3 Sub block 0x1002 – AWR_AE_RF_CPUFAULT_SB

This sub block indicates CPU fault status of BIST SS.

Table 5.62 describes the content of this sub block.

Table 5.62 – AWR_AE_RF_CPUFAULT_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1002
SBLKLEN	2	Value = 36
FAULT_TYPE	1	0 RF Processor Undefined Instruction Abort 1 RF Processor Instruction pre-fetch Abort 2 RF Processor Data Access Abort 3 RF Processor Firmware Fatal Error 0x4 – 0xFE Reserved 0xFF No fault
RESERVED	1	0x00
LINE_NUM	2	Valid only in case of FAULT type is 0x3, provides the firmware line number at which fatal error occurred.
FAULT_LR	4	The instruction PC address at which Fault occurred
FAULT_PREV_LR	4	The return address of the function from which fault function has been called (Call stack LR)
FAULT_SPSR	4	The CPSR register value at which fault occurred
FAULT_SP	4	The SP register value at which fault occurred
FAULT_CAUSE_ ADDRESS	4	The address access at which fault occurred (valid only for fault type 0x0 to 0x2)

FAULT_ERROR_ STATUS	2	The status of Error (Error Cause type - valid only for fault type 0x0 to 0x2) 0x000 BACKGROUND_ERR 0x001 ALIGNMENT_ERR 0x002 DEBUG_EVENT 0x00D PERMISSION_ERR 0x008 SYNCH_EXTER_ERR 0x406 ASYNCH_EXTER_ERR 0x409 SYNCH_ECC_ERR 0x408 ASYNCH_ECC_ERR
FAULT_ERROR_ SOURCE	1	The Source of the Error (Error Source type - valid only for fault type 0x0 to 0x2) 0x0 ERR_SOURCE_AXI_MASTER 0x1 ERR_SOURCE_ATCM 0x2 ERR_SOURCE_BTCM
FAULT_AXI_ ERROR_TYPE	1	The AXI Error type (Error Source type - valid only for fault type 0x0 to 0x2) 0x0 AXI_DECOD_ERR 0x1 AXI_SLAVE_ERR
FAULT_ACCESS_ TYPE	1	The Error Access type (Error Access type - valid only for fault type 0x0 to 0x2) 0x0 READ_ERR 0x1 WRITE_ERR
FAULT_RECOVERY _TYPE	1	The Error Recovery type (Error Recovery type - Valid only for fault type 0x0 to 0x2) 0x0 UNRECOVERY 0x1 RECOVERY
RESERVED	2	0x0000

5.13.4 *Sub block 0x1003 – AWR_AE_RF_ESMFAULT_SB*

This sub block indicates the status of any other faults in the BIST SS.

Table 5.63 describes the content of this sub block.

Table 5.63 – AWR_AE_RF_ESMFAULT_STATUS_SB contents

Field Name	**Number**	**Description**

	of bytes		
SBLKID	2	Value = 0x1003	
SBLKLEN	2	Value = 12	
ESM_GROUP1_ ERRORS	4	Bit	Error Information
		0	ESM_G1_RAMPGEN_SB_ERR
		1	RESERVED
		2	ESM_G1_GPADC_RAM_SB_ERR
		3	ESM_G1_VIM_RAM_SB_ERR
		4	ESM_G1_DFE_SELFTEST_ERR
		5	ESM_G1_VIM_SELFTEST_ERR
		6	ESM_G1_B0TCM_SB_ERR
		7	ESM_G1_B1TCM_SB_ERR
		8	ESM_G1_CCMR4_SELFTEST_ERR
		9	ESM_G1_ATCM_SB_ERR
		10	ESM_G1_RAMPGEN_SELFTEST_ERR
		11	ESM_G1_RAMPGEN_PAR_SELFTST_ERR
		12	ESM_G1_SEQ_EXT_SELFTEST_ERR
		13	ESM_G1_SEQ_EXT_SB_ERR
		14	RESERVED
		15	ESM_G1_AGC_RAM_SB_ERR
		16	ESM_G1_B1TCM_PAR_CHK_ERR
		17	ESM_G1_B0TCM_PAR_CHK_ERR
		18	ESM_G1_ATCM_PAR_CHK_ERR
		19	ESM_G1_MB_MSS2BSS_SB_ERR
		20	ESM_G1_MB_BSS2MSS_SB_ERR
		21-31	RESERVED
ESM_GROUP2_ ERRORS	4	Bit	Error Information
		0	ESM_G2_DFE_STC_ERR
		1	ESM_G2_CR4_STC_ERR
		2	ESM_G2_CCMR4_COMP_ERR
		3	ESM_G2_B0TCM_DB_ERR
		4	ESM_G2_B1TCM_DB_ERR
		5	ESM_G2_ATCM_DB_ERR
		6	ESM_G2_DCC_ERR
		7	ESM_G2_SEQ_EXT_ERR

		8	ESM_G2_SYNT_FREQ_MON_ERR
		9	ESM_G2_DFE_PARITY_ERR
		10	ESM_G2_RAMPGEN_DB_ERR
		11	ESM_G2_BUBBLE_CORRECTION_FAIL
		12	ESM_G2_RAMPGEN_LOCSTEP_ERR
		13	ESM_G2_RTI_RESET_ERR
		14	ESM_G2_GPADC_RAM_DB_ERR
		15	ESM_G2_VIM_COMP_ERR
		16	ESM_G2_CR4_LIVE_LOCK_ERR
		17	ESM_G2_WDT_NMI_ERR
		18	ESM_G2_VIM_RAM_DB_ERR
		19	ESM_G2_RAMPGEN_PAR_ERR
		20	ESM_G2_SEQ_EXT_DB_ERR
		21	ESM_G2_DMA_MPU_ERR
		22	ESM_G2_AGC_RAM_DB_ERR
		23	ESM_G2_CRC_COMP_ERR
		24	ESM_G2_WAKEUP_STS_ERR
		25	ESM_G2_SHORT_CIRCUIT_ERR
		26	ESM_G2_B1TCM_PAR_ERR
		27	ESM_G2_B0TCM_PAR_ERR
		28	ESM_G2_ATCM_PAR_ERR
		29	ESM_G2_MB_MSS2BSS_DB_ERR
		30	ESM_G2_MB_BSS2MSS_DB_ERR
		31	ESM_G2_CCC_ERR

5.13.5 Sub block 0x1004 – AWR_AE_RF_INITCALIBSTATUS_SB

This sub block indicates the initial calibrations of RF BIST SS are complete.

Table 5.64 describes the content of this sub block.

Table 5.64 – AWR_AE_RF_INITCALIBSTATUS_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1004

SBLKLEN	2	Value = 24
CALIBRATION_ STATUS	4	This field indicates the status of each calibration (0 – FAIL, 1 – PASS). If a particular calibration was not enabled, then its corresponding field should be ignored. **Bit** | **Definition (0 – FAIL, 1 – PASS)** 0 | RESERVED 1 | APLL tuning 2 | SYNTH VCO1 tuning 3 | SYNTH VCO2 tuning 4 | LODIST calibration 5 | RX ADC DC offset calibration 6 | HPF cutoff calibration 7 | LPF cutoff calibration 8 | Peak detector calibration 9 | TX Power calibration 10 | RX gain calibration 11 | TX Phase calibration 12 | RX IQMM calibration 31:13 | RESERVED
CALIBRATION_ UPDATE	4	This field indicates if a particular calibration data has been updated in hardware. (0 – no update, 1 – updated) **Bit** | **Definition** 0 | RESERVED 1 | APLL tuning 2 | SYNTH VCO1 tuning 3 | SYNTH VCO2 tuning 4 | LODIST calibration 5 | RX ADC DC offset calibration 6 | HPF cutoff calibration 7 | LPF cutoff calibration 8 | Peak detector calibration 9 | TX Power calibration 10 | RX gain calibration 11 | TX Phase calibration

		12 — RX IQMM calibration 31:13 — RESERVED
TEMPERATURE	2	Measured temperature, based on average of temperature sensors near all enabled TX and RX channels at the time of calibration. 1 LSB = 1^{o} Celsius
RESERVED	2	0x0000
TIME_STAMP	4	This field indicates time stamp at the time of performing calibration updates. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)
RESERVED	4	0x00000000

5.13.6 Sub block 0x1005 – RESERVED

5.13.7 Sub block 0x1006 – RESERVED

5.13.8 Sub block 0x1007 – RESERVED

5.13.9 Sub block 0x1008 – RESERVED

5.13.10 Sub block 0x1009 – RESERVED

5.13.11 Sub block 0x100A – RESERVED

5.13.12 Sub block 0x100B – AWR_AE_RF_FRAME_TRIGGER_RDY_SB

This sub block indicates that the slave device is now ready to receive the external sync in for frame triggers. In SW triggered mode, this async event indicates that frames are triggered.

Table 5.65 describes the content of this sub block.

Table 5.65 – AWR_AE_RF_FRAME_TRIGGER_RDY_SB contents

Field Name	Number	Description

	of bytes	
SBLKID	2	Value = 0x100B
SBLKLEN	2	Value = 4

5.13.13 Sub block 0x100C – AWR_AE_RF_GPADC_RESULT_DATA_SB

This sub block indicates that GPADC measurement is complete and it also contains the measured data of each of the enabled channels. The data for channels which are not enabled can be ignored.

Table 5.66 describes the content of this sub block.

Table 5.66 – AWR_AE_RF_GPADC_RESULT_DATA_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x100C
SBLKLEN	2	Value = 76
ANATEST1_MIN_ DATA	2	Minimum GPADC reading across the captured samples for ANATEST1 input 1 LSB = 1.8V/1024
ANATEST1_MAX_ DATA	2	Maximum GPADC reading across the captured samples for ANATEST1 input 1 LSB = 1.8V/1024
ANATEST1_AVG_ DATA	2	Average GPADC reading across the captured samples for ANATEST1 input 1 LSB = 1.8V/1024
ANATEST2_MIN_ DATA	2	Minimum GPADC reading across the captured samples for ANATEST2 input 1 LSB = 1.8V/1024
ANATEST2_MAX_ DATA	2	Maximum GPADC reading across the captured samples for ANATEST2 input 1 LSB = 1.8V/1024
ANATEST2_AVG_ DATA	2	Average GPADC reading across the captured samples for ANATEST2 input 1 LSB = 1.8V/1024
ANATEST3_MIN_ DATA	2	Minimum GPADC reading across the captured samples for ANATEST3 input 1 LSB = 1.8V/1024
ANATEST3_MAX_ DATA	2	Maximum GPADC reading across the captured samples for ANATEST3 input

		1 LSB = 1.8V/1024
ANATEST3_AVG_ DATA	2	Average GPADC reading across the captured samples for ANATEST3 input 1 LSB = 1.8V/1024
ANATEST4_MIN_ DATA	2	Minimum GPADC reading across the captured samples for ANATEST4 input 1 LSB = 1.8V/1024
ANATEST4_MAX_ DATA	2	Maximum GPADC reading across the captured samples for ANATEST4 input 1 LSB = 1.8V/1024
ANATEST4_AVG_ DATA	2	Average GPADC reading across the captured samples for ANATEST4 input 1 LSB = 1.8V/1024
IFORCE_MIN_DATA	2	Minimum GPADC reading across the captured samples for IFORCE input 1 LSB = 1.8V/1024
IFORCE_MAX_ DATA	2	Maximum GPADC reading across the captured samples for IFORCE input 1 LSB = 1.8V/1024
IFORCE_AVG_ DATA	2	Average GPADC reading across the captured samples for IFORCE input 1 LSB = 1.8V/1024
VSENSE_MIN_ DATA	2	Minimum GPADC reading across the captured samples for VSENSE input 1 LSB = 1.8V/1024
VSENSE_MAX_ DATA	2	Maximum GPADC reading across the captured samples for VSENSE input 1 LSB = 1.8V/1024
VSENSE_AVG_ DATA	2	Average GPADC reading across the captured samples for VSENSE input 1 LSB = 1.8V/1024
IFORCEBUF_MIN_ DATA	2	Minimum GPADC reading across the captured samples for IFORCEBUF input 1 LSB = 1.8V/1024
IFORCEBUF_MAX_ DATA	2	Maximum GPADC reading across the captured samples for IFORCEBUF input 1 LSB = 1.8V/1024
IFORCEBUF_AVG_ DATA	2	Average GPADC reading across the captured samples for IFORCEBUF input 1 LSB = 1.8V/1024

RESERVED	2	0x0000
RESERVED	4	0x00000000
RESERVED	4	0x00000000
RESERVED	4	0x00000000
RESERVED	4	0x00000000
RESERVED	4	0x00000000
RESERVED	4	0x00000000
RESERVED	4	0x00000000

5.13.14 Sub block 0x100E – RESERVED

5.13.15 Sub block 0x100D – RESERVED

5.13.16 Sub block 0x100E – RESERVED

5.13.17 Sub block 0x100F – AWR_FRAME_END_AE_SB

This sub block indicates end of the frames.

Table 5.67 describes the content of this sub block.

Table 5.67 – AWR_FRAME_END_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x100F
SBLKLEN	2	Value = 4

5.13.18 Sub block 0x1010 – AWR_ANALOGFAULT_AE_SB

This sub block indicates fault in analog supplies or LDO short circuit condition. Once a fault is detected the functionality cannot be resumed from then on and the sensor needs to be re-started.

Table 5.68 – AWR_ANALOGFAULT_AE_SB

Field Name	**Number of bytes**	**Description**

SBLKID	2	Value = 0x1010
SBLKLEN	2	Value = 16
FAULT_TYPE	1	**Value** / **Definition** 0 / NO FAULT 1 / ANALOG_SUPPLY_FAULT Others / RESERVED
RESERVED	1	0x00
RESERVED	2	0x0000
FAULT_SIG	4	**Bit** / **Definition** 0 / 1.8V BB ANA supply fault 1 / 13V/1.0V RF supply fault 2 / Synth LDO short 3 / PA LDO short 31:4 / RESERVED
RESERVED	4	0x00000000

5.13.19 *Sub block 0x1011 – AWR_CAL_MON_TIMING_FAIL_REPORT_AE_SB*

This sub block indicates any timing failure related to calibration or monitoring.

Table 5.69 describes the content of this sub block.

Table 5.69 – AWR_CAL_MON_TIMING_FAIL_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1011
SBLKLEN	2	Value = 8
TIMING_FAILURE_CODE	2	**Bit** / **Definition** b0 / RESERVED b1 / 0 No Failure 1 Total monitoring and calibration time do not fit in one CALIB_MON_TIME_UNIT in AWR_RUN_TIME_CALIBRATION_CONF_A

				ND_TRIGGER when ONE_TIME_CALIB is enabled
		b2	0	No Failure
			1	Total monitoring and calibration time do not fit in one CALIB_MON_TIME_UNIT in AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER when PERIODIC_CALIB is enabled
		b3	0	No Failure
			1	Runtime timing violation: Monitoring functions or calibrations could not be completed in one CALIB_MON_TIME_UNIT
		b15:4	RESERVED	
RESERVED	2	0x0000		

5.13.20 Sub block 0x1012 – AWR_RUN_TIME_CALIB_SUMMARY_REPORT_AE_SB

This sub block indicates the calibration status (one time or run time) if the calibration reports are enabled in the AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB.

NOTE: The calibration report is sent if the calibrations are triggered due to temperature change or whenever the internal calibratons are triggered i.e. every 1 s

Table 5.70 – AWR_RUN_TIME_CALIB_SYMMARY_REPORT_AE_SB contents

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x1012	
SBLKLEN	2	Value = 24	
CALIBRATION_ ERROR_FLAG	4	This field indicates the status of each calibration. 1 - calibration is passed, 0 - calibration is failed or not enabled/performed at least once.	
		Bit	**Definition**
		0	RESERVED
		1	APLL tuning
		2	SYNTH VCO1 tuning
		3	SYNTH VCO2 tuning

		<table><tr><td>4</td><td>LODIST calibration</td></tr><tr><td>5</td><td>RESERVED</td></tr><tr><td>6</td><td>RESERVED</td></tr><tr><td>7</td><td>RESERVED</td></tr><tr><td>8</td><td>RESERVED</td></tr><tr><td>9</td><td>TX Power calibration</td></tr><tr><td>10</td><td>RX gain calibration</td></tr><tr><td>11</td><td>RESERVED</td></tr><tr><td>12</td><td>RESERVED</td></tr><tr><td>31:13</td><td>RESERVED</td></tr></table>
CALIBRATION_ UPDATE_STATUS	4	Each bit corresponding to a calibration indicates if each calibration resulted in a reconfiguration of RF is indicated by a value of 1 in the respective bit in this field. 0 – Analog/RF is not updated 1 – Analog/RF is updated after a respective calibration <table><tr><th>Bit</th><th>Definition</th></tr><tr><td>0</td><td>RESERVED</td></tr><tr><td>1</td><td>APLL tuning</td></tr><tr><td>2</td><td>SYNTH VCO1 tuning</td></tr><tr><td>3</td><td>SYNTH VCO2 tuning</td></tr><tr><td>4</td><td>LODIST calibration</td></tr><tr><td>5</td><td>RESERVED</td></tr><tr><td>6</td><td>RESERVED</td></tr><tr><td>7</td><td>RESERVED</td></tr><tr><td>8</td><td>RESERVED</td></tr><tr><td>9</td><td>TX Power calibration</td></tr><tr><td>10</td><td>RX gain calibration</td></tr><tr><td>11</td><td>RESERVED</td></tr><tr><td>12</td><td>RESERVED</td></tr><tr><td>31:13</td><td>RESERVED</td></tr></table>
TEMPERATURE	2	Measured temperature, based on average of temperature sensors near all enabled TX and RX channels at the time of calibration. 1 LSB = 1^0 Celsius
RESERVED	2	RESERVED

TIME_STAMP	4	This field indicates time stamp at the time of performing calibration updates. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)
RESERVED	4	0x00000000

5.13.21 Sub block 0x1013 – AWR_MONITOR_RF_DIG_LATENTFAULT_REPORT_AE_SB

This async event contains the status of digital monitoring for latent faults.

Table 5.71 – AWR_MONITOR_RF_DIG_LATENTFAULT_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x1013
SBLKLEN	2	Value = 8
DIG_MON_ LATENT_FAULT_ STATUS	4	1 – PASS, 0 – FAIL **Bit** **Definition** 0 RESERVED 1 CR4 and VIM lockstep test 2 RESERVED 3 VIM test 4 RESERVED 5 RESERVED 6 CRC test 7 RAMPGEN memory ECC test 8 DFE Parity test 9 DFE memory ECC test 10 RAMPGEN lockstep test 11 FRC lockstep test 12 RESERVED 13 RESERVED 14 RESERVED 15 RESERVED 16 ESM test 17 DFE STC 18 RESERVED

		19	ATCM, BTCM ECC test
		20	ATCM, BTCM parity test
		21	RESERVED
		22	RESERVED
		23	RESERVED
		24	FFT test
		25	RTI test
		26	PCR test
		31:27	RESERVED

5.13.22 Sub block 0x1014 – RESERVED

5.13.23 Sub block 0x1015 – AWR_MONITOR_REPORT_HEADER_AE_SB

The report header includes common information across all enabled monitors like current FTTI number and current temperature.

Table 5.72 – AWR_MONITORING_REPORT_HEADER_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1015
SBLKLEN	2	Value = 12
FTTI_COUNT	4	FTTI free running counter value, incremented every CAL_MON_TIME_UNIT
AVG_ TEMPERATURE	2	Average temperature at which was monitoring performed
RESERVED	2	0x0000

5.13.24 Sub block 0x1016 – AWR_MONITOR_RF_DIG_PERIODIC_REPORT_AE_SB

This async event is sent periodically to indicate the status of periodic digital monitoring tests.

Table 5.73 – AWR_MONITOR_RF_DIG_PERIODIC_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**

SBLKID	2	Value = 0x1016
SBLKLEN	2	Value = 16
RF_DIG_MON_ PERIODIC_STATUS	4	1 – PASS, 0 – FAIL **Bit** **Monitoring type** 0 PERIODIC_CONFG_REGISTER_READ 1 ESM_MONITORING 2 DFE_STC 3 FRAME_TIMING_MONITORING 31:4 RESERVED
RESERVED	4	0x00000000
TIMESTAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.13.25 Sub block 0x1017 – AWR_MONITOR_TEMPERATURE_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured temperature near various RF analog and digital modules. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.74 – AWR_MONITORING_TEMPERATURE_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1017
SBLKLEN	2	Value = 36

STATUS_FLAGS	2	Status flags indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** – **STATUS_FLAG for monitor** 0 – STATUS_ANA_TEMP_MIN 1 – STATUS_ANA_TEMP_MAX 2 – STATUS_DIG_TEMP_MIN 3 – STATUS_DIG_TEMP_MAX 4 – STATUS_TEMP_DIFF_THRESH 15:5 – RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
TEMPERATURE_ VALUES	20	The measured onchip temperatures are reported here. Byte numbers corresponding to different temperature sensors reported in this field are here: **Bytes** – **SIGNAL** 1:0 – TEMP_RX0 3:2 – TEMP_RX1 5:4 – TEMP_RX2 7:6 – TEMP_RX3 9:8 – TEMP_TX0 11:10 – TEMP_TX1 13:12 – TEMP_TX2 15:14 – TEMP_PM 17:16 – TEMP_DIG1 19:18 – TEMP_DIG2 (Applicable only in AR1642) 1 LSB = 1^0C, signed number
RESERVED	4	0x00000000

TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.13.26 Sub block 0x1018 – AWR_MONITOR_RX_GAIN_PHASE_REPORT_AE_SB

This sub block is a monitoring report which the AWR device sends to the host, containing the measured RX gain and phase values. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.75 – AWR_MONITOR_RX_GAIN_PHASE_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1018
SBLKLEN	2	Value = 72
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit Location** / **STATUS_FLAG for monitor** 0 / STATUS_RX_GAIN_ABS 1 / STATUS_RX_GAIN_MISMATCH 2 / STATUS_RX_GAIN_FLATNESS 3 / STATUS_RX_PHASE_MISMATCH 15:4 / RESERVED 0 – FAIL or check wasn’t done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies.
RESERVED	3	0x000000

Nested table in STATUS_FLAGS description:

Bit Location	**STATUS_FLAG for monitor**
0	STATUS_RX_GAIN_ABS
1	STATUS_RX_GAIN_MISMATCH
2	STATUS_RX_GAIN_FLATNESS
3	STATUS_RX_PHASE_MISMATCH
15:4	RESERVED

RX_GAIN_VALUE	24	The measured RX gain for each enabled channel, at each enabled RF frequency (i.e., lowest, center and highest in the profile's RF band) is reported here. Byte numbers corresponding to different RX and RF, in this field are here: | | RF1 | RF2 | RF3 | |---|---|---|---| | RX0 | 1:0 | 9:8 | 17:16 | | RX1 | 3:2 | 11:10 | 19:18 | | RX2 | 5:4 | 13:12 | 21:20 | | RX3 | 7:6 | 15:14 | 23:22 | 1 LSB = 0.1 dB Only the entries of enabled RF Frequencies and enabled RX channels are valid.
RX_PHASE_VALUE	24	The measured RX phase for each enabled channel, at each enabled RF frequency is reported here. Byte numbers corresponding to different RX and RF, in this field are here: | | RF1 | RF2 | RF3 | |---|---|---|---| | RX0 | 1:0 | 9:8 | 17:16 | | RX1 | 3:2 | 11:10 | 19:18 | | RX2 | 5:4 | 13:12 | 21:20 | | RX3 | 7:6 | 15:14 | 23:22 | 1 LSB = $360°/2^{16}$. Only the entries of enabled RF Frequencies and enabled RX channels are valid. **Note:** these phases include an unknown bias common to all RX channels.
RESERVED	4	0x00000000
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.13.27 Sub block 0x1019 – AWR_MONITOR_RX_NOISE_FIGURE_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured RX noise figure values corresponding to the full IF band of a profile. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.76 – AWR_MONITOR_RX_NOISE_FIGURE_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1019
SBLKLEN	2	Value = 52
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor.

Bit	**STATUS_FLAG for monitor**
0	STATUS_RX_NOISE_FIGURE
15:1	RESERVED

0 – FAIL or check wasn't done
1 – PASS

Field Name	**Number of bytes**	**Description**
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies.
RESERVED	3	0x000000
RX_NOISE_FIGURE_VALUE	24	The measured RX input referred for each enabled channel, at each enabled RF frequency is reported here. Byte numbers corresponding to different RX and RF, in this field are here:

	RF1	RF2	RF3
RX0	1:0	9:8	17:16
RX1	3:2	11:10	19:18
RX2	5:4	13:12	21:20
RX3	7:6	15:14	23:22

1 LSB = 0.1 dB
Only the entries of enabled RF Frequencies and enabled RX channels are valid.

RESERVED	4	0x00000000
RESERVED	4	0x00000000
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.13.28 Sub block 0x101A – AWR_MONITOR_RX_IFSTAGE_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured RX IF filter attenuation values at the given IF frequencies. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.77 – AWR_MONITOR_RX_IFSTAGE_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x101A
SBLKLEN	2	Value = 48
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** – **STATUS_FLAG for monitor** 0 – STATUS_RX_HPF_ERROR 1 – STATUS_RX_LPF_ERROR 2 – STATUS_RX_IFA_GAIN_ERROR 15:3 – RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies.
RESERVED	3	0x000000

HPF_CUTOFF_ FREQ_ ERROR_ VALUE	8	The deviations of RX IFA HPF cutoff frequency from the ideally expected values for all the enabled RX channels are reported here. HPF_CUTOFF_FREQ_ERROR = 100*(Measured Cutoff Frequency / Expected Cutoff Frequency) – 100, for RX IF filter in the HPF region. Byte numbers corresponding to measured cutoff frequency error on different RX channels, in this field are here:

	I channel	Q channel
RX0	0	4
RX1	1	5
RX2	2	6
RX3	3	7

1 LSB = 1%, signed number
Applicable only for the enabled channels.

LPF_CUTOFF_ FREQ_ ERROR_ VALUE	8	The deviations of RX IFA LPF cutoff frequency from the ideally expected values for all the enabled RX channels are reported here. LPF_CUTOFF_FREQ_ERROR = 100*(Measured Cutoff Frequency / Expected Cutoff Frequency) – 100, for RX IF filter in the LPF region. Byte numbers corresponding to measured cutoff frequency error on different RX channels, in this field are here:

	I channel	Q channel
RX0	0	4
RX1	1	5
RX2	2	6
RX3	3	7

1 LSB = 1%, signed number
Applicable only for the enabled channels.

RX_IFA_GAIN_ ERROR_ VALUE	8	The deviations of RX IFA Gain from the ideally expected values for all the enabled RX channels are reported here. Byte numbers corresponding to measured cutoff frequency error on different RX channels and HPF/LPF, in this field are here: (blank) | I channel | Q channel RX0 | 0 | 4 RX1 | 1 | 5 RX2 | 2 | 6 RX3 | 3 | 7 1 LSB = 0.1dB, signed number Applicable only for the enabled channels.
RESERVED	4	0x00000000
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.13.29 Sub block 0x101B – AWR_MONITOR_TX0_POWER_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX power values during an explicit monitoring chirp. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.78 – AWR_MONITOR_TX0_POWER_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x101B
SBLKLEN	2	Value = 24

STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** / **STATUS_FLAG for monitor** 0 / STATUS_ABS_ERR 1 / STATUS_FLATNESS_ERR 15:2 / RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TX_POWER_VALUE	6	The measured TX power for each enabled channel, at each enabled RF frequency is reported here. Byte numbers corresponding to different TX and RF, in this field are here: / RF1 / RF2 / RF3 TX0 / 1:0 / 3:2 / 5:4 (other bytes are reserved) 1 LSB = 0.1 dBm, signed number Only the entries of enabled RF Frequencies and enabled RX channels are valid.
RESERVED	2	0x0000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.13.30 Sub block 0x101C – AWR_MONITOR_TX1_POWER_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX power values during an explicit monitoring chirp. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.79 – AWR_MONITOR_TX2_POWER_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x101C
SBLKLEN	2	Value = 24
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** / **STATUS_FLAG for monitor**: 0 – STATUS_ABS_ERR 1 – STATUS_FLATNESS_ERR 15:2 – RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TX_POWER_VALUE	6	The measured TX power for each enabled channel, at each enabled RF frequency is reported here. Byte numbers corresponding to different TX and RF, in this field are here: | | RF1 | RF2 | RF3 | | TX1 | 1:0 | 3:2 | 5:4 | (other bytes are reserved) 1 LSB = 0.1 dBm, signed number Only the entries of enabled RF Frequencies and enabled RX channels are valid.
RESERVED	2	0x0000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.13.31 Sub block 0x101D – AWR_MONITOR_TX2_POWER_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX power values during an explicit monitoring chirp. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.80 – AWR_MONITOR_TX2_POWER_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x101D
SBLKLEN	2	Value = 24
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** | **STATUS_FLAG for monitor** 0 | STATUS_ABS_ERR 1 | STATUS_FLATNESS_ERR 15:2 | RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TX_POWER_VALUE	6	The measured TX power for each enabled channel, at each enabled RF frequency is reported here. Byte numbers corresponding to different TX and RF, in this field are here: | RF1 | RF2 | RF3 TX2 | 1:0 | 3:2 | 5:4 (other bytes are reserved) 1 LSB = 0.1 dBm, signed number Only the entries of enabled RF Frequencies and enabled RX channels are valid.

RESERVED	2	0x0000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.13.32 Sub block 0x101E – AWR_MONITOR_TX0_BALLBREAK_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX reflection coefficient's magnitude values, meant for detecting TX ball break. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.81 – AWR_MONITOR_TX0_BALLBREAK_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x101E
SBLKLEN	2	Value = 20
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** | **STATUS_FLAG for monitor** 0 | STATUS_TX0_BALLBREAK 15:1 | RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
TX_REFL_COEFF_ VALUE	2	The TX reflection coefficient's magnitude for this channel is reported here. 1 LSB = 0.1 dB, signed number
RESERVED	2	0x0000
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.13.33 Sub block 0x101F – AWR_MONITOR_TX1_BALLBREAK_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX reflection coefficient's magnitude values, meant for detecting TX ball break. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.82 – AWR_MONITOR_TX1_BALLBREAK_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x101F
SBLKLEN	2	Value = 20
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** | **STATUS_FLAG for monitor** 0 | STATUS_TX1_BALLBREAK 15:1 | RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
TX_REFL_COEFF_ VALUE	2	The TX reflection coefficient's magnitude for this channel is reported here. 1 LSB = 0.1 dB, signed number
RESERVED	2	0x0000
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14 Sub blocks related to AWR_RF_ASYNC_EVENT_MSG2

5.14.1 Sub block 0x1020 – AWR_MONITOR_TX2_BALLBREAK_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX reflection coefficient's magnitude values, meant for detecting TX ball break. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.83 – AWR_MONITOR_TX2_BALLBREAK_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1020
SBLKLEN	2	Value = 20
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** | **STATUS_FLAG for monitor** 0 | STATUS_TX2_BALLBREAK 15:1 | RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
TX_REFL_COEFF_ VALUE	2	The TX reflection coefficient's magnitude for this channel is reported here. 1 LSB = 0.1 dB, signed number
RESERVED	2	0x0000
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.2 Sub block 0x1021 – AWR_MONITOR_TX_GAIN_PHASE_MISMATCH_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX gain and phase mismatch values during an explicit monitoring chirp. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.84 – AWR_MONITOR_TX_GAIN_PHASE_MISMATCH_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x1021
SBLKLEN	2	Value = 60
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** | **STATUS_FLAG for monitor** 0 | STATUS_TX_GAIN_MISMATCH 1 | STATUS_TX_PHASE_MISMATCH 15:2 | RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000

TX_GAIN_VALUE	18	The measured TX PA loopback tone power at the RX ADC input, for each enabled TX channel, at each enabled RF frequency is reported here. Byte numbers corresponding to different TX and RF, in this field are here: | | RF1 | RF2 | RF3 | |---|---|---|---| | TX0 | 1:0 | 7:6 | 13:12 | | TX1 | 3:2 | 9:8 | 15:14 | | TX2 | 5:4 | 11:10 | 17:16 | 1 LSB = 0.1dBm, signed number Only the entries of enabled RF Frequencies and enabled TX channels are valid.
TX_PHASE_VALUE	18	The measured TX phase for each enabled channel, at each enabled RF frequency is reported here. Byte numbers corresponding to different TX and RF, in this field are here: | | RF1 | RF2 | RF3 | |---|---|---|---| | TX0 | 1:0 | 7:6 | 13:12 | | TX1 | 3:2 | 9:8 | 15:14 | | TX2 | 5:4 | 11:10 | 17:16 | 1 LSB = 360°/2^{16}. Only the entries of enabled RF Frequencies and enabled TX channels are valid. **Note:** these phases include an unknown bias common to all TX channels.
RESERVED	4	0x00000000
RESERVED	4	0x00000000
TIMESTAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.3 Sub block 0x1022 – AWR_MONITOR_TX0_BPM_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX0 BPM error values. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.85 – AWR_MONITOR_TX0_BPM_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x1022
SBLKLEN	2	Value = 20
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. Bit | STATUS_FLAG for monitor 0 | STATUS_TX0_BPM_PHASE 1 | STATUS_TX0_BPM_AMPLITUDE 15:2 | RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TX_BPM_PHASE_ DIFF_VALUE	2	The TX output phase difference between the two BPM settings (phase for TX BPM setting 0 – phase for TX BPM setting 1) is reported here. 1 LSB = $360^{\circ} / 2^{16}$ Valid range: TBD Recommended value = TBD
TX_BPM_ AMPLITUDE_DIFF_ VALUE	1	The deviation of the TX output amplitude difference between the two BPM settings (amplitude for TX BPM setting 0 – amplitude for TX BPM setting 1) from the ideal 0dB is reported here. 1 LSB = 0.1dB, signed number Valid range: TBD Recommended value = TBD

RESERVED	1	0x00
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.4 Sub block 0x1023 – AWR_MONITOR_TX1_BPM_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX1 BPM error values. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.86 – AWR_MONITOR_TX1_BPM_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1023
SBLKLEN	2	Value = 20
STATUS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** | **STATUS_FLAG for monitor** 0 | STATUS_TX1_BPM_PHASE 1 | STATUS_TX1_BPM_AMPLITUDE 15:2 | RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TX_BPM_PHASE_DIFF_VALUE	2	The TX output phase difference between the two BPM settings (phase for TX BPM setting 0 – phase for TX BPM setting 1) is reported here. 1 LSB = $360^{\circ} / 2^{16}$. Valid range: TBD Recommended value = TBD

TX_BPM_ AMPLITUDE_DIFF_ VALUE	1	The deviation of the TX output amplitude difference between the two BPM settings from the ideal 0dB is reported here. 1 LSB = 0.1dB, signed number Valid range: TBD Recommended value = TBD
RESERVED	1	0x00
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.5 *Sub block 0x1024 – AWR_MONITOR_TX2_BPM_REPORT_AE_SB*

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured TX2 BPM error values. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.87 – AWR_MONITOR_TX2_BPM_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1024
SBLKLEN	2	Value = 20
STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** / **STATUS_FLAG for monitor** 0 / STATUS_TX2_BPM_PHASE 1 / STATUS_TX2_BPM_AMPLITUDE 15:2 / RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000

TX_BPM_PHASE_ DIFF_VALUE	2	The TX output phase difference between the two BPM settings (phase for TX BPM setting 0 – phase for TX BPM setting 1) is reported here. 1 LSB = $360^{\circ} / 2^{16}$ Valid range: TBD Recommended value = TBD
TX_BPM_ AMPLITUDE_DIFF_ VALUE	1	The deviation of the TX output amplitude difference between the two BPM settings from the ideal 0dB is reported here. 1 LSB = 0.1dB, signed number Valid range: TBD Recommended value = TBD
RESERVED	1	0x00
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.6 Sub block 0x1025 – AWR_MONITOR_SYNTHESIZER_FREQUENCY_ REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing information related to measured frequency error during the chirp. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.88 – AWR_MONITOR_SYNTH_FREQUENCY_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1025
SBLKLEN	2	Value = 32

STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** | **STATUS_FLAG for monitor** 0 | STATUS_SYNTH_FREQ_ERR 15:1 | RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
MAX_FREQUENCY_ ERROR_ VALUE	4	This field indicates the maximum instantaneous frequency error measured during the chirps for which frequency monitoring has been enabled in the previous monitoring period. **Bits** | **Parameter** 31:0 | Maximum frequency error value, signed number. 1 LSB = 1kHz.
FREQUENCY_ FAILURE_ COUNT	4	This field indicates the number of times during chirping in the previous monitoring period in which the measured frequency error violated the allowed threshold. Frequency error threshold violation is counted every 10ns. **Bits** | **Parameter** 31:19 | RESERVED 18:0 | Failure count, unsigned number.
RESERVED	4	0x00000000
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.7 Sub block 0x1026 – AWR_MONITOR_EXTERNAL_ANALOG_SIGNALS_ REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the external signal voltage values measured using the GPADC. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.89 – AWR_MONITOR_EXTERNAL_ANALOG_SIGNALS_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1026
SBLKLEN	2	Value = 28
STATUS_FLAGS	2	Status flags indicating pass fail results corresponding to various threshold checks under this monitor. Bit Number / Definition: 0 – STATUS_ANALOGTEST1 1 – STATUS_ANALOGTEST2 2 – STATUS_ANALOGTEST3 3 – STATUS_ANALOGTEST4 4 – STATUS_ANAMUX 5 – STATUS_VSENSE 15:3 – RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error

EXTERNAL_ ANALOG_SIGNAL_ VALUES	12	MEASURED_VALUE **Bytes** / **SIGNAL** 1:0 – ANALOGTEST1 3:2 – ANALOGTEST2 5:4 – ANALOGTEST3 7:6 – ANALOGTEST4 9:8 – ANAMUX 11:10 – VSENSE 1 LSB = 1.8V/1024
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.8 Sub block 0x1027 – AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_ REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing information about Internal TX0 internal analog signals. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.90 – AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x1027
SBLKLEN	2	Value = 16

STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit Location** / **STATUS_FLAG for monitor** 0 – STATUS_SUPPLY_TX0 1 – STATUS_DCBIAS_TX0 15:2 – RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.9 *Sub block 0x1028 – AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_ REPORT_AE_SB*

This API is a Monitoring Report API which the AWR device sends to the host, containing information about Internal TX1 internal analog signals. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.91 – AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x1028
SBLKLEN	2	Value = 16

STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** / **STATUS_FLAG for monitor** 0 / STATUS_SUPPLY_TX1 1 / STATUS_DCBIAS_TX1 15:2 / RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.10 Sub block 0x1029 – AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing information about Internal TX2 internal analog signals. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.92 – AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x1029
SBLKLEN	2	Value = 16

STATUS_FLAGS	2	Status flag indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** / **STATUS_FLAG for monitor** 0 – STATUS_SUPPLY_TX2 1 – STATUS_DCBIAS_TX2 15:2 – RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.11 Sub block 0x102A – AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing information about Internal RX internal analog signals. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.93 – AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x102A
SBLKLEN	2	Value = 16

STATUS_FLAGS	2	Status flags indicating pass fail results corresponding to various threshold checks under this monitor.

Bit	STATUS_FLAG for monitor
0	STATUS_SUPPLY_RX0
1	STATUS_SUPPLY_RX1
2	STATUS_SUPPLY_RX2
3	STATUS_SUPPLY_RX3
4	STATUS_DCBIAS_RX0
5	STATUS_DCBIAS_RX1
6	STATUS_DCBIAS_RX2
7	STATUS_DCBIAS_RX3
8	STATUS_PWRDET_RX0
9	STATUS_PWRDET_RX1
10	STATUS_PWRDET_RX2
11	STATUS_PWRDET_RX3
15:12	RESERVED

0 – FAIL or check wasn't done
1 – PASS

ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.12 Sub block 0x102B – AWR_MONITOR_PMCLKLO_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing information about Internal PM, CLK and LO subsystems' internal analog signals. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.94 – AWR_MONITOR_PM_CLK_LO_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x102B
SBLKLEN	2	Value = 16
STATUS_FLAGS	2	Status flags indicating pass fail results corresponding to various threshold checks under this monitor. Bit Location: STATUS_FLAG for monitor 0: STATUS_SUPPLY_PMCLKLO 1: STATUS_DCBIAS_PMCLKLO 2: STATUS_LVDS_PMCLKLO (Use this status bit only if LVDS is used, else ignore this) 15:3: RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.13 Sub block 0x102C – AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing information about the measured value of the GPADC input DC signals whose measurements were enabled. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.95 – AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x102C
SBLKLEN	2	Value = 20
STATUS_FLAGS	2	Status flags indicating pass fail results corresponding to various threshold checks under this monitor. **Bit Location** – **STATUS_FLAG for monitor** 0 – STATUS_GPADC_REF1 1 – STATUS_GPADC_REF2 15:2 – RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error
GPADC_REF1_ VALUE	2	The measured GPADC outputs corresponding to internal DC signal (GPADC_REF1, expected level 0.45V) is reported here. 1 LSB = 1.8V/1024
GPADC_REF2_ VALUE	2	The measured GPADC outputs corresponding to internal DC signal (GPADC_REF2, expected level 1.0V) is reported here. 1 LSB = 1.8V/1024
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.14 Sub block 0x102D – AWR_MONITOR_PLL_CONTROL_VOLTAGE_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured PLL control voltage values during explicit monitoring chirps. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.96 – AWR_MONITOR_PLL_CONTROL_VOLTAGE_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x102D
SBLKLEN	2	Value = 32
STATUS_FLAGS	2	Status flags indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** / **STATUS_FLAG for monitor** 0 – STATUS_APLL_VCTRL 1 – STATUS_SYNTH_VCO1_VCTRL_MAX_ FREQ 2 – STATUS_SYNTH_VCO1_VCTRL_MIN_ FREQ 3 – STATUS_SYNTH_VCO1_SLOPE 4 – STATUS_SYNTH_VCO2_VCTRL_MAX_ FREQ 5 – STATUS_SYNTH_VCO2_VCTRL_MIN_ FREQ 6 – STATUS_SYNTH_VCO2_SLOPE 15:7 – RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error

Nested table for STATUS_FLAGS:

Bit	STATUS_FLAG for monitor
0	STATUS_APLL_VCTRL
1	STATUS_SYNTH_VCO1_VCTRL_MAX_ FREQ
2	STATUS_SYNTH_VCO1_VCTRL_MIN_ FREQ
3	STATUS_SYNTH_VCO1_SLOPE
4	STATUS_SYNTH_VCO2_VCTRL_MAX_ FREQ
5	STATUS_SYNTH_VCO2_VCTRL_MIN_ FREQ
6	STATUS_SYNTH_VCO2_SLOPE
15:7	RESERVED

PLL_CONTROL_ VOLTAGE_VALUES	16	The measured values of PLL control voltage levels and Synthesizer VCO slopes are reported here. Byte numbers corresponding to different control voltage values reported in this field are here:

Bytes	SIGNAL	1 LSB
1:0	APLL_VCTRL	1mV
3:2	SYNTH_VCO1_VCTRL_MAX_ FREQ	1mV
5:4	SYNTH_VCO1_VCTRL_MIN_ FREQ	1mV
7:6	SYNTH_VCO1_SLOPE	1MHz/V
9:8	SYNTH_VCO2_VCTRL_MAX_ FREQ	1mV
11:10	SYNTH_VCO2_VCTRL_MIN_ FREQ	1mV
13:12	SYNTH_VCO2_SLOPE	1MHz/V
Others	RESERVED	RESERVED

		Only the fields corresponding to the enabled monitors are valid. The failure thresholds are based on the following: Valid VCTRL values are [150 to 1250] mV. Valid VCO1_SLOPE values are [1760 to 2640] MHz/V. Valid VCO2_SLOPE values are [3520 to 5280] MHz/V.
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

5.14.15 Sub block 0x102E – AWR_MONITOR_DUAL_CLOCK_COMP_REPORT_AE_SB

This API is a monitoring report API which the AWR device sends to the host, containing information about the relative frequency measurements. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.97 – AWR_MONITOR_DUAL_CLOCK_COMP_REPORT_AE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x102E
SBLKLEN	2	Value = 32
STATUS_FLAGS	2	Status flags indicating pass fail results corresponding to various threshold checks under this monitor. **Bit** / **STATUS_FLAG for monitor** 0 / STATUS_CLK_PAIR0 1 / STATUS_CLK_PAIR1 2 / STATUS_CLK_PAIR2 3 / STATUS_CLK_PAIR3 4 / STATUS_CLK_PAIR4 5 / STATUS_CLK_PAIR5 15:6 / RESERVED 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Indicates any error reported during monitoring Value of 0 indicates no error

Bit	STATUS_FLAG for monitor
0	STATUS_CLK_PAIR0
1	STATUS_CLK_PAIR1
2	STATUS_CLK_PAIR2
3	STATUS_CLK_PAIR3
4	STATUS_CLK_PAIR4
5	STATUS_CLK_PAIR5
15:6	RESERVED

FREQ_MEAS_ VALUES	16	The measured clock frequencies from the enabled clock pair measurements are reported here. Byte numbers corresponding to different frequency measurement values reported in this field are here: 1 LSB = 0.1 MHz, unsigned number
RESERVED	4	0x00000000
TIME_STAMP	4	This field indicates when the last monitoring in the enabled set was performed. 1 LSB = 1 millisecond (the stamp rolls over upon exceeding allotted bit width)

Bytes	**CLOCK PAIR**	**MEASURED CLOCK FREQUENCY**
1:0	0	BSS_600M
3:2	1	BSS_200M
5:4	2	BSS_100M
7:6	3	GPADC_10M
9:8	4	RCOSC_10M
11:10	5	RAMPGEN_100M
15:12	RESERVED	RESERVED

5.14.16 Sub block 0x1031 – AWR_MONITOR_RX_MIXER_IN_POWER_REPORT_AE_SB

This API is a Monitoring Report API which the AWR device sends to the host, containing the measured RX mixer input voltage swing values. The AWR device sends this to host at the programmed periodicity or when failure occurs, as programmed by the configuration API SB.

Table 5.98 – AWR_MONITOR_RX_MIXER_IN_POWER_REPORT_AE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1031
SBLKLEN	2	Value = 24

STATUS_FLAGS	2	<table><tr><th>Bit</th><th>STATUS_FLAG for monitor</th></tr><tr><td>0</td><td>STATUS_MIXER_IN_POWER_RX0</td></tr><tr><td>1</td><td>STATUS_ MIXER_IN_POWER_RX1</td></tr><tr><td>2</td><td>STATUS_ MIXER_IN_POWER_RX2</td></tr><tr><td>3</td><td>STATUS_ MIXER_IN_POWER_RX3</td></tr><tr><td>15:4</td><td>RESERVED</td></tr></table> 0 – FAIL or check wasn't done 1 – PASS
ERROR_CODE	2	Internal sanity check violations are reported here. Value = 0: No error Other values: Error (see error code definition matrix)
PROFILE_INDX	1	Profile Index for which this monitoring report applies
RESERVED	3	0x000000
RX_MIXER_IN_ VOLTAGE_VALUE	4	The measured RX mixer input voltage swing values are reported here. The byte location of the value for each receivers is tabulated here: <table><tr><th>Receiver</th><th>Byte Location</th></tr><tr><td>RX0</td><td>0</td></tr><tr><td>RX1</td><td>1</td></tr><tr><td>RX2</td><td>2</td></tr><tr><td>RX3</td><td>3</td></tr></table> 1 LSB = 1800 mV/256, unsigned number Only the entries of enabled RX channels are valid.
RESERVED	4	0x00000000
TIME_STAMP	4	When this monitoring began is indicated here. 1LSB = TIME_STAMP_TBD.

5.15 Sub blocks related to AWR_DEV_RFPOWERUP_MSG

5.15.1 Sub block 0x4000 – AWR_DEV_RFPOWERUP_SB

This sub block is a command to power up the BSS

Table 5.99 describes the content of this sub block.

Table 5.99 – AWR_DEV_RFPOWERUP_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4000
SBLKLEN	2	Value = 4

5.16 Sub blocks related to AWR_DEV_CONF_SET_MSG

5.16.1 Sub block 0x4040 – AWR_DEV_MCUCLOCK_CONF_SET_SB

This sub block contains the configurations to setup the desired frequency of the MCU Clock that is output from the device.

Table 5.47 describes the contents of this sub block.

Table 5.47 – AWR_DEV_MCUCLOCK_CONF_SET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4040
SBLKLEN	2	Value = 8
MCUCLOCK_CTRL	1	This field controls the enable-disable of the MCU clock. Value Description 0x0 Disable MCU clock 0x1 Enable MCU clock
MCUCLOCK_SRC	1	This field specifies the source of the MCU clock. Applicable only in case of MCU clock enable. Else ignored. Value Description 0x0 XTAL (as connected to the device) 0x2 600MHz PLL divided clock
SRCCLOCK_DIV	1	This field specifies the division factor to be applied to source clock. Applicable only in case of MCU clock enable. Else ignored. Value Description 0x0 Divide by 1 0x1 Divide by 2 … …

		0xFF Divide by 256
RESERVED	1	0x00

5.16.2 Sub block 0x4041 – AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB

This sub block contains the configuration of the data format of the samples received over the receive chain to be transferred out to an external host over the configured data path (LVDS or CSI2).

Table 5.100 describes the content of this sub block.

Table 5.100 – AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB contents

Field Name	Number of bytes	Description		
SBLKID	2	Value = 0x4041		
SBLKLEN	2	Value = 16		
RX_CHAN_EN	2	b0	RX_CHAN0_EN	
			0	Disable RX Channel 0
			1	Enable RX Channel 0
		b1	RX_CHAN1_EN	
			0	Disable RX Channel 1
			1	Enable RX Channel 1
		b2	RX_CHAN2_EN	
			0	Disable RX Channel 2
			1	Enable RX Channel 2
		b3	RX_CHAN3_EN	
			0	Disable RX Channel 3
			1	Enable RX Channel 3
		b15:4	Reserved	
NUM_ADC_BITS	2	b1:0	00	12 bits
			01	14 bits
			10	16 bits
			Other	Reserved
		b15:2	Reserved	
ADC_OUT_FMT	2	b1:0	00	Real
			01	Complex

		Other Reserved b15:2 Reserved
IQ_SWAP_SEL	1	b1:0 To swap the IQ samples (if complex format) 00 Sample interleave mode – I first 01 Sample interleave mode – Q first Other Reserved b7:2 Reserved
CHAN_INTERLEAVE	1	b1:0 Channel interleaving of the samples stored in the ADC buffer to be transferred out on the data path. 00 Interleaved mode of storage 01 Non-interleaved mode of storage Other Reserved b7:2 Reserved
RESERVED	4	0x00000000

5.16.3 Sub block 0x4042 – AWR_DEV_RX_DATA_PATH_CONF_SET_SB

This sub block contains the configurations of the data path to transfer the captured ADC samples received over the receive chain to be transferred out to an external host.

Table 5.101 describes the content of this sub block.

Table 5.101 – AWR_DEV_RX_DATA_PATH_CONF_SET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4042
SBLKLEN	2	Value = 12
DATA_INTF_SEL	1	This field specifies the data path selected to transfer the Radar info. Value Description 0x0 CSI2 interface select 0x1 LVDS interface select 0x2 SPI interface selected (applicable when a large inter-frame time is provided to transfer the data over the SPI)

DATA_TRANS_FMT_PKT0	1	b5:0 Packet 0 content selection Value Description 000001 ADC 000110 CP_ADC (See note at the bottom of this table) 001001 ADC_CP 110110 CP_ADC_CQ (See note at the bottom of this table) b7:6 Packet 0 virtual channel number **(valid only for CSI2)** Value Description 00 Virtual channel number 0 (Default) 01 Virtual channel number 1 10 Virtual channel number 2 11 Virtual channel number 3
DATA_TRANS_FMT_PKT1	1	b5:0 Packet 1 content selection Value Description 000000 Suppress packet 1 transmission 001110 CP_CQ (See note at the bottom of this table) 001011 CQ_CP (See note at the bottom of this table) b7:6 Packet 1 virtual channel number **(valid only for CSI2)** Value Description 00 Virtual channel number 0 (Default) 01 Virtual channel number 1 10 Virtual channel number 2 11 Virtual channel number 3
CQ_CONFIG	1	This specifies the data size of CQ samples on the lanes Bits 1:0 Description 00 12 bit 01 14 bit 10 16 bit 11 RESERVED Bits 7:2: RESERVED **Note:** The CQ size can be configured only if CQ and ADC data is sent in separate packets. When ADC and CQ is sent in the same

		packet, then CQ size will be same as ADC data size.
CQ0_TRANS_SIZE	1	Number of samples (in 16 bit halfwords) of CQ0 data to be transferred. Valid range [32 halfwords to 128 halfwords] **Note:** Ensure that the number of halfwords specified are a multiple of the number of lanes selected.
CQ1_TRANS_SIZE	1	Number of samples (in 16 bit halfwords) of CQ1 data to be transferred. Valid range [32 halfwords to 128 halfwords] **Note:** Ensure that the number of halfwords specified are a multiple of the number of lanes selected.
CQ2_TRANS_SIZE	1	Number of samples (in 16 bit halfwords) of CQ2 data to be transferred. Valid range [32 halfwords to 128 halfwords] **Note:** Ensure that the number of halfwords specified are a multiple of the number of lanes selected.
RESERVED	1	0x00

NOTE 1: CP is Chirp Parameter information which is defined for each RX as follows

Bit	Description
11:0	Chirp number In legacy frame configuration, chirp number for starts from 1 and increments for each chirp within the frame and resets to 0 for the next frame. In advanced frame configuration chirp number starts from 1 and increments for each chirp within the burst and resets to 0 for the next burst.
15:12	Reserved
17:16	Channel number The receive channel number which is encoded as 00 – RX0 01 – RX1 10 – RX2 11 – RX3
21:18	Profile number The profile number to which the chirp belongs
31:22	Reserved

NOTE 2: CQ is Chirp Quality information which is defined in Section 8

5.16.4 Sub block 0x4043 – AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB

This sub block contains the configurations to enables the lanes of the LVDS path to transfer Radar information to an external host.

Table 5.102 describes the content of this sub block.

Table 5.102 – AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4043
SBLKLEN	2	Value = 8
LANE_EN	2	b0 LANE0_EN 0 Disable lane 0 1 Enable lane 0 b1 LANE1_EN 0 Disable lane 1 1 Enable lane 1 b2 LANE2_EN 0 Disable lane 2 1 Enable lane 2 b3 LANE3_EN 0 Disable lane 3 1 Enable lane 3 b15:4 Reserved
RESERVED	2	0x0000

5.16.5 Sub block 0x4044 – AWR_DEV_RX_DATA_PATH_CLK_SET_SB

This sub block contains the clock configurations of the LVDS lanes.

Table 5.103 describes the content of this sub block.

Table 5.103 – AWR_DEV_RX_DATA_PATH_CLK_SET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4044
SBLKLEN	2	Value = 8

LANE_CLK_CFG (Selection valid only for LVDS. For CSI2, DDR is used always)	1	b0 BIT_CLK_SEL 0 SDR clock 1 DDR clock (Only valid value for CSI2) b7:1 Reserved
DATA_RATE	1	Data rate selection 0000b 900 Mbps (DDR only) 0001b 600 Mbps (DDR only) 0010b 450 Mbps (SDR, DDR) 0011b 400 Mbps (DDR only) 0100b 300 Mbps (SDR, DDR) 0101b 225 Mbps (DDR only) 0110b 150 Mbps (SDR, DDR) Others Reserved
RESERVED	2	0x0000

5.16.6 *Sub block 0x4045 – AWR_DEV_LVDS_CFG_SET_SB*

This sub block contains the configurations of the LVDS lanes.

Table 5.104 describes the content of this sub block.

Table 5.104 – AWR_DEV_LVDS_CFG_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x4045
SBLKLEN	2	Value = 8
LANE_FMT_MAP	2	LANE0 Format Map. The mapping of the data on the lanes is depicted in the figure below 0x0000 Format map 0 0x0001 Format map 1
LANE_PARAM_CFG	2	b0 MSB_FIRST 0 Disable (LSB First) 1 Enable (MSB First) b1 Packet End Pulse Enable 0 Disable 1 Enable

		b2 CRC Enable 0 Disable 1 Enable b15:3 Reserved

The mapping of the 8 sample (8*16 = 128 bit) information onto the serial interface lanes is determined by the LANE_FMT_MAP parameter. The choice of format map translating to the transfer of data on the lanes is depicted in the image below (the x axis represents time – hence the samples are as available on the lanes in time and the receiver will receive the samples in the reverse order as depicted below).

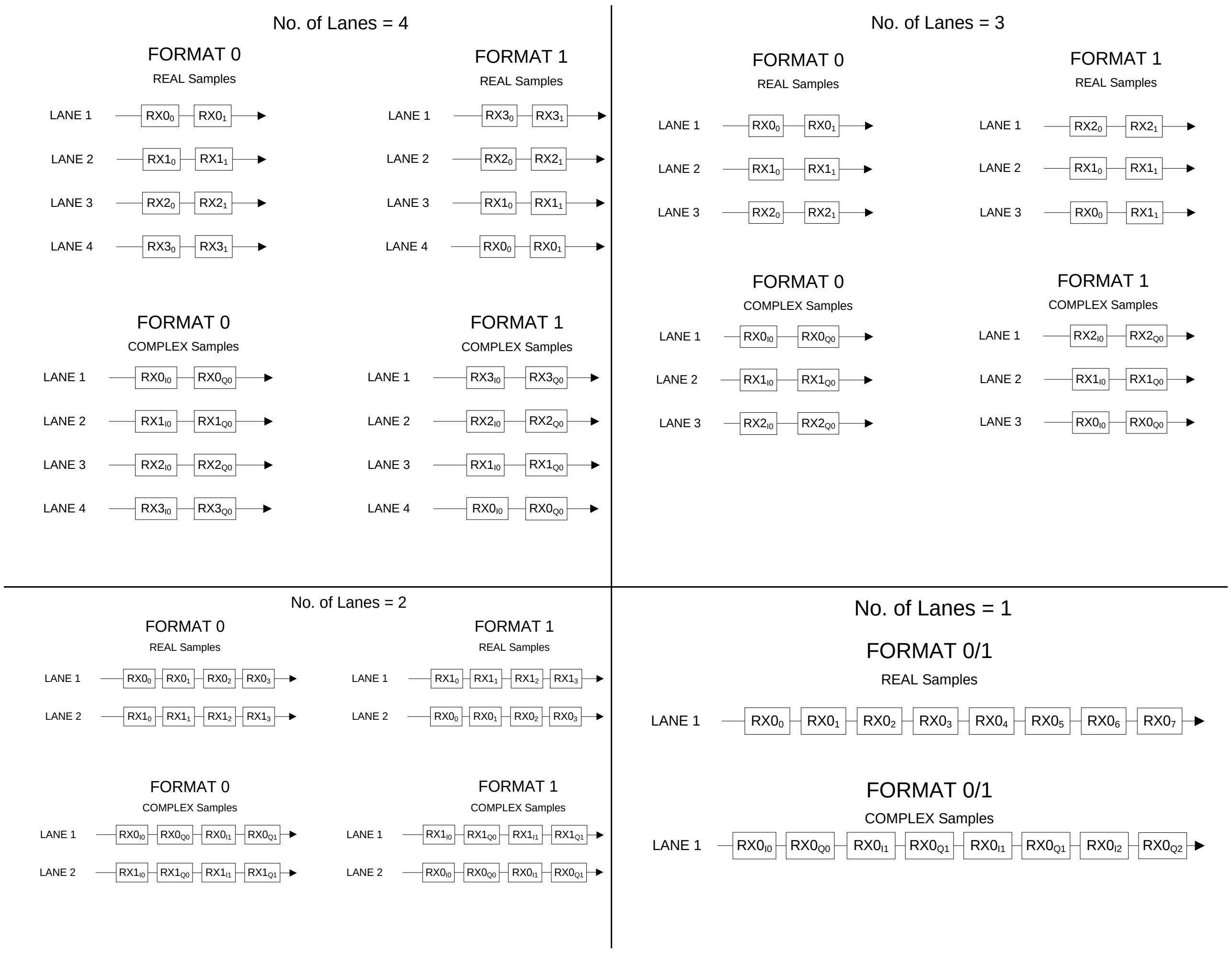


Figure 5.2 – Lane formats and the order of receiving the data from the lanes

5.16.7 Sub block 0x4046 – AWR_DEV_RX_CONTSTREAMING_MODE_CONF_SET_SB

This sub block contains the configurations of the data path to transfer the captured ADC samples continuously without missing any sample to an external host.

Table 5.105 describes the content of this sub block.

Table 5.105 – AWR_DEV_RX_CONTSTREAMING_MODE_CONF_SET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4046
SBLKLEN	2	Value = 8
CONT_STREAMING _MODE	2	Continuous streaming mode enable Value Description 0x0 Continuous streaming mode data transfer disable 0x1 Continuous streaming mode data transfer enable
RESERVED	2	0x0000

5.16.8 Sub block 0x4047 – AWR_DEV_CSI2_CFG_SET_SB

This sub block contains the various configurations of the parameters of the CSI2 module.

Table 5.106 describes the content of this sub block.

Table 5.106 – AWR_DEV_CSI2_CFG_SET_SB contents

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x4047	
SBLKLEN	2	Value = 12	
LANE_POS_POL_ SEL	4	**Bits**	**Definition**
		b2:0	DATA_LANE0_POS Valid values (Should be a unique position if lane 0 is enabled, ignored if lane 0 is not enabled): 000b – Unused, 001b – Position 1 (default), 010b – Position 2, 011b – Position 3, 100b – Position 4, 101b – Position 5
		b3	DATA_LANE0_POL 0b – PLUSMINUS pin order, 1b – MINUSPLUS pin order

		b6:4 DATA_LANE1_POS Valid values (Should be a unique position if lane 1 is enabled, ignored if lane 1 is not enabled): 000b – Unused, 001b – Position 1, 010b – Position 2 (default), 011b – Position 3, 100b – Position 4, 101b – Position 5 b7 DATA_LANE1_POL 0b – PLUSMINUS pin order, 1b – MINUSPLUS pin order b10:8 DATA_LANE2_POS Valid values (Should be a unique position if lane 2 is enabled, ignored if lane 2 is not enabled): 000b – Unused, 001b – Position 1, 010b – Position 2, 011b – Position 3, 100b – Position 4 (default), 101b – Position 5 b11 DATA_LANE2_POL 0b – PLUSMINUS pin order, 1b – MINUSPLUS pin order b14:12 DATA_LANE3_POS Valid values (Should be a unique position if lane 3 is enabled, ignored if lane 3 is not enabled): 000b – Unused, 001b – Position 1, 010b – Position 2, 011b – Position 3, 100b – Position 4, 101b – Position 5 (default) b15 DATA_LANE3_POL 0b – PLUSMINUS pin order, 1b – MINUSPLUS pin order b18:16 CLOCK_POS Valid values (Should be a unique position): 0000b – Unused, 001b – Unused, 010b – Position 2, 011b – Position 3 (default), 100b – Position 4 b19 CLOCK_POL 0b – PLUSMINUS pin order, 1b – MINUSPLUS pin order b31:20 RESERVED
RESERVED	4	0x00000000

5.16.9 *Sub block 0x4048 – AWR_DEV_PMICCLOCK_CONF_SET_SB*

This sub block contains the configurations to setup the desired frequency of the PMIC Clock that is output from the device. The configurations also allow setting up the dither values for the clock.

Table 5.107 describes the contents of this sub block.

Table 5.107 – AWR_DEV_PMICCLOCK_CONF_SET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x4048
SBLKLEN	2	Value = 16
PMICCLOCK_CTRL	1	This field controls the enable-disable of the PMIC clock. Value Description 0x0 Disable PMIC clock 0x1 Enable PMIC clock
PMICCLOCK_SRC	1	This field specifies the source of the PMIC clock. Applicable only in case of PMIC clock enable. Else ignored. Value Description 0x0 XTAL (as connected to the device) 0x2 600MHz PLL divided clock
SRCCLOCK_DIV	1	This field specifies the division factor to be applied to source clock. Applicable only in case of PMIC clock enable. Else ignored. Value Description 0x0 Divide by 1 0x1 Divide by 2 … … 0xFF Divide by 256
MODE_SELECT	1	This field specifies the mode of operation for the PMIC clock generation. Applicable only in case of PMIC clock enable. Else ignored. Value Description 0x0 Continuous mode (free running mode where the frequency change/jump is triggered based on configured number of internal clock ticks) 0x1 Chirp-to-Chirp staircase mode (frequency change/jump is triggered at every chirp boundary)
FREQ_SLOPE	4	Applicable only in case of PMIC clock enable. Else ignored. Bit[25:0] – Frequency slope value to be applied in [8.18] format. 1 LSB = $1/2^{18}$ In continuous mode this value is accumulated every 100 MHz internal clock tick with the seed as MIN_NDIV_VAL till MAX_NDIV_VAL is reached

		In the stair case mode this value is accumulated every chirp with the seed as MIN_NDIV_VAL till MAX_NDIV_VAL is reached
MIN_NDIV_VAL	1	Applicable only in case of PMIC clock enable. Else ignored. Min allowed divider value (depends upon the highest desired clock frequency)
MAX_NDIV_VAL	1	Applicable only in case of PMIC clock enable. Else ignored. Max allowed divider value (depends upon the lowest desired clock frequency)
CLK_DITHER_EN	1	Applicable only in case of PMIC clock enable. Else ignored. This field controls the enable-disable of the clock dithering. Adds a pseudo random real number (0 or 1) to the accumulated divide value. Hence it brings a random dithering of 1 LSB. Value Description 0x0 Clock dithering disabled 0x1 Clock dithering enabled
RESERVED	1	0x00

5.16.10 Sub block 0x4049 – AWR_MSS_PERIODICTESTS_CONF_SB

This sub block is used to trigger the periodic tests in MSS.

Table 5.108 describes the content of this sub block.

Table 5.108 – AWR_MSS_PERIODICTESTS_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x4049
SBLKLEN	2	Value = 16
PERIODICITY	4	Periodicity at which tests need to be run 1 LSB = 1 ms Minimum value is 40ms
TEST_ EN	4	1 – Enable, 0 – Disable **Bit** **Monitoring type** 0 PERIODIC_CONFG_REGISTER_READ_EN 1 ESM_MONITORING_EN 31:2 RESERVED
REPORTING_MODE	1	Controls when the AWR device sends the report corresponding to

		the periodic tests to the host. A report generically refers to both success/failure status flags. **Value** / **Definition** 0 — Report is sent only on a failure 1 — Report is sent every monitoring period
RESERVED	3	0x000000

5.16.11 *Sub block 0x404A – AWR_MSS_LATENTFAULT_TEST_CONF_SB*

This sub block is used to trigger the periodic tests in MSS.

Table 5.109 describes the content of this sub block.

Table 5.109 – AWR_MSS_LATENTFAULT_TEST_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x404A
SBLKLEN	2	Value = 16

		Bits	**Definition**
		0	MibSPI self-test
		1	DMA self-test
		2	RESERVED
		3	RTI self-test
		4	ESM self-test
		5	EDMA self-test
		6	CRC self-test
		7	VIM self-test
		8	RESERVED
		9	Mailbox self-test
		10	LVDS pattern generation test
		11	CSI2 pattern generation test
		12	Generating NERROR
		13	MibSPI single bit error test
		14	MibSPI double bit error test
TEST_EN_1	4	15	DMA Parity error
		16	TCMA RAM single bit errors
		17	TCMB RAM single bit errors
		18	TCMA RAM double bit errors
		19	TCMB RAM double bit errors
		20	TCMA RAM parity errors.
		21	TCMB RAM parity errors.
		22	RESERVED
		23	RESERVED
		24	DMA MPU Region tests
		25	MSS Mailbox single bit errors
		26	MSS Mailbox double bit errors
		27	BSS Mailbox single bit errors
		28	BSS Mailbox double bit errors
		29	EDMA MPU test
		30	EDMA parity test
		31	CSI2 parity test

TEST_EN_2	4	<table><tr><th>Bits</th><th>Definition</th></tr><tr><td>0</td><td>DCC self-test</td></tr><tr><td>1</td><td>DCC fault insertion test</td></tr><tr><td>2</td><td>PCR fault generation test</td></tr><tr><td>3</td><td>VIM RAM parity test</td></tr><tr><td>4</td><td>SCI boot time test</td></tr><tr><td>31:5</td><td>RESERVED</td></tr></table>
REPORTING_ MODE	1	<table><tr><th>Value</th><th>Definition</th></tr><tr><td>0</td><td>Report is send only upon a failure</td></tr><tr><td>1</td><td>Report is sent every monitoring period</td></tr></table>
TEST_MODE	1	<table><tr><th>Value</th><th>Definition</th></tr><tr><td>0</td><td>Production mode. Latent faults are tested and any failures are reported</td></tr><tr><td>1</td><td>Characterization mode. Faults are injected and failures are reported which allows testing of the failure reporting</td></tr></table>
RESERVED	2	0x0000

5.16.12 *Sub block 0x404B – AWR_DEV_TESTPATTERN_GEN_SET_SB*

This sub block contains the configurations to setup the test pattern to be generated and transferred over the selected high speed interface (LVDS/CSI2). This command has to be issued after the data path configurations commands are issued. This can be used to perform a sanity test of the high speed interface connectivity and correct reception.

Table 5.110 describes the contents of this sub block.

Table 5.110 – AWR_DEV_TESTPATTERN_GEN_SET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x404B
SBLKLEN	2	Value = 48
TESTPATTERN_	1	This field controls the enable-disable of the generation of the test

GEN_CTRL		pattern. Value Description 0x0 Disable test pattern generation 0x1 Enable test pattern generation
TESTPATTERN_ GEN_TIMING	1	Number of system clocks (200 MHz) between successive samples for the test pattern gen. Applicable only in case of Test pattern enable. Else ignored.
TESTPATTERN_ PKT_SIZE	2	Number of ADC samples to capture for each RX Valid range: 64 to MAX_NUM_SAMPLES, Where MAX_NUM_SAMPLES is such that all the enabled RX channels' data fits into 16 kB memory, with each sample consuming 2 bytes for real ADC output case and 4 bytes for complex 1x and complex 2x ADC output cases. For example: 4 RX, Complex ADC output → MAX_NUM_SAMPLES = 1024 4 RX, Real ADC output → MAX_NUM_SAMPLES = 2048 2 RX, Complex ADC output → MAX_NUM_SAMPLES = 2048 2 RX, Real ADC output → MAX_NUM_SAMPLES = 4096
NUM_TESTPATTERN _PKTS	4	Number of test pattern packets to send For infinite packets set it to 0
TESTPATTERN_ RX0_ICFG	4	This field specifies the values for Rx0, I channel. Applicable only in case of test pattern enable. Else ignored. Bits Description 15:0 Start offset value to be used for the first sample for the test pattern data 31:16 Value to be added for each successive sample for the test pattern data
TESTPATTERN_ RX0_QCFG	4	This field specifies the values for Rx0, Q channel. Applicable only in case of test pattern enable. Else ignored. Bits Description 15:0 Start offset value to be used for the first sample for the test pattern data 31:16 Value to be added for each successive sample for the test pattern data
TESTPATTERN_ RX1_ICFG	4	This field specifies the values for Rx1, I channel. Applicable only in case of test pattern enable. Else ignored. Bits Description 15:0 Start offset value to be used for the first sample for the test pattern data

		31:16 Value to be added for each successive sample for the test pattern data
TESTPATTERN_ RX1_QCFG	4	This field specifies the values for Rx1, Q channel. Applicable only in case of test pattern enable. Else ignored. Bits Description 15:0 Start offset value to be used for the first sample for the test pattern data 31:16 Value to be added for each successive sample for the test pattern data
TESTPATTERN_ RX2_ICFG	4	This field specifies the values for Rx2, I channel. Applicable only in case of test pattern enable. Else ignored. Bits Description 15:0 Start offset value to be used for the first sample for the test pattern data 31:16 Value to be added for each successive sample for the test pattern data
TESTPATTERN_ RX2_QCFG	4	This field specifies the values for Rx2, Q channel. Applicable only in case of test pattern enable. Else ignored. Bits Description 15:0 Start offset value to be used for the first sample for the test pattern data 31:16 Value to be added for each successive sample for the test pattern data
TESTPATTERN_ RX3_ICFG	4	This field specifies the values for Rx3, I channel. Applicable only in case of test pattern enable. Else ignored. Bits Description 15:0 Start offset value to be used for the first sample for the test pattern data 31:16 Value to be added for each successive sample for the test pattern data
TESTPATTERN_ RX3_QCFG	4	This field specifies the values for Rx3, Q channel. Applicable only in case of test pattern enable. Else ignored. Bits Description 15:0 Start offset value to be used for the first sample for the test pattern data 31:16 Value to be added for each successive sample for the test pattern data
RESERVED	4	0x00000000

5.16.13 Sub block 0x404C – AWR_DEV_CONFIGURATION_SET_SB

This API is used to configure the CRC type for the async events from MSS. The default is 16 bit CRC if this API is not issued. The first async event after MSS powerup will have a 16 bit CRC.

Table 5.111 – AWR_DEV_CONFIGURATION_SET_SB

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x404C
SBLKLEN	2	Value = 12
ASYNC_EVENT_ CRC_CFG	1	Value Description 0 16 bit CRC for MSS async events 1 32 bit CRC for MSS async events 2 64 bit CRC for MSS async events
RESERVED1	1	0x00
RESERVED2	2	0x0000
RESERVED3	4	0x00000000
RESERVED4	4	0x00000000

5.17 Sub blocks related to AWR_DEV_CONF_GET_MSG

5.17.1 Sub block 0x4060 – AWR_DEV_MCUCLOCK_GET_SB

This API is used to read the MCU clock configuration. Response packet structure will be same as AWR_DEV_MCUCLOCK_SET_SB

Table 5.112 – AWR_DEV_MCUCLOCK_GET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x4060
SBLKLEN	2	Value = 4

5.17.2 Sub block 0x4061 – AWR_DEV_RX_DATA_FORMAT_CONF_GET_SB

This API is used to read the RX data format configuration. Response packet structure will be same as AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB

Table 5.113 – AWR_DEV_RX_DATA_FORMAT_CONF_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4061
SBLKLEN	2	Value = 4

5.17.3 Sub block 0x4062 – AWR_DEV_RX_DATA_PATH_CONF_GET_SB

This API is used to read the RX data path configuration. Response packet structure will be same as AWR_DEV_RX_DATA_PATH_CONF_SET_SB

Table 5.114 – AWR_DEV_RX_DATA_PATH_CONF_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4062
SBLKLEN	2	Value = 4

5.17.4 Sub block 0x4063 – AWR_DEV_RX_DATA_PATH_LANEEN_GET_SB

This API is used to read the RX data path lane enable configuration. Response packet structure will be same as AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB

Table 5.115 – AWR_DEV_RX_DATA_PATH_LANEEN_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4063
SBLKLEN	2	Value = 4

5.17.5 Sub block 0x4064 – AWR_DEV_RX_DATA_PATH_CLK_GET_SB

This API is used to read the RX data path clock configuration. Response packet structure will be same as AWR_DEV_RX_DATA_PATH_CLK_SET_SB

Table 5.116 – AWR_DEV_RX_DATA_PATH_CLK_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4064

SBLKLEN	2	Value = 4

5.17.6 Sub block 0x4065 – AWR_DEV_LVDS_CFG_GET_SB

This API is used to read the LVDS configuration. Response packet structure will be same as AWR_DEV_LVDS_CFG_SET_SB

Table 5.117 – AWR_DEV_LVDS_CFG_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4065
SBLKLEN	2	Value = 4

5.17.7 Sub block 0x4066 – AWR_DEV_RX_CONTSTREAMING_MODE_CONF_GET_SB

This API is used to read the continuous streaming mode configuration. Response packet structure will be same as AWR_DEV_RX_CONTSTREAMING_MODE_CONF_SET_SB

Table 5.118 – AWR_DEV_RX_CONTSTREAMING_MODE_CONF_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4066
SBLKLEN	2	Value = 4

5.17.8 Sub block 0x4067 – AWR_DEV_CSI2_CFG_GET_SB

This API is used to read the CSI2 configuration. Response packet structure will be same as AWR_DEV_CSI2_CFG_SET_SB

Table 5.119 – AWR_DEV_CSI2_CFG_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4067
SBLKLEN	2	Value = 4

5.17.9 Sub block 0x4068 – AWR_DEV_PMICCLOCK_CONF_GET_SB

This API is used to read the PMIC clock configuration. Response packet structure will be same as AWR_DEV_PMICCLOCK_CONF_SET_SB

Table 5.120 – AWR_DEV_PMICCLOCK_CONF_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4068
SBLKLEN	2	Value = 4

5.17.10 Sub block 0x4069 – AWR_MSS_LATENTFAULT_TEST_CONF_GET_SB

This API is used to read the MSS latent fault test configuration. Response packet structure will be same as AWR_MSS_LATENTFAULT_TEST_CONF_SET_SB

Table 5.121 – AWR_MSS_LATENTFAULT_TEST_CONF_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4069
SBLKLEN	2	Value = 4

5.17.11 Sub block 0x406A – AWR_MSS_PERIODICTESTS_CONF_GET_SB

This API is used to read the MSS periodic tests configuration. Response packet structure will be same as AWR_MSS_PERIODICTESTS_CONF_SET_SB

Table 5.122 – AWR_MSS_PERIODICTESTS_CONF_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x406A
SBLKLEN	2	Value = 4

5.17.12 Sub block 0x406B – AWR_DEV_TESTPATTERN_GEN_GET_SB

This API is used to read the test pattern generation configuration. Response packet structure will be same as AWR_DEV_TESTPATTERN_GEN_SET_SB

Table 5.123 – AWR_DEV_TESTPATTERN_GEN_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x406B
SBLKLEN	2	Value = 4

5.18 Sub blocks related to AWR_DEV_FILE_DOWNLOAD_MSG

5.18.1 Sub block 0x4080 – AWR_DEV_FILE_DOWNLOAD_SB

This sub block is used to send the file in chunks/parts for download into RAM.

Table 5.124 describes the content of this sub block.

Table 5.124 – AWR_DEV_FILE_DOWNLOAD_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x4080
SBLKLEN	2	Value = Variable
FILE_TYPE	4	Value Description 0x0 FILETYPE_BSS_BUILD 0x1 FILETYPE_CALIB_DATA 0x2 FILETYPE_CONFIG_INFO 0x3 FILETYPE_MSS_BUILD
FILE_LENGTH	4	Length of File
FILE_CONTENT	Variable	Content of File, may split into multiple chunks.

NOTE: In the first chunk of file, FILE_TYPE and FILE_LENGTH is available and then first chunk onward these two fields will not be part of SB content

5.19 Sub blocks related to AWR_DEV_FRAME_CONFIG_APPLY_MSG

5.19.1 Sub block 0x40C0 – AWR_DEV_FRAME_CONFIG_APPLY_SB

This sub block is used to indicate to MSS to apply all the device configurations in the hardware.

Table 5.125 describes the content of this sub block.

Table 5.125 – AWR_DEV_FRAME_CONFIG_APPLY_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x40C0
SBLKLEN	2	Value = 12
NUM_CHIRPS	4	Number of chirps per frame
HALF_WORDS_ PER_CHIRP	2	Number of half words in ADC buffer per chirp Example 1: In real mode, if number of ADC samples per chirp is 256 then this value will be 256 Example 2: In complex1x or complex2x modes, if number of ADC samples per chirp is 256 then this value will be 512
RESERVED	2	0x0000

5.19.2 Sub block 0x40C1 – AWR_DEV_ADV_FRAME_CONFIG_APPLY_SB

This sub block is used to indicate to MSS to apply all the advanced frame configuration configurations in the hardware.

Table 5.126 describes the content of this sub block.

Table 5.126 – AWR_DEV_ADV_FRAME_CONFIG_APPLY_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x40C1
SBLKLEN	2	Value = 40
NUM_SUBFRAMES	1	Number of sub frames enabled in this frame Valid range: 1 to 4
RESERVED	3	0x00
SF1_TOT_NUM_ CHIRPS	4	Number of chirps in sub frame 1

SF1_NUM_ADC_ SAMPLES_PER_ DATA_PKT	2	Number of half words of ADC samples per data packet in sub frame 1 Example 1: In real mode, if number of ADC samples per chirp in subframe1 is 256 then this value will be 256 Example 2: In complex1x or complex2x modes, if number of ADC samples per chirp in subframe1 is 256 then this value will be 512 In AWR12xx: Program this as the same as number of ADC samples in each chirp of this sub frame (required to be the same) Exception: Can do #chirps based ping-pong as in AWR16xx (see below), if CP/CQ are not needed. Useful for chirp stitching use case. In AWR16xx: The ADC samples corresponding to one or more chirps can be grouped and sent to the DSP as a single packet. Program this as the number of half words of ADC samples per packet. Ensure that in one sub frame, there is integer number of such packets. Maximum size of a data packet: (16384 - 1) half words.
SF1_PROC_NUM_ CHIRPS_PER_ DATA_PKT	1	Number of Chirps Per Data Packet to process at a time in sub frame 1. In AWR12xx: Program this as 1. Exception: Can be > 1 as in 16xx if CP/CQ is not needed. Useful for chirp stitching use case. In AWR16xx: The ADC samples corresponding to one or more chirps can be grouped and sent to the DSP as a single packet. Program this as the corresponding number of chirps per packet. Maximum value = 8. Note on maximum size: 8 chirps for CP and BPM.
RESERVED	1	0x00
SF2_TOT_NUM_ CHIRPS	4	Number of chirps in sub frame 2
SF2_NUM_ADC_ SAMPLES_PER_ DATA_PKT	2	Number of ADC Samples per data packet in sub frame 2 Same conditions apply as in sub frame 1.
SF2_PROC_NUM_ CHIRPS_PER_ DATA_PKT	1	Number of chirps per data packet to process at a time in sub frame 2 Same conditions apply as in Sub Frame 1.
RESERVED	1	0x00
SF3_TOT_NUM_ CHIRPS	4	Number of chirps in Sub frame3

SF3_NUM_ADC_ SAMPLES_PER_ DATA_PKT	2	Number of ADC samples per data packet in sub frame 3 Same conditions apply as in Sub Frame 1.
SF3_PROC_NUM_ CHIRPS_PER_ DATA_PKT	1	Number of chirps per data packet to process at a time in sub frame 3 Same conditions apply as in Sub Frame 1.
RESERVED	1	0x00
SF4_TOT_NUM_ CHIRPS	4	Number of chirps in Sub frame4
SF4_NUM_ADC_ SAMPLES_PER_ DATA_PKT	2	Number of ADC samples per data packet in sub frame 4 Same conditions apply as in sub frame 1.
SF4_PROC_NUM_ CHIRPS_PER_ DATA_PKT	1	Number of chirps per data packet to process at a time in sub frame 4 Same conditions apply as in sub frame 1.
RESERVED	1	0x00

5.20 Sub blocks related to AWR_DEV_STATUS_GET_MSG

5.20.1 Sub block 0x40E0 – AWR_MSSVERSION_GET_SB

This sub block reads MSS FW version. The information returned by the device will be in the format as given in AWR_MSSVERSION_SB.

Table 5.127 describes the contents of the request sub block

Table 5.127 – AWR_MSSVERSION_GET_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x40E0
SBLKLEN	2	Value = 4

Response to AWR_MSSVERSION_GET_SB

AWR_MSSVERSION_SB sub block is sent by the radar device in response to AWR_MSSVERSION_GET_SB. Note that SBLKID for both AWR_MSSVERSION_GET_SB and AWR_MSSVERSION_SB are same.

Table 5.128 describes the contents of the response sub block.

Table 5.128 – AWR_MSSVERSION_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x40E0
SBLKLEN	2	Value = 20
HW_VARIANT	1	HW variant number
HW_VERSION_ MAJOR	1	HW version major number
HW_VERSION_ MINOR	1	HW version minor number
MSS_FW_ VERSION_MAJOR	1	MSS FW version major number
MSS_FW_ VERSION_MINOR	1	MSS FW version minor number
MSS_FW_ VERSION_BUILD	1	MSS FW version build number
MSS_FW_ VERSION_DEBUG	1	MSS FW version debug number
MSS_FW_ VERSION_YEAR	1	Year of MSS FW version release
MSS_FW_ VERSION_MONTH	1	Month of MSS FW version release
MSS_FW_ VERSION_DAY	1	Day of MSS FW version release
MSS_FW_ VERSION_PATCH_ MAJOR	1	MSS FW version patch major number
MSS_FW_ VERSION_PATCH_ MINOR	1	MSS FW version patch minor number
MSS_FW_ VERSION_PATCH_ YEAR	1	Year of MSS FW patch release
MSS_FW_ VERSION_PATCH_ MONTH	1	Month of MSS FW patch release
MSS_FW_ VERSION_PATCH_ DAY	1	Day of MSS FW patch release
RESERVED	1	0x00

5.20.2 Sub block 0x40E1 – AWR_MSSCPUFAULT_STATUS_GET_SB

This sub block provides the MSS CPU fault information.

Table 5.129 describes the content of this sub block.

Table 5.129 – AWR_MSSCPUFAULT_STATUS_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x40E1
SBLKLEN	2	Value = 4

Response to AWR_MSSCPUFAULT_STATUS_GET_SB

AWR_MSSCPUFAULT_STATUS_SB is sent in response to AWR_MSSCPUFAULT_STATUS_GET_SB.

Table 5.130 describes the content of AWR_MSSCPUFAULT_STATUS_SB

Table 5.130 – AWR_MSSCPUFAULT_STATUS_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x40E1
SBLKLEN	2	Value = 36
FAULT_TYPE	1	0 MSS Processor Undefined Instruction Abort 1 MSS Processor Instruction pre-fetch Abort 2 MSS Processor Data Access Abort 3 MSS Processor Firmware Fatal Error 0x4 - 0xFF Reserved
RESERVED	1	0x00
LINE_NUM	2	Valid only in case of FAULT type is 0x3, provides the firmware line number at which fatal error occurred.
FAULT_LR	4	The instruction PC address at which Fault occurred
FAULT_PREV_LR	4	The return address of the function from which fault function has been called (Call stack LR)
FAULT_SPSR	4	The CPSR register value at which fault occurred
FAULT_SP	4	The SP register value at which fault occurred
FAULT_CAUSE_ ADDRESS	4	The address access at which Fault occurred (valid only for fault type 0x0 to 0x2)

FAULT_ERROR_ STATUS	2	The status of Error (Error Cause type - valid only for fault type 0x0 to 0x2) 0x000 BACKGROUND_ERR 0x001 ALIGNMENT_ERR 0x002 DEBUG_EVENT 0x00D PERMISSION_ERR 0x008 SYNCH_EXTER_ERR 0x406 ASYNCH_EXTER_ERR 0x409 SYNCH_ECC_ERR 0x408 ASYNCH_ECC_ERR
FAULT_ERROR_ SOURCE	1	The Source of the Error (Error Source type - valid only for fault type 0x0 to 0x2) 0x0 ERR_SOURCE_AXI_MASTER 0x1 ERR_SOURCE_ATCM 0x2 ERR_SOURCE_BTCM
FAULT_AXI_ERROR _TYPE	1	The AXI Error type (Error Source type - valid only for fault type 0x0 to 0x2) 0x0 AXI_DECOD_ERR 0x1 AXI_SLAVE_ERR
FAULT_ACCESS_ TYPE	1	The Error Access type (Error Access type - valid only for fault type 0x0 to 0x2) 0x0 READ_ERR 0x1 WRITE_ERR
FAULT_RECOVERY _TYPE	1	The Error Recovery type (Error Recovery type - Valid only for fault type 0x0 to 0x2) 0x0 UNRECOVERY 0x1 RECOVERY
RESERVED	2	0x0000

5.20.3 Sub block 0x40E2 – AWR_MSSESMFAULT_STATUS_GET_SB

This sub block provides the information regarding additional Master sub system faults.

Table 5.131 describes the content of this sub block.

Table 5.131 – AWR_MSSESMFAULT_STATUS_GET_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x40E2
SBLKLEN	2	Value = 4

The Response to above request is given in the AWR_MSSESMFAULT_STATUS_SB.

Table 5.132 describes the contents of AWR_MSSESMFAULT_STATUS_SB.

Table 5.132 – AWR_MSSESMFAULT_STATUS_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x40E2
SBLKLEN	2	Value = 20
ESM_GROUP1_ ERRORS	4	TBD
RESERVED	4	0x00000000
RESERVED	4	0x00000000
RESERVED	4	0x00000000

5.21 Sub blocks related to AWR_DEV_ASYNC_EVENT_MSG

5.21.1 Sub block 0x5000 – AWR_AE_DEV_MSSPOWERUPDONE_SB

This sub block indicates that BIST SS power up is now complete.

Table 5.133 describes the contents of this sub block

Table 5.133 – AWR_AE_DEV_MSSPOWERUPDONE_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x5000
SBLKLEN	2	Value = 20
MSS_POWERUP_ BIST_STATUS_ FLAGS	4	1 – PASS, 0 – FAIL Value Status Information 0 ROM CRC check 1 TBD
POWERUP_TIME	4	Master SS power up time

		1 LSB = 5ns
RESERVED	4	0x00000000
RESERVED	4	0x00000000

5.21.2 Sub block 0x5001 – AWR_AE_DEV_RFPOWERUPDONE_SB

This sub block indicates that BIST SS power up is now complete.

Table 5.134 describes the contents of this sub block

Table 5.134 – AWR_AE_DEV_RFPOWERUPDONE_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x5001
SBLKLEN	2	Value = 20
BSS_POWERUP_BIST_STATUS_FLAGS	4	1 – PASS, 0 – FAIL Value Status Information 0 ROM CRC check 1 TBD
POWERUP_TIME	4	RF BIST SS Power up time 1 LSB = 5 ns
RESERVED	4	0x00000000
RESERVED	4	0x00000000

5.21.3 Sub block 0x5002 – AWR_AE_MSS_CPUFAULT_SB

This sub block indicates CPU fault status of Master SS.

Table 5.135 describes the content of this sub block.

Table 5.135 – AWR_AE_MSSCPUFAULT_STATUS_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x5002
SBLKLEN	2	Value = 36

FAULT_TYPE	1	0 MSS Processor Undefined Instruction Abort 1 MSS Processor Instruction pre-fetch Abort 2 MSS Processor Data Access Abort 3 MSS Processor Firmware Fatal Error 0x4 - 0xFF Reserved
RESERVED	1	0x00
LINE_NUM	2	Valid only in case of FAULT type is 0x3, provides the firmware line number at which fatal error occurred.
FAULT_LR	4	The instruction PC address at which Fault occurred
FAULT_PREV_LR	4	The return address of the function from which fault function has been called (Call stack LR)
FAULT_SPSR	4	The CPSR register value at which fault occurred
FAULT_SP	4	The SP register value at which fault occurred
FAULT_CAUSE_ ADDRESS	4	The address access at which Fault occurred (valid only for fault type 0x0 to 0x2)
FAULT_ERROR_ STATUS	2	The status of Error (Error Cause type - valid only for fault type 0x0 to 0x2) 0x000 BACKGROUND_ERR 0x001 ALIGNMENT_ERR 0x002 DEBUG_EVENT 0x00D PERMISSION_ERR 0x008 SYNCH_EXTER_ERR 0x406 ASYNCH_EXTER_ERR 0x409 SYNCH_ECC_ERR 0x408 ASYNCH_ECC_ERR
FAULT_ERROR_ SOURCE	1	The Source of the Error (Error Source type - valid only for fault type 0x0 to 0x2) 0x0 ERR_SOURCE_AXI_MASTER 0x1 ERR_SOURCE_ATCM 0x2 ERR_SOURCE_BTCM
FAULT_AXI_ ERROR_TYPE	1	The AXI Error type (Error Source type - valid only for fault type 0x0 to 0x2) 0x0 AXI_DECOD_ERR 0x1 AXI_SLAVE_ERR

FAULT_ACCESS_ TYPE	1	The Error Access type (Error Access type - valid only for fault type 0x0 to 0x2) 0x0 READ_ERR 0x1 WRITE_ERR
FAULT_ RECOVERY_TYPE	1	The Error Recovery type (Error Recovery type - Valid only for fault type 0x0 to 0x2) 0x0 UNRECOVERY 0x1 RECOVERY
RESERVED	2	0x0000

5.21.4 *Sub block 0x5003 – AWR_AE_MSS_ESMFAULT_SB*

This sub block indicates any other faults inside the MSS.

Table 5.136 describes the content of this sub block.

Table 5.136 – AWR_AE_MSSESMFAULT_STATUS_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x5003
SBLKLEN	2	Value = 12
ESM_GROUP1_ ERRORS	4	**Bits** / **Definition** 0 NERROR in sync 1 RESERVED 2 DMA MPU Region tests 3 DMA Parity error 4 RESERVED 5 RESERVED 6 Access error interrupt from FFT ACC 7 CSI TX FIFO Parity Err 8 TPCC parity error 9 CBUF ECC single bit error 10 RESERVED 11 RESERVED 12 RESERVED

		13	Error response from the Peripheral when a DMA transfer is done
		14	RESERVED
		15	VIM RAM double bit errors
		16	RESERVED
		17	MibSPI double bit error test
		18	ECC error on CBUFF
		19	RESERVED
		20	VIM RAM single bit errors
		21	RESERVED
		22	RESERVED
		23	RESERVED
		24	RESERVED
		25	MibSPI single bit error test
		26	TCMB RAM single bit errors
		27	STC error
		28	TCMB RAM single bit errors
		29	TPTC0 read to protected memory
		30	DCC compare error
		31	CR4F self-test error.(test of error path by error forcing)
ESM_GROUP2_ ERRORS	4	**Bits**	**Definition**
		0	TCMA RAM single bit errors
		1	RESERVED
		2	RESERVED
		3	TPTC0 write to protected memory
		4	RESERVED
		5	RESERVED
		6	Access error interrupt from FFT ACC
		7	VIM Self-Test Error
		8	RESERVED
		9	RESERVED
		10	RESERVED

		11	RESERVED
		12	QSPI not able to perform the Write to FLASH
		13	RESERVED
		14	RESERVED
		15	RESERVED
		16	RESERVED
		17	RESERVED
		18	RESERVED
		19	RESERVED
		20	RESERVED
		21	RESERVED
		22	RESERVED
		23	RESERVED
		24	RESERVED
		25	RESERVED
		26	BSS Mailbox single bit errors
		27	BSS Mailbox double bit errors
		28	MSS Mailbox single bit errors
		29	MSS Mailbox double bit errors
		30	RESERVED
		31	RESERVED
RESERVED	4	0x00000000	

5.21.5 Sub block 0x5004 – RESERVED

5.21.6 Sub block 0x5005 – AWR_AE_MSS_BOOTERRORSTATUS_SB

This sub block indicates error status of MSS when booted over SPI.

Table 5.137 describes the content of this sub block.

Table 5.137 – AWR_AE_MSS_BOOTERRORSTATUS_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x5005
SBLKLEN	2	Value = 12
ERROR_STATUS	4	Refer Table 6.3 for the bitmap of error status
RESERVED	4	Reserved

5.21.7 Sub block 0x5006 – AWR_AE_MSS_LATENTFAULT_TESTREPORT_SB

This sub block indicates the test status report of the latent fault tests.

Table 5.138 describes the content of this sub block.

Table 5.138 – AWR_AE_MSS_LATENTFAULT_TESTREPORT_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x5006
SBLKLEN	2	Value = 16

TEST_STATUS_ FLAG1	4	1 – PASS, 0 - FAIL

Bits	Definition
0	MibSPI self-test
1	DMA self-test
2	Reserved
3	RTI self-test
4	ESM self-test
5	EDMA self-test
6	CRC self-test
7	VIM self-test
8	Reserved
9	Mailbox self-test
10	LVDS pattern generation test
11	CSI2 pattern generation test
12	Generating NERROR
13	MibSPI single bit error test
14	MibSPI double bit error test
15	DMA Parity error
16	TCMA RAM single bit errors
17	TCMB RAM single bit errors
18	TCMA RAM double bit errors
19	TCMB RAM double bit errors
20	TCMA RAM parity errors.
21	TCMB RAM parity errors.
22	Reserved
23	Reserved
24	DMA MPU Region tests
25	MSS Mailbox single bit errors
26	MSS Mailbox double bit errors
27	BSS Mailbox single bit errors
28	BSS Mailbox double bit errors
29	EDMA MPU test
30	EDMA parity test
31	CSI2 parity test

TEST_STATUS_ FLAG2	4	**Bits**	**Definition**
		0	DCC self-test
		1	DCC fault insertion test
		2	PCR fault generation test
		3	VIM RAM parity test
		4	SCI boot time test
		31:5	RESERVED
RESERVED	4	0x00000000	

5.21.8 Sub block 0x5007 – AWR_AE_MSS_PERIODICTEST_STATUS_SB

This sub block indicates test status of the periodic tests.

Table 5.139 describes the content of this sub block.

Table 5.139 – AWR_AE_MSS_PERIODCTEST_STATUS_SB contents

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x5007	
SBLKLEN	2	Value = 12	
TEST_STATUS_ FLAG	4	1 – PASS, 0 – FAIL	
		Bits	**Definition**
		0	Periodic read back of static registers
		1	ESM self-test
		31:2	RESERVED
RESERVED	4	0x00000000	

5.21.9 Sub block 0x5008 – AWR_AE_MSS_RFERROR_STATUS_SB

This sub block indicates the RF error status.

Table 5.140 describes the content of this sub block.

Table 5.140 – AWR_AE_MSS_RFERROR_STATUS_SB contents

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x5008	
SBLKLEN	2	Value = 12	
ERROR_STATUS_FLAG	4	**Value**	**Definition**
		0	No fault
		1	BSS FW assert
		2	BSS FW abort
		3	BSS ESM GROUP1 ERROR
		4	BSS ESM GROUP2 ERROR
		Others	RESERVED
RESERVED	4	0x00000000	

5.21.10 Sub block 0x5009 – AWR_AE_MSS_VMON_ERRORSTATUS_SB

This sub block indicates fault in analog supplies or LDO short circuit condition. Once a fault is detected the functionality cannot be resumed from then on and the sensor needs to be re-started.

Table 5.141 – AWR_AE_MSS_VMON_ERRORSTATUS_SB

Field Name	Number of bytes	Description	
SBLKID	2	Value = 0x5009	
SBLKLEN	2	Value = 16	
FAULT_TYPE	1	**Value**	**Definition**
		0	NO FAULT
		1	ANALOG_SUPPLY_FAULT
		Others	RESERVED
RESERVED	1	0x00	
RESERVED	2	0x0000	
FAULT_SIG	4	**Bit**	**Definition**
		0	VDDIN under voltage indication
		1	VDDIN over voltage indication

		2	VIN_18CLK supply fault
		3	VIOIN supply fault (Unable to resolve between 1.8V and 3.3V)
		4	VIN_SRAM under voltage indication
		5	VIOIN_18 supply fault
		6	APLL_VCO_LDO short circuit
		31:7	RESERVED
RESERVED	4	0x00000000	

5.21.10 Sub block 0x500A – AWR_AE_MSS_ADC_DATA_SB

This async event is in response to the command which indicates ADC data needs to be transferred over SPI. This async event contains the ADC data followed by more such async events for additional data.

Table 5.142 – AWR_AE_MSS_ADC_DATA_SB

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x500A
SBLKLEN	2	Value = Variable (max = 226)
REMCHUNKS	2	Number of remaining chunks expected. (Remaining length / 220 bytes)
ADC_DATA	Variable (4 - 220 bytes)	ADC data captured by the MMIC

5.21.11 Sub block 0x500B – RESERVED

5.22 Brief notes on the order of issuing API SBs

5.22.1 Single device mode

This section briefly describes in which order to issue the various API SBs defined in this document for a single device.

1. Power up the device

2. Wait for AWR_AE_MSSPOWERUPDONE_SB
3. AWR_DEV_RFPOWERUP_SB
4. Wait for AWR_AE_RFPOWERUPDONE_SB
5. AWR_RF_MISC_CONF_SET_MSG
   a. AWR_RF_LDO_BYPASS_SB with RFLDOBYPASS_EN set to 1 if RF supply is 1.0V
6. AWR_RF_STATIC_CONF_SET_MSG
   a. AWR_CHAN_CONF_SET_SB
   b. AWR_ADCOUT_CONF_SET_SB
   c. AWR_LOWPOWERMODE_CONF_SET_SB
   d. AWR_DYNAMICPOWERSAVE_CONF_SET_SB
7. Data path configurations
   a. AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB
   b. AWR_DEV_RX_DATA_PATH_CONF_SET_SB
   c. AWR_DEV_RX_DATA_PATH_LANE_EN_SB
   d. AWR_DEV_RX_DATA_PATH_CLK_SET_SB
   e. AWR_HIGHSPEEDINTFCLK_CONF_SET_SB
   f. AWR_DEV_LVDS_CFG_SET_SB / AWR_DEV_CSI2_CFG_SET_SB
8. AWR_RF_INIT_MSG
   a. AWR_RFINIT_SB --> This triggers very basic calibrations and RF initializations
   b. Wait for AWR_AE_RF_INITCALIBSTATUS_SB
9. AWR_RF_DYNAMIC_CONF_SET_MSG
   a. AWR_PROG_FILT_COEFF_RAM_SET_SB (Applicable only in AWR1642)
   b. AWR_PROG_FILT_CONF_SET_SB (Applicable only in AWR1642)
   c. AWR_LOOPBACK_BURST_CONF_SET_SB
   d. AWR_PROFILE_CONF_SET_SB
   e. AWR_CHIRP_CONF_SET_SB
   f. AWR_FRAME_CONF_SET_SB
   g. AWR_CALIB_MON_TIME_UNIT_CONF_SB with CALIB_MON_TIME_UNIT value set to a value such that the total frame idle time across multiple CALIB_MON_TIME_UNITs is

sufficient for all calibrations and monitoring. See Section 9 for details on calibration and monitoring durations.

h. AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB (set all ONE_TIME_CALIB_ENABLE_MASK and set ENABLE_CAL_REPORT = 1)

i. Wait for AWR_RUN_TIME_CALIBRATION_SUMMARY_REPORT_AE_SB

j. AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB (set all RUN_TIME_CALIB_ENABLE_MASK and set ENABLE_CAL_REPORT = 0 to avoid receiving periodic async events)

k. AWR_DEV_FRAME_CONFIG_APPLY_MSG

10. AWR_RF_FRAME_TRIG_MSG

    a. AWR_FRAMESTARTSTOP_CONF_SB in Start mode --> after this, frames get transmitted

11. AWR_RF_FRAME_TRIG_MSG

    a. AWR_FRAMESTARTSTOP_CONF_SB in Stop mode --> after this, frames are stopped

    The AWR_RF_FRAME_TRIG_MSG may be issued multiple times for multiple sets of frames.

5.22.2 Cascaded device mode

This section briefly describes in which order to issue the various API SBs defined in this document for master and slave devices in a cascaded configuration.

When using cascaded devices, the reference clock is provided by master to slave. So unless master is powered-up and clock is available from master to slave, the slave device cannot be powered up.

Table 5.143 – Sequence of APIs to be issued to master and slave devices in cascaded mode configuration for FMCW mode measurements

Sl. No.	Master device sequence	Slave device sequence
1	Power up master device	
2	Wait for AWR_AE_DEV_MSSPOWERUPDONE_SB	
3	AWR_DEV_RFPOWERUP_SB	
4	Wait for AWR_AE_DEV_RFPOWERUPDONE_SB	
5	AWR_RF_LDO_BYPASS_SB with RFLDOBYPASS_EN = 1 if RF supply is 1.0V	
6	AWR_CHAN_CONF_SET_SB with	

	CASCADING_CFG = 0x0001. This will enable the reference clock for slave device	
7		Power on slave device
8		Wait for AWR_AE_DEV_MSSPOWERUPDONE_SB
9		AWR_DEV_RFPOWERUP_SB
10		Wait for AWR_AE_DEV_RFPOWERUPDONE_SB
11		AWR_RF_LDO_BYPASS_SB with RFLDOBYPASS_EN = 1 if RF supply is 1.0V
12		AWR_CHAN_CONF_SET_SB with CASCADING_CFG = 0x0002.
13	AWR_ADCOUT_CONF_SET_SB	AWR_ADCOUT_CONF_SET_SB
14	AWR_LOWPOWERMODE_CONF_SET_SB	AWR_LOWPOWERMODE_CONF_SET_SB
15	AWR_DYNAMICPOWERSAVE_CONF_SET_SB	AWR_DYNAMICPOWERSAVE_CONF_SET_SB
16	AWR_RF_INIT_SB	AWR_RF_INIT_SB
17	Wait for AWR_AE_RF_INITALIBSTATUS_SB	Wait for AWR_AE_RF_INITALIBSTATUS_SB
18	AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB	AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB
19	AWR_DEV_RX_DATA_PATH_CONF_SET_SB	AWR_DEV_RX_DATA_PATH_CONF_SET_SB
20	AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB	AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB
21	AWR_HIGHSPEEEDINTFCLK_CONF_SET_SB	AWR_HIGHSPEEEDINTFCLK_CONF_SET_SB
22	AWR_DEV_RX_DATA_PATH_CLK_SET_SB	AWR_DEV_RX_DATA_PATH_CLK_SET_SB
23	AWR_DEV_LVDS_CFG_SET_SB/AWR_DEV_CSI2_CFG_SET_SB	AWR_DEV_LVDS_CFG_SET_SB/AWR_DEV_CSI2_CFG_SET_SB
24	AWR_PROFILE_CONF_SET_SB	AWR_PROFILE_CONF_SET_SB
25	AWR_CHIRP_CONF_SET_SB	AWR_CHIRP_CONF_SET_SB
26	AWR_FRAME_CONF_SET_SB with TRIGGER_SELECT = 0x0001	AWR_FRAME_CONF_SET_SB with TRIGGER_SELECT = 0x0002
27	AWR_DEV_FRAME_CONFIG_APPLY_MSG	AWR_DEV_FRAME_CONFIG_APPLY_MSG

28		AWR_FRAMESTARTSOP_CONF_SB with STARTSTOP_CMD = 0x0001
29		Wait for AWR_AE_RF_FRAME_TRIGGER_RDY_SB
30	AWR_FRAMESTARTSOP_CONF_SB with STARTSTOP_CMD = 0x0001	
31	Wait for AWR_AE_RF_FRAME_TRIGGER_RDY_SB	

5.22.3 Continuous streaming mode (in single device case)

This section briefly describes in which order to issue the various API SBs defined in this document to enable continuous streaming mode on a single device

1. Power up the device
2. Wait for AWR_AE_MSSPOWERUPDONE_SB
3. AWR_DEV_RFPOWERUP_SB
4. Wait for AWR_AE_RFPOWERUPDONE_SB
5. AWR_RF_MISC_CONF_SET_MSG
    b. AWR_RF_LDO_BYPASS_SB with RFLDOBYPASS_EN set to 1 if RF supply is 1.0V
6. AWR_RF_STATIC_CONF_SET_MSG
    a. AWR_CHAN_CONF_SET_SB
    b. AWR_ADCOUT_CONF_SET_SB
    c. AWR_LOWPOWERMODE_CONF_SET_SB
    d. AWR_DYNAMICPOWERSAVE_CONF_SET_SB
7. Data path configurations
    a. AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB
    b. AWR_DEV_RX_DATA_PATH_CONF_SET_SB
    c. AWR_DEV_RX_DATA_PATH_LANE_EN_SB
    d. AWR_DEV_RX_DATA_PATH_CLK_SET_SB
    e. AWR_HIGHSPEEDINTFCLK_CONF_SET_SB
    f. AWR_DEV_LVDS_CFG_SET_SB / AWR_DEV_CSI2_CFG_SET_SB
8. AWR_RF_INIT_MSG

c. AWR_RFINIT_SB --> This triggers very basic calibrations and RF initializations

d. Wait for AWR_AE_RF_INITCALIBSTATUS_SB

9. AWR_CONT_STREAMING_MODE_CONF_SET_SB
10. AWR_DEV_RX_CONTSTREAMING_MODE_CONF_SET_SB
11. AWR_CONT_STREAMING_MODE_EN_SB with CONT_STREAMING_EN = 0x0001 to start continuous streaming
12. AWR_CONT_STREAMING_MODE_EN_SB with CONT_STREAMING_EN = 0x0000 to stop continuous streaming
13. Repeat steps 9-11 for a different configuration

5.22.4 Continuous streaming (CW) mode (in cascaded device case)

Table 5.144 – Sequence of APIs to be issued to master and slave devices in cascaded mode for CW mode measurements

Sl. No.	Master device sequence	Slave device sequence
1	Power up master device	
2	Wait for AWR_AE_DEV_MSSPOWERUPDONE_SB	
3	AWR_DEV_RFPOWERUP_SB	
4	Wait for AWR_AE_DEV_RFPOWERUPDONE_SB	
5	AWR_RF_LDO_BYPASS_SB with RFLDOBYPASS_EN = 1 if RF supply is 1.0V	
6	AWR_CHAN_CONF_SET_SB with CASCADING_CFG = 0x0001. This will enable the reference clock for slave device	
7		Power on slave device
8		Wait for AWR_AE_DEV_MSSPOWERUPDONE_SB
9		AWR_DEV_RFPOWERUP_SB
10		Wait for AWR_AE_DEV_RFPOWERUPDONE_SB
11		AWR_RF_LDO_BYPASS_SB with RFLDOBYPASS_EN = 1 if RF supply is 1.0V
12		AWR_CHAN_CONF_SET_SB with CASCADING_CFG = 0x0002.

13	AWR_ADCOUT_CONF_SET_SB	AWR_ADCOUT_CONF_SET_SB
14	AWR_LOWPOWERMODE_CONF_SET_SB	AWR_LOWPOWERMODE_CONF_SET_SB
15	AWR_DYNAMICPOWERSAVE_CONF_SET_SB	AWR_DYNAMICPOWERSAVE_CONF_SET_SB
16	AWR_RF_INIT_SB	AWR_RF_INIT_SB
17	Wait for AWR_AE_RF_INITALIBSTATUS_SB	Wait for AWR_AE_RF_INITALIBSTATUS_SB
18	AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB	AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB
19	AWR_DEV_RX_DATA_PATH_CONF_SET_SB	AWR_DEV_RX_DATA_PATH_CONF_SET_SB
20	AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB	AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB
21	AWR_HIGHSPEEEDINTFCLK_CONF_SET_SB	AWR_HIGHSPEEEDINTFCLK_CONF_SET_SB
22	AWR_DEV_RX_DATA_PATH_CLK_SET_SB	AWR_DEV_RX_DATA_PATH_CLK_SET_SB
23	AWR_DEV_LVDS_CFG_SET_SB/AWR_DEV_CSI2_CFG_SET_SB	AWR_DEV_LVDS_CFG_SET_SB/AWR_DEV_CSI2_CFG_SET_SB
24	AWR_CONT_STREAMING_MODE_CONF_SET_SB	
25	AWR_CONT_STREAMING_MODE_EN_SB with CONT_STREAMING_EN = 0x0001 to start continuous streaming	
26		AWR_CONT_STREAMING_MODE_CONF_SET_SB with the same RF frequency configuration as in master device
27		AWR_CONT_STREAMING_MODE_EN_SB with CONT_STREAMING_EN = 0x0001 to start continuous streaming
28		AWR_CONT_STREAMING_MODE_EN_SB with CONT_STREAMING_EN = 0x0000 to stop continuous streaming
29	AWR_CONT_STREAMING_MODE_EN_SB with CONT_STREAMING_EN = 0x0000 to stop continuous streaming	
30	Repeat steps 24-29 for a different CW mode configuration	

6. API Error Codes

Table 6.1 – BSS API error codes

API sub block	Error Code	Description
Applicable to all API sub blocks	1	Incorrect API MSGID
	2	Sub block not found in the MSG
	3	Incorrect Sub block ID
	4	Incorrect Sub block length
	5	Incorrect Sub block data
	6	Error in processing the command
	7	Binary file CRC mismatch error
	8	Binary file type mismatch w.r.t. magic number
AWR_FRAMESTARTSTOP_CONF_ SB	20	Frames are already started when the FRAME_START command was issued
	21	Frames are already stopped when the FRAME_STOP command was issued
	22	No valid frame configuration API was issued and frames are started
	23	START_STOP_CMD parameter is out of range
AWR_CHAN_CONF_SET_SB	24	RX_CHAN_EN parameter is out of range (Max range may vary based on device variant)
	25	TX_CHAN_EN parameter is out of range (Max range may vary based on device variant)
	26	CASCADING_CFG parameter is out of range [0, 2]
	130	Device variant does not allow cascading but API is issued to enable cascading mode
AWR_ADCOUT_CONF_SET_SB	27	NUM_ADC_BITS parameter is out of range [0, 2]
	28	ADC_OUT_FMT parameter is out of range [0, 3]
	127	FULL_SCALE_REDUCTION_FACTOR is > 0 for 16 bit ADC, or > 2 for 14 bit ADC mode or > 4 for 12 bit ADC mode
AWR_LOWPOWERMODE_CONF_ SET_SB	29	LP_ADC_MODE parameter is out of range [0, 1]
AWR_DYNAMICPOWERSAVE_CONF_SET_	30	BLOCK_CFG parameter is out of range [0, 7]

SB		
AWR_HIGHSPEEDINTFCLK_CONF_SET_ SB	31	HSICLKRATECODE[1:0] is 0
	32	RESERVED
	33	HSICLKRATECODE[3:2] is 3 and HSICLKRATECODE[1:0] is 2
	34	HSICLKRATECODE[3:2] is 3 and HSICLKRATECODE[1:0] is 2
AWR_PROFILE_CONF_SET_SB	35	PF_INDX is $\geq$ 4
	36	PF_FREQ_START_CONST is not within [76, 81] GHz
	37	PF_IDLE_TIME_ CONST > 5.24ms
	38	Maximum DFE spill time > PF_IDLE_TIME_CONST
	39	PF_ADC_START_ TIME_CONST > 4095
	40	PF_RAMP_END_ TIME > 524287
	41	PF_RAMP_END_ TIME < PF_ADC_START_TIME_CONST + ADC_SAMPLING_TIME (ADC_SAMPLING_TIME is time taken to sample NUM_ADC_SAMPLES)
	42	PF_TX_OUTPUT_ POWER_BACKOFF for TX0 ＞30
	43	PF_TX_OUTPUT_ POWER_BACKOFF for TX1 ＞30
	44	PF_TX_OUTPUT_ POWER_BACKOFF for TX2 ＞30
	45	RESERVED
	46	Ramp end frequency is not within [76, 81] GHz
	47	Absolute value of TX_START_TIME is > 38.45us
	48	Number of ADC samples is not within [64, 8192]
	49	Output sampling rate is not within [2, 37.5] Msps
	50	HPF1 corner frequency is > 700 kHz
	51	HPF2 corner frequency is > 2.8 MHz
	52	PF_RX_GAIN is not within [24, 52] dB or PF_RX_GAIN is an odd number

	53	RESERVED
	54	RESERVED
	55	RESERVED
	56	RESERVED
	57	RESERVED
	58	RESERVED
AWR_CHIRP_CONF_SET_SB	59	CHIRP_START_INDX ≥ 512
	60	CHIRP_END_INDX ≥ 512
	61	CHIRP_START_INDX > CHIRP_END_INDX
	62	PROFILE_INDX ≥ 4
	63	If the profile corresponding to PROFILE_INDX is not defined
	64	CHIRP_FREQ_START_VAR > 8388607
	65	CHIRP_FREQ_SLOPE_VAR > 63
	66	Chirp start frequency is outside[76, 81] GHz or chirp end frequency is outside [76, 81] GHz
	67	CHIRP_IDLE_TIME_VAR > 4095
	68	CHIRP_ADC_START_TIME_VAR > 4095
	69	RAMP_END_TIME < ADC_START_TIME + ADC_SAMPLING_TIME
	70	CHIRP_TX_EN > 7
	71	CHIRP_TX_EN indicates to enable a TX which is not enabled in AWR_CHAN_CONF_SET_SB
AWR_FRAME_CONF_SET_SB	72	CHIRP_START_INDX ≥ 512
	73	CHIRP_END_INDX ≥ 512
	74	CHIRP_START_INDX > CHIRP_END_INDX
	75	Chirp used in the frame is not configured by AWR_CHIRP_CONF_SET_SB
	76	One of the profiles used in the frame is not configured by AWR_PROF_CONF_SET_SB
	77	NUM_LOOPS is outside [1, 255]
	78	RESERVED
	79	FRAME_PERIODICITY is outside [100us, 1.342s]
	80	FRAME_ON_TIME < FRAME_PERIODICITY

	81	TRIGGER_SELECT is outside [1, 2]
	82	FRAME_TRIGGER_DELAY $>$ 100 us
	83	API is issued when frames are ongoing
AWR_ADVANCED_FRAME_CONF_SET_SB	84	NUM_SUBFRAMES is outside [1, 4]
	85	FORCE_SINGLE_PROFILE is outside [0, 1]
	86	FORCE_SINGLE_PROFILE $\geq$ 4
	87	Profile defined by FORCE_SINGLE_PROFILE is not defined
	88	SFx_CHIRP_START_INDX $\geq$ 512
	89	SFx_NUM_UNIQUE_CHIRPS_PER_BURST is outside the range [1, 512]
	90	Chirp used in the frame is not configured by AWR_CHIRP_CONF_SET_SB
	91	One of the profiles used in the frame is not configured by AWR_PROF_CONF_SET_SB
	92	SFx_NUM_LOOPS_PER_BURST is outside the range [1, 255]
	93	SFx_BURST_PERIOD is outside the range [100us, 1.342s]
	94	Burst on time is $>$ BURST_PERIOD
	95	SFx_CHIRP_START_INDX_OFFSET $\geq$ 512
	96	SFx_CHIRP_START_INDX $\geq$ 512 or SFx_CHIRP_START_INDX + SFx_NUM_UNIQUE_CHIRPS_PER_BURST – 1 is $\geq$ 512
	97	SFx_NUM_BURSTS is outside the range [1, 512]
	98	SFx_NUM_OUTER_LOOPS is outside the range [1, 64]
	99	SFx_PERIOD is outside the range [100us, 1.342s]
	100	Subframe on time $>$ SFx_PERIOD or when TESTSOURCE is enabled, SubFrame Idle time is $<$ 150 us
	101	RESERVED
	102	TRIGGER_SELECT is outside the range [1, 2]
	103	FRAME_TRIGGER_DELAY is $>$ 100 us
	104	API is issued when frames are on going

AWR_RF_TEST_SOURCE_CONFIG_SET_SB	105	POSITION_VECx[y] < 0
	106	RESERVED
	107	VELOCITY_VECx[x] >5000 or VELOCITY_VECx[y] >5000 or VELOCITY_VECx[z] >5000
	108	SIG_LEV_VECx >950
	109	RX_ANT_POS_XZ[Bytex] >120
	110	RESERVED
AWR_PROG_FILT_CONF_SET_SB	111	PROG_FILT_COEFF_START_INDEX is an odd number
	112	PROFILE_INDX ≥ 4
	126	DFE mode is pseudo real
AWR_PROG_FILT_COEFF_RAM_SET_SB	113	API is issued for a non AWR1642 device
	126	DFE mode is pseudo real
AWR_RF_RADAR_MISC_CTL_SB	114	API is issued for a non AWR1243 device
AWR_PERCHIRPPHASESHIFT_CONF_SB	115	CHIRP_START_INDX ≥ 512
	116	CHIRP_END_INDX ≥ 512
	117	CHIRP_START_INDX > CHIRP_END_INDX
AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB	118	Boot time calibrations are not done so cannot run runtime calibrations
AWR_CAL_MON_FREQUENCY_LIMITS_SB	119	FREQ_LIMIT_HIGH < 76 GHz or FREQ_LIMIT_HIGH > 81 GHz or FREQ_LIMIT_LOW > FREQ_LIMIT_HIGH
AWR_CALIB_MON_TIME_UNIT_CONF_SB	120	CALIB_MON_TIME_UNIT ≤ 0
AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB	121	CALIBRATION_ PERIODICITY = 0
	122	API is issued when continuous streaming mode is on
AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB	123	RX gain run time calibration was requested but boot time calibration was not performed
	124	LO distribution run time calibration was requested but boot time calibration was not performed
	125	TX power run time calibration was requested but boot time calibration was not performed
AWR_LOOPBACK_BURST_CONF_SET_SB	132	LOOPBACK_SEL is > 3
	133	BURST_INDX ≥ 16
	134	Burst is not valid but loopback is enabled for

		this burst
AWR_DYN_CHIRP_CONF_SET_SB	135	CHIRP_SEGMENT_SELECT > 31
AWR_DYN_PER_CHIRP_PHASESHIFTER_CONF_SB	136	CHIRP_SEGMENT_SELECT > 31
AWR_CAL_DATA_RESTORE_SB	137	CHUNK_ID ≥ NUM_CHUNKS
	138	CAL_DATA is invalid
AWR_INTERCHIRP_BLOCKCONTROLS_SB	139	RX02_RF_TURN_OFF_TIME is not within the range [-1024, 1023]
	140	RX13_RF_TURN_OFF_TIME is not within the range [-1024, 1023]
	141	RX02_BB_TURN_OFF_TIME is not within the range [-1024, 1023]
	142	RX13_BB_TURN_OFF_TIME is not within the range [-1024, 1023]
	143	RX02_RF_PREENABLE_TIME is not within the range [-1024, 1023]
	144	RX13_RF_PREENABLE_TIME is not within the range [-1024, 1023]
	145	RX02_BB_PREENABLE_TIME is not within the range [-1024, 1023]
	146	RX13_BB_PREENABLE_TIME is not within the range [-1024, 1023]
	147	RX02_RF_TURN_ON_TIME is not within the range [-1024, 1023]
	148	RX13_RF_TURN_ON_TIME is not within the range [-1024, 1023]
	149	RX02_BB_TURN_ON_TIME is not within the range [-1024, 1023]
	150	RX13_BB_TURN_ON_TIME is not within the range [-1024, 1023]
	151	RX_LO_TURN_OFF_TIME is not within the range [-1024, 1023]
	152	TX_LO_TURN_OFF_TIME is not within the range [-1024, 1023]
	153	RX_LO_TURN_ON_TIME is not within the range [-1024, 1023]
	154	TX_LO_TURN_ON_TIME is not within the range [-1024, 1023]
Common to all monitoring configuration APIs	250	Device type is not ASILB

	251	Fault injection API or Digital latent fault API is issued when frames are ongoing
	252	Invalid reporting mode
	253	Configured profile ID is not within [0, 3]
	254	Monitoring profile ID is not configured yet
	260	Invalid RF bit mask
	281	Analog monitoring is not supported
AWR_MONITOR_RF_DIG_LATENTFAULT_CONF_SB	251	API is issued when frames are on-going
AWR_MONITORING_EXTERNAL_ANALOG_SIGNALS_CONF_SB	255	Settling time is configured is more than 12us
AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_CONF_SB	256	None of the RXs are enabled
AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_CONF_SB	257	TX0 is not enabled
AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_CONF_SB	258	TX1 is not enabled
AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_CONF_SB	259	TX2 is not enabled
-	261	RESERVED
-	262	RESERVED
AWR_MONITOR_TXn_BALLBREAK_CONF_SB	263	Monitored TX is not enabled
AWR_MONITOR_RX_GAIN_PHASE_CONF_SB	264	Monitored RX is not enabled
AWR_MONITOR_RX_NOISE_FIGURE_CONF_SB		
AWR_MONITOR_RX_GAIN_PHASE_CONF_SB	265	TX selected for RX gain phase monitor is TX2 (Only TX0 or TX1 is allowed)
AWR_MONITOR_RX_SATURATION_DETECTOR_CONF_SB	266	SAT_MON_SEL is not in [0, 3]
	267	SAT_MON_PRIMARY_TIME_SLICE_DURATION is less than 0.64us or greater than ADC sampling time
	268	SAT_MON_NUM_SLICES is 0 or greater than 127
	283	RX saturation monitor is not supported
AWR_MONITOR_SIG_IMG_MONITOR_CONF_SB	269	SIG_IMG_MON_NUM_SLICES is 0 or greater than 127

	270	NUM_SAMPLES_ PER_PRIMARY_TIME_SLICE is odd, or less than 4 in Complex1x mode or less than 8 in non-Complex1x modes or greater than NUM_ADC_SAMPLES
	280	Signal and image band monitor is not supported
AWR_ANALOG_FAULT_INJECTION_CONF _SB	279	LDO fault inject is requested but LDOs are bypassed
AWR_MONITOR_TXn_POWER_CONF_SB	291	PD power level is less than -40dBm (Used for RX Gain Monitor)
	292	ADC power level higher than +7dBm or lower than -9.5dBm
	293	High RX noise figure (Noise Figure is less than 0 dB)
	294	PD Reading incorrect (RF OFF reading higher than RF ON reading)
	295	PGA Gain is non monotonic

Table 6.2 – MSS API error codes (Applicable for AWR1243)

API sub block	Error Code	Description
Applicable to all API sub blocks	1	Incorrect API MSGID
	2	Sub block not found in the MSG
	3	Incorrect Sub block ID
	4	Incorrect Sub block length
	5	Incorrect Sub block data
	6	Error in processing the command
	7	Binary file CRC mismatch error
	8	Binary file type mismatch w.r.t. magic number
AWR_DEV_RX_DATA_FORMAT_CONF_ SET_SB	1001	RX_CHAN_EN > 0xF
	1002	NUM_ADC_BITS > 2
	1003	ADC_OUT_FMT > 1
	1004	IQ_SWAP_SEL > 1
	1005	CHAN_INTERLEAVE > 1
AWR_DEV_RX_DATA_PATH_CONF_SET_ SB	1006	DATA_INTF_SEL > 1
	1007	DATA_TRANS_FMT_PKT0 [5:0] not a valid value. Valid set {0x1, 0x6, 0x9, 0x36}
	1008	DATA_TRANS_FMT_PKT1 [5:0] not a valid value. Valid set {0x0, 0xD, 0xB}
AWR_DEV_RX_DATA_PATH_LANEEN_ SET_SB	1009	LANE_EN > 0xF
	1010	Reserved
AWR_DEV_RX_DATA_PATH_CLK_SET_SB	1011	LANE_CLK_CFG > 1
	1012	LANE_CLK_CFG != 1 for CSI2
	1013	DATA_RATE – Invalid combination of data rate and DDR or SDR operation
AWR_DEV_LVDS_CFG_SET_SB	1014	LANE_FMT_MAP > 1
	1015	LANE_PARAM_CFG > 7
AWR_DEV_RX_CONTSTREAMING_MODE_ CONF_ SET_SB	1016	CONT_STREAMING_MODE > 1
	1017	CONT_STREAMING_MODE already in requested mode
AWR_DEV_CSI2_CFG_SET_SB	1018	LANE_POS_POL_SEL [DATA_LANE0_POS] > 5
	1019	LANE_POS_POL_SEL [DATA_LANE1_POS] > 5

	1020	LANE_POS_POL_SEL [DATA_LANE2_POS] ＞5
	1021	LANE_POS_POL_SEL [DATA_LANE3_POS] ＞5
	1022	LANE_POS_POL_SEL [CLOCK_POS] is outside the range [2,4]
AWR_DEV_FRAME_CONFIG_APPLY_SB	1023	HALF_WORDS_PER_CHIRP is outside the range [64, 8192]
AWR_DEV_ADV_FRAME_CONFIG_APPLY_SB	1024	NUM_SUBFRAMES is outside the range [1,4]
	1025	SF1_TOT_NUM_ CHIRPS is outside the range [1, 0xFFFF]
	1026	SF1_NUM_ADC_ SAMPLES_PER_ DATA_PKT is outside the range [64, 8192]
	1027	SF1_PROC_NUM_ CHIRPS_PER_ DATA_PKT != 1
	1028	SF2_TOT_NUM_ CHIRPS is outside the range [1, 0xFFFF], if NUM_SUBFRAMES ≥ 2
	1029	SF2_NUM_ADC_ SAMPLES_PER_ DATA_PKT is outside the range [64, 8192], if NUM_SUBFRAMES ≥2
	1030	SF2_PROC_NUM_ CHIRPS_PER_ DATA_PKT != 1, if NUM_SUBFRAMES ≥2
	1031	SF3_TOT_NUM_ CHIRPS is outside the range [1, 0xFFFF], if NUM_SUBFRAMES ≥3
	1032	SF3_NUM_ADC_ SAMPLES_PER_ DATA_PKT is outside the range [64, 8192], if NUM_SUBFRAMES ≥3
	1033	SF3_PROC_NUM_ CHIRPS_PER_ DATA_PKT != 1, if NUM_SUBFRAMES ≥3
	1034	SF4_TOT_NUM_ CHIRPS is outside the range [1, 0xFFFF], if NUM_SUBFRAMES == 4
	1035	SF4_NUM_ADC_ SAMPLES_PER_ DATA_PKT is outside the range [64, 8192], if NUM_SUBFRAMES == 4
	1036	SF4_PROC_NUM_ CHIRPS_PER_ DATA_PKT != 1, if NUM_SUBFRAMES == 4

6.1 Error codes for boot on SPI

Table 6.3 – Bit field describing the error status during boot on SPI

Error description	**Error code**	**Error code bit position**
CERT_AUTH_FAILURE	0x00000001	BIT0
CERT_PARSER_FAILURE	0x00000002	BIT1
RPRC_IMG1_AUTH_FAILURE	0x00000004	BIT2
RPRC_IMG2_AUTH_FAILURE	0x00000008	BIT3
RPRC_IMG3_AUTH_FAILURE	0x00000010	BIT4
RPRC_HDR_NOT_FOUND	0x00000020	BIT5
METAHEADER_NOT_FOUND	0x00000040	BIT6
SW_ANTIROLLBACK_CHK_FAILURE	0x00000080	BIT7
EFUSE_INTEGRITY_FAILURE	0x00000100	BIT8
CERT_FIELD_VALIDITY_FAILURE	0x00000200	BIT9
CERT_FIELD_INVALID_AUTH_KEY_INDEX	0x00000400	BIT10
CERT_FIELD_INVALID_HASH_TYPE	0x00000800	BIT11
CERT_FIELD_INVALID_SUBSYSTEM	0x00001000	BIT12
CERT_FIELD_INVALID_DECRYPT_KEY_INDEX	0x00002000	BIT13
CERT_FIELD_CEK_EFUSE_MISMATCH	0x00004000	BIT14
CERT_FIELD_CEK1_EFUSE_MISMATCH	0x00008000	BIT15
CERT_FIELD_CEK2_EFUSE_MISMATCH	0x00010000	BIT16
CERT_FIELD_INVALID_SUBSYSTEM_BANK_ ALLOCATION	0x00020000	BIT17
CERT_FIELD_INVALID_TOTAL_BANKS_ ALLOCATION	0x00040000	BIT18
RPRC_PARSER_FILE_LENGTH_MISMATCH	0x00080000	BIT19
RPRC_PARSER_MSS_FILE_OFFSET_MISMATCH	0x00100000	BIT20
RPRC_PARSER_BSS_FILE_OFFSET_MISMATCH	0x00200000	BIT21
RPRC_PARSER_DSS_FILE_OFFSET_MISMATCH	0x00400000	BIT22
CERT_FIELD_INVALID_DECRYPT_KEY	0x00800000	BIT23
CERT_FIELD_INVALID_AUTH_KEY	0x01000000	BIT24
HS_DEVICE_CERT_NOT_PRESENT	0x02000000	BIT25
TEST_PORT_ENABLING_FAILED	0x04000000	BIT26

SHARED_MEM_ALLOC_FAILED	0x08000000	BIT27
MSSIMAGE_NOT_FOUND	0x10000000	BIT28
METAHEADER_NUMFILES_ERROR	0x20000000	BIT29
METAHEADER_CRC_FAILURE	0x40000000	BIT30

7. Radar Monitoring APIs

AWR monitoring can be configured through a set of API Sub Blocks defined in this section. Note that these APIs cover the RF/Analog related monitoring mechanisms. There are separate monitoring mechanisms for the digital logic (including the processor, memory, etc.) which are internal to the device and not explicitly enabled through these APIs.

The monitoring APIs are structured as follows. There are common configuration APIs that control the overall periodicity of monitoring, as well as, enable/disable control for each monitoring mechanism. Then, for each monitoring mechanism there is an individual API to allow the customer to set an appropriate threshold for declaring failure from that monitoring. Also, for each monitoring mechanism, there is an individual API to report soft (raw) values from that monitoring.

7.1 Common Configurations and Reports

This section covers the APIs corresponding to the common configurations and reports.

7.1.1 *Sub block 0x01C0 – AWR_MONITOR_RF_DIG_LATENTFAULT_CONF_SB*

This API SB contains the consolidated configuration of all digital monitoring. This is issued by the host to the AWR device.

The enabled monitoring functions are executed when the API is issued. The scheduling of these monitoring should be handled in the external application. Report of these monitoring will be available in the async event AWR_MONITOR_RF_DIG_LATENTFAULT_REPORT_AE_SB.

Table 7.1 – AWR_MONITOR_RF_DIG_LATENTFAULT_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x1C0
SBLKLEN	2	Value = 16
DIG_MONITORING_ ENABLES	4	1 – Enable, 0 – Disabled **Bit** **Definition** 0 RESERVED 1 CR4 and VIM lockstep test 2 RESERVED 3 VIM test 4 RESERVED 5 RESERVED 6 CRC test

		7 RAMPGEN memory ECC 8 DFE Parity test 9 DFE memory ECC 10 RAMPGEN lockstep test 11 FRC lockstep test of diagnostic 12 RESERVED 13 RESERVED 14 RESERVED 15 RESERVED 16 ESM test 17 DFE STC 18 RESERVED 19 ATCM, BTCM ECC test 20 ATCM, BTCM parity test 21 RESERVED 22 RESERVED 23 RESERVED 24 FFT test 25 RTI test 26 PCR test 31:27 RESERVED
TEST_MODE	1	**Value** / **Definition** 0 – Production mode. Latent faults are tested and any failures are reported 1 – Characterization mode. Faults are injected and failures are reported which allows testing of the failure reporting path
RESERVED	3	0x000000
RESERVED	4	0x00000000

7.1.2 *Sub block 0x01C1 – AWR_MONITOR_RF_DIG_PERIODIC_CONF_SB*

This API SB contains the consolidated configuration of all periodic digital monitoring within radar sub-system. This is issued by the host to the AWR device.

The enabled monitoring functions are executed periodically and reports are sent based on reporting mode. Report of these monitoring will be available in the async event AWR_MONITOR_RF_DIG_PERIODIC_REPORT_AE_SB.

Table 7.2 – AWR_MONITOR_RF_DIG_PERIODIC_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x1C1
SBLKLEN	2	Value = 16
REPORTING_MODE	1	**Value** / **Definition** 0 – Report is sent every monitoring period without any threshold check 1 – Report is sent only on a failure 2 – Report is sent every monitoring period with threshold check
RESERVED	3	0x000000
PERIODIC_DIG_ MON_ EN	4	1 – Enable, 0 – Disable **Bit** **Monitoring type** 0 PERIODIC_CONFG_REGISTER_READ_EN 1 ESM_MONITORING_EN 2 DFE_STC_EN 3 FRAME_TIMING_MONITORING_EN 31:4 RESERVED
RESERVED	4	0x00000000

7.1.3 *Sub block 0x01C2 – AWR_MONITOR_ANALOG_ENABLES_CONF_SB*

This API SB contains the consolidated configuration of all analog monitoring. This is issued by the host to the AWR device.

The enabled monitoring functions are executed with a periodicity of CAL_MON_TIME_UNITS number of logical frames. The host should ensure that all the enabled monitors can be completed in the available inter-frame times, based on the monitoring durations (to be provided separately).

Table 7.3 – AWR_MONITOR_ANALOG_ENABLES_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x01C2
SBLKLEN	2	Value = 12
ANA_MONITORING_ENABLES	4	If any bit in this field is set to 1, the associate monitors are enabled. The configurations and reports of each monitors are described in respective sub sections. **Bit** **Definition** 0 TEMPERATURE_MONITOR 1 RX_GAIN_PHASE_MONITOR 2 RX_NOISE_FIGURE_MONITOR 3 RX_IFSTAGE_MONITOR 4 TX0_POWER_MONITOR 5 TX1_POWER_MONITOR 6 TX2_POWER_MONITOR 7 TX0_BALLBREAK_MONITOR 8 TX1_BALLBREAK_MONITOR 9 TX2_BALLBREAK_MONITOR 10 TX_GAIN_PHASE_MISMATCH_MONITOR 11 TX0_BPM_MONITOR 12 TX1_BPM_MONITOR 13 TX2_BPM_MONITOR 14 SYNTH_FREQ_MONITOR 15 EXTERNAL_ANALOG_SIGNALS_MONITOR 16 INTERNAL_TX0_SIGNALS_MONITOR 17 INTERNAL_TX1_SIGNALS_MONITOR 18 INTERNAL_TX2_SIGNALS_MONITOR 19 INTERNAL_RX_SIGNALS_MONITOR 20 INTERNAL_PMCLKLO_SIGNALS_MONITOR 21 INTERNAL_GPADC_SIGNALS_MONITOR 22 PLL_CONTROL_VOLTAGE_MONITOR 23 DCC_CLOCK_FREQ_MONITOR 24 RX_SATURATION_DETECTOR_MONITOR 25 RX_SIG_IMG_BAND_MONITOR 26 RX_MIXER_INPUT_POWER_MONITOR 31:27 RESERVED

RESERVED	4	0x00000000

7.2 Temperature Monitor

This section contains API SBs that configure the on chip temperature monitors and report the soft results from the monitor. The corresponding monitors are collectively named TEMPERATURE_MONITOR. These monitors observe the temperature near various RF analog and digital modules using temperature sensors and GPADC and compare them against configurable thresholds. The report is sent as an async event AWR_MONITOR_TEMPERATURE_REPORT_AE_SB.

7.2.1 Sub block 0x01C3 – AWR_MONITOR_TEMPERATURE_CONF_SB

This API is a monitoring configuration API which the host sends to the AWR device, containing information related to temperature monitoring. Report of this monitoring will be available in the async event AWR_MONITOR_TEMPERATURE_REPORT_AE_SB.

Table 7.4 – AWR_MONITOR_TEMPERATURE_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01C3
SBLKLEN	2	Value = 24
REPORTING_MODE	1	**Value** / **Definition** 0 – Report is sent every monitoring period without threshold check 1 – Report is send only upon a failure (after checking for thresholds) 2 – Report is sent every monitoring period with threshold check
RESERVED	1	0x00

ANA_TEMP_ THRESH_MIN	2	The temperatures read from near the sensors near the RF analog modules are compared against a minimum threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). 1 LSB = 1 degree Celsius, signed number Valid range: TBD Recommended value = TBD
ANA_TEMP_ THRESH_MAX	2	The temperatures read from near the sensors near the RF analog modules are compared against a maximum threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). 1 LSB = 1 degree Celsius, signed number Valid range: TBD Recommended value = TBD
DIG_TEMP_ THRESH_MIN	2	The temperatures read from near the sensor near the digital module are compared against a minimum threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). 1 LSB = 1 degree Celsius, signed number Valid range: TBD Recommended value = TBD
DIG_TEMP_ THRESH_MAX	2	The temperatures read from near the sensor near the digital module are compared against a maximum threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). 1 LSB = 1 degree Celsius, signed number Valid range: TBD Recommended value = TBD
TEMP_DIFF_ THRESH	2	The maximum difference across temperatures read from all the enabled sensors is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if the measured difference exceeds this field). 1 LSB = 1^o Celsius, signed number Valid range: TBD Recommended value = TBD

RESERVED	4	0x00000000
RESERVED	4	0x00000000

7.3 RX Gain and Phase Monitor

This section contains API SBs that configure the monitors of receiver gain and phase. The corresponding monitors are collectively named RX_GAIN_PHASE_MONITOR. The report is sent as an async event AWR_MONITOR_RX_GAIN_PHASE_REPORT_AE_SB.

7.3.1 Sub block 0x01C4 – AWR_MONITOR_RX_GAIN_PHASE_CONF_SB

This is a monitoring configuration API which the host sends to the AWR device, containing information related to RX gain and phase monitoring.

Table 7.5 – AWR_MONITOR_RX_GAIN_PHASE_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01C4
SBLKLEN	2	Value = 72
PROFILE_INDX	1	This field indicates the profile Index for which this monitoring configuration applies.
RF_FREQ_BITMASK	1	This field indicates the RF frequencies inside the profile's RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled w.r.t. the profile's RF band. Bit number | RF frequency | RF name 0 | Lowest RF frequency in profile's sweep bandwidth | RF1 1 | Center RF frequency in profile's sweep bandwidth | RF2 2 | Highest RF frequency in profile's sweep bandwidth | RF3 The RF name column is mentioned here to set the convention for the purpose of reporting and describing many monitoring packets.

REPORTING_MODE	1	**Value** – **Definition** 0 – Report is sent every monitoring period without threshold check 1 – Report is send only upon a failure (after checking for thresholds) 2 – Report is sent every monitoring period with threshold check
TX_SEL	1	**Value** – **Definition** 0 – TX0 is used for generating loopback signal for RX gain measurement 1 – TX1 is used for generating loopback signal for RX gain measurement
RX_GAIN_ABS_ ERROR_THRESH	2	The magnitude of difference between the programmed and measured RX gain for each enabled channel at each enabled RF frequency, is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting the offset given in the RX_GAIN_MISMATCH_OFFSET_VALUE field 1 LSB = 0.1 dB Valid range: TBD to 60 Recommended value = TBD
RX_GAIN_ MISMATCH_ THRESH	2	The magnitude of difference between measured RX gains across the enabled channels at each enabled RF frequency is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if the measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting the offset given in the RX_GAIN_MISMATCH_OFFSET_VALUE field. 1 LSB = 0.1 dB Valid range: TBD to 60 Recommended value = TBD

RX_GAIN_ FLATNESS_ ERROR_THRESH	2	The magnitude of measured RX gain flatness error, for each enabled channel, is compared against this threshold. The flatness error for a channel is defined as the peak to peak variation across RF frequencies. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting the offset given in the RX_GAIN_MISMATCH_OFFSET_VALUE field. 1 LSB = 0.1 dB Valid range: TBD to 60 Recommended value = TBD This flatness check is applicable only if multiple RF Frequencies are enabled, i.e., RF_FREQ_BITMASK has bit numbers 0,1,2 set.
RX_PHASE_ MISMATCH_ THRESH	2	The magnitude of measured RX phase mismatch across the enabled channels at each enabled RF frequency is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). Before the comparison, the measured phases for each RF and RX are adjusted by subtracting the offset given in the RX_PHASE_MISMATCH_OFFSET_VALUE field. 1 LSB = $360^{\circ} / 2^{16}$. Valid range: corresponding to 2° (TBD) to 20°. Recommended value = TBD
RX_GAIN_ MISMATCH_ OFFSET_VALUE	24	The offsets to be subtracted from the measured RX gain for each RX and RF before the relevant threshold comparisons are given here. Byte numbers corresponding to different RX and RF, in this field are here:

	RF1	RF2	RF3
RX0	1:0	9:8	17:16
RX1	3:2	11:10	19:18
RX2	5:4	13:12	21:20
RX3	7:6	15:14	23:22

1 LSB = 0.1 dB, signed number

Only the entries of enabled RF Frequencies and enabled RX channels are considered.

RX_PHASE_ MISMATCH_ OFFSET_VALUE	24	The offsets to be subtracted from the measured RX phase for each RX and RF before the relevant threshold comparisons are given here. Byte numbers corresponding to different RX and RF, in this field are here: | | RF1 | RF2 | RF3 | | RX0 | 1:0 | 9:8 | 17:16 | | RX1 | 3:2 | 11:10 | 19:18 | | RX2 | 5:4 | 13:12 | 21:20 | | RX3 | 7:6 | 15:14 | 23:22 | 1 LSB = $360^\circ / 2^{16}$, unsigned number Only the entries of enabled RF Frequencies and enabled RX channels are considered.
RESERVED	4	0x00000000
RESERVED	4	0x00000000

7.4 RX Noise Monitor

This section contains API SBs that configure the monitor of receiver noise, and report the soft results from the monitor. The corresponding monitor is named RX_NOISE_FIGURE_MONITOR. The report is sent as an async event AWR_MONITOR_RX_NOISE_FIGURE_REPORT_AE_SB.

7.4.1 Sub block 0x01C5 – AWR_MONITOR_RX_NOISE_FIGURE_CONF_SB

This is a monitoring configuration API which the host sends to the AWR device, containing information related to RX noise monitoring of a profile.

Table 7.6 – AWR_MONITOR_RX_NOISE_FIGURE_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01C5
SBLKLEN	2	Value = 16
PROFILE_INDX	1	This field indicates the profile Index for which this monitoring configuration applies.

RF_FREQ_BITMASK	1	This field indicates the exact RF frequencies inside the profile's RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled w.r.t. the profile's RF band.

Bit number	RF frequency	RF name
0	Lowest RF frequency in profile's sweep bandwidth	RF1
1	Center RF frequency in profile's sweep bandwidth	RF2
2	Highest RF frequency in profile's sweep bandwidth	RF3

The RF name column is mentioned here to set the convention for the purpose of reporting and describing many monitoring packets.

RESERVED	2	0x0000
REPORTING_MODE	1	

Value	**Definition**
0	Report is sent every monitoring period without threshold check
1	Report is send only upon a failure (after checking for thresholds)
2	Report is sent every monitoring period with threshold check

RESERVED	1	0x00
RX_NOISE_ FIGURE_ THRESHOLD	2	The measured RX input referred noise figure at the enabled RF frequencies, for all channels, is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). 1 LSB = 0.1 dB Valid range: TBD Recommended value = TBD
RESERVED	4	0x00000000

7.5 RX IF Stage Monitor

This section contains API SBs that configure the monitors of receiver IF filter attenuation, and report the soft results from the monitor. The corresponding monitor is named

RX_IFSTAGE_MONITOR. The report is sent as an async event AWR_MONITOR_RX_IFSTAGE_REPORT_AE_SB.

7.5.1 *Sub block 0x01C6 – AWR_MONITOR_RX_IFSTAGE_CONF_SB*

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to RX IF filter attenuation monitoring. The report is sent as as an async event AWR_MONITOR_RX_IFSTAGE_REPORT_AE_SB.

Table 7.7 – AWR_MONITOR_RX_IFSTAGE_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01C6
SBLKLEN	2	Value = 20
PROFILE_INDX	1	This field indicates the Profile Index for which this monitoring configuration applies.
REPORTING_MODE	1	Value: Definition 0: Report is sent every monitoring period without threshold check 1: Report is send only upon a failure (after checking for thresholds) 2: Report is sent every monitoring period with threshold check
RESERVED	2	0x0000
RESERVED	2	0x0000
HPF_CUTOFF_ FREQ_ERROR_ THRESH	2	The absolute values of RX IF HPF cutoff percentage frequency errors are compared against the corresponding thresholds given in this field. The comparison results are part of the monitoring report message (Error bit is set if the absolute value of the errors exceeds respective thresholds). 1 LSB = 1%, unsigned number Valid range: TBD to 255 Recommended value = TBD

LPF_CUTOFF_ FREQ_ERROR_ THRESH	2	The absolute values of RX IF LPF cutoff percentage frequency errors are compared against the corresponding thresholds given in this field. The comparison results are part of the monitoring report message (Error bit is set if the absolute value of the errors exceeds respective thresholds). 1 LSB = 1%, unsigned number Valid range: TBD to 255 Recommended value = TBD
IFA_GAIN_ ERROR_THRESH	2	The absolute deviation of RX IFA Gain from the expected gain for each enabled RX channel is compared against the thresholds given in this field. The comparison result is part of the monitoring report message (Error bit is set if the absolute value of the errors exceeds respective thresholds). 1 LSB = 0.1dB, unsigned number Valid range: TBD to 255 Recommended value = TBD
RESERVED	4	0x00000000

7.6 TX Power Monitor

This section contains API SBs that configure the monitors of transmitter output power, and report the soft results from the monitor. The corresponding monitors are collectively named TXn_POWER_MONITOR where n is the TX channel number.

7.6.1 Sub block 0x01C7 – AWR_MONITOR_TX0_POWER_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX0 power monitoring. Absolute TX power and flatness across RF frequencies are monitored here. The report is sent as an async event AWR_MONITOR_TX0_POWER_REPORT_AE_SB.

Table 7.8 – AWR_MONITOR_TX0_POWER_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01C7
SBLKLEN	2	Value = 20
PROFILE_INDX	1	This field indicates the Profile Index for which this monitoring configuration applies.

RF_FREQ_BITMASK	1	This field indicates the exact RF frequencies inside the profile's RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled w.r.t. the profile's RF band. Bit number | RF frequency | RF name 0 | Lowest RF frequency in profile's sweep bandwidth | RF1 1 | Center RF frequency in profile's sweep bandwidth | RF2 2 | Highest RF frequency in profile's sweep bandwidth | RF3 The RF Name column is mentioned here to set the convention for the purpose of reporting and describing many monitoring packets.
RESERVED	2	0x0000
REPORTING_MODE	1	**Value** | **Definition** 0 | Report is sent every monitoring period without threshold check 1 | Report is send only upon a failure (after checking for thresholds) 2 | Report is sent every monitoring period with threshold check
RESERVED	1	0x00
TX_POWER_ ABSOLUTE_ ERROR_THRESH	2	The magnitude of difference between the programmed and measured TX power for each enabled channel at each enabled RF frequency, is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). 1 LSB = 0.1 dBm Valid range: TBD to 60 Recommended value = TBD

TX_POWER_ FLATNESS_ ERROR_THRESH	2	The magnitude of measured TX power flatness error, for each enabled channel, is compared against this threshold. The flatness error for a channel is defined as the peak to peak variation across RF frequencies. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). 1 LSB = 0.1 dB Valid range: TBD to 60 Recommended value = TBD This flatness check is applicable only if multiple RF Frequencies are enabled.
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.6.2 Sub block 0x01C8 – AWR_MONITOR_TX1_POWER_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX1 power monitoring. Absolute TX power and flatness across RF frequencies are monitored here. The report is sent as an async event AWR_MONITOR_TX1_POWER_REPORT_AE_SB.

Table 7.9 – AWR_MONITOR_TX1_POWER_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01C8
SBLKLEN	2	Value = 20
PROFILE_INDX	1	This field indicates the Profile Index for which this monitoring configuration applies.

RF_FREQ_BITMASK	1	This field indicates the exact RF frequencies inside the profile's RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled w.r.t. the profile's RF band. Bit number / RF frequency / RF name; 0 / Lowest RF frequency in profile's sweep bandwidth / RF1; 1 / Center RF frequency in profile's sweep bandwidth / RF2; 2 / Highest RF frequency in profile's sweep bandwidth / RF3. The RF Name column is mentioned here to set the convention for the purpose of reporting and describing many monitoring packets.
REPORTING_MODE	1	**Value** / **Definition**; 0 / Report is sent every monitoring period without threshold check; 1 / Report is send only upon a failure (after checking for thresholds); 2 / Report is sent every monitoring period with threshold check
RESERVED	1	0x00
TX_POWER_ ABSOLUTE_ ERROR_THRESH	2	The magnitude of difference between the programmed and measured TX power for each enabled channel at each enabled RF frequency, is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). 1 LSB = 0.1 dBm. Valid range: TBD to 60. Recommended value = TBD

TX_POWER_ FLATNESS_ ERROR_THRESH	2	The magnitude of measured TX power flatness error, for each enabled channel, is compared against this threshold. The flatness error for a channel is defined as the peak to peak variation across RF frequencies. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). 1 LSB = 0.1 dB Valid range: TBD to 60 Recommended value = TBD This flatness check is applicable only if multiple RF Frequencies are enabled.
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.6.3 *Sub block 0x01C9 – AWR_MONITOR_TX2_POWER_CONF_SB*

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX2 power monitoring. Absolute TX power and flatness across RF frequencies are monitored here. The report is sent as an async event AWR_MONITOR_TX2_POWER_REPORT_AE_SB.

Table 7.10 – AWR_MONITOR_TX2_POWER_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01C9
SBLKLEN	2	Value = 20
PROFILE_INDX	1	This field indicates the Profile Index for which this monitoring configuration applies.

RF_FREQ_BITMASK	1	This field indicates the exact RF frequencies inside the profile's RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled w.r.t. the profile's RF band. Bit number: 0 – Lowest RF frequency in profile's sweep bandwidth – RF1 Bit number: 1 – Center RF frequency in profile's sweep bandwidth – RF2 Bit number: 2 – Highest RF frequency in profile's sweep bandwidth – RF3 The RF Name column is mentioned here to set the convention for the purpose of reporting and describing many monitoring packets.
REPORTING_MODE	1	Value 0: Report is sent every monitoring period without threshold check Value 1: Report is send only upon a failure (after checking for thresholds) Value 2: Report is sent every monitoring period with threshold check
RESERVED	1	0x00
TX_POWER_ ABSOLUTE_ ERROR_THRESH	2	The magnitude of difference between the programmed and measured TX power for each enabled channel at each enabled RF frequency, is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). 1 LSB = 0.1 dBm Valid range: TBD to 60 Recommended value = TBD

Nested table in RF_FREQ_BITMASK:

Bit number	RF frequency	RF name
0	Lowest RF frequency in profile's sweep bandwidth	RF1
1	Center RF frequency in profile's sweep bandwidth	RF2
2	Highest RF frequency in profile's sweep bandwidth	RF3

Nested table in REPORTING_MODE:

Value	**Definition**
0	Report is sent every monitoring period without threshold check
1	Report is send only upon a failure (after checking for thresholds)
2	Report is sent every monitoring period with threshold check

TX_POWER_ FLATNESS_ ERROR_THRESH	2	The magnitude of measured TX power flatness error, for each enabled channel, is compared against this threshold. The flatness error for a channel is defined as the peak to peak variation across RF frequencies. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). 1 LSB = 0.1 dB Valid range: TBD to 60 Recommended value = TBD This flatness check is applicable only if multiple RF Frequencies are enabled.
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.7 TX Ball Break Monitor

This section contains API SBs that configure the monitors of transmitter balls and impedance matching. The corresponding monitors are collectively named TXn_BALLBREAK_MONITOR where n is the TX channel number.

TX ball break detection is performed through measurement of TX reflection coefficient's magnitude. The breakage of a TX ball is detected by observing high reflection magnitude.

7.7.1 Sub block 0x01CA – AWR_MONITOR_TX0_BALLBREAK_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX ball break detection.

This API SB controls the thresholds for the reflection coefficient magnitude check and the parameters for the reflection coefficient error distance check (variation from values at the time of factory calibration). The report is sent as an async event AWR_MONITOR_TX0_BALLBREAK_REPORT_AE_SB.

Table 7.11 – AWR_MONITOR_TX0_BALLBREAK_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01CA
SBLKLEN	2	Value = 16

REPORTING_MODE	1	**Value**	**Definition**
		0	Report is sent every monitoring period without threshold check
		1	Report is send only upon a failure (after checking for thresholds)
		2	Report is sent every monitoring period with threshold check
RESERVED	1	0x00	
TX_REFL_COEFF_THRESH	2	The TX reflection coefficient's magnitude for each enabled channel is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is higher than this threshold, with the units of both quantities being the same). 1 LSB = 0.1 dB, signed number Valid range: -90 (TBD) to -250 Recommended value = TBD	
RESERVED	4	0x00000000	
RESERVED	4	0x00000000	

7.7.2 *Sub block 0x01CB – AWR_MONITOR_TX1_BALLBREAK_CONF_SB*

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX ball break detection.

This API SB controls the thresholds for the reflection coefficient magnitude check and the parameters for the reflection coefficient error distance check (variation from values at the time of factory calibration). The report is sent as an async event AWR_MONITOR_TX1_BALLBREAK_REPORT_AE_SB.

Table 7.12 – AWR_MONITOR_TX1_BALLBREAK_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01CB
SBLKLEN	2	Value = 16

REPORTING_MODE	1	<table><tr><th>Value</th><th>Definition</th></tr><tr><td>0</td><td>Report is sent every monitoring period without threshold check</td></tr><tr><td>1</td><td>Report is send only upon a failure (after checking for thresholds)</td></tr><tr><td>2</td><td>Report is sent every monitoring period with threshold check</td></tr></table>
RESERVED	1	0x00
TX_REFL_COEFF_ THRESH	2	The TX reflection coefficient's magnitude for each enabled channel is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is higher than this threshold, with the units of both quantities being the same). 1 LSB = 0.1 dB Valid range: -90 (TBD) to -250 Recommended value = TBD
RESERVED	4	0x00000000
RESERVED	4	0x00000000

7.7.3 Sub block 0x01CC – AWR_MONITOR_TX2_BALLBREAK_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX ball break detection.

This API SB controls the thresholds for the reflection coefficient magnitude check and the parameters for the reflection coefficient error distance check (variation from values at the time of factory calibration). The report is sent as an async event AWR_MONITOR_TX2_BALLBREAK_REPORT_AE_SB.

Table 7.13 – AWR_MONITOR_TX2_BALLBREAK_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x01CC
SBLKLEN	2	Value = 16

REPORTING_MODE	1	<table><tr><th>Value</th><th>Definition</th></tr><tr><td>0</td><td>Report is sent every monitoring period without threshold check</td></tr><tr><td>1</td><td>Report is send only upon a failure (after checking for thresholds)</td></tr><tr><td>2</td><td>Report is sent every monitoring period with threshold check</td></tr></table>
RESERVED	1	0x00
TX_REFL_COEFF_ THRESH	2	The TX reflection coefficient's magnitude for each enabled channel is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is higher than this threshold, with the units of both quantities being the same). 1 LSB = 0.1 dB Valid range: -90 (TBD) to -250 Recommended value = TBD
RESERVED	4	0x00000000
RESERVED	4	0x00000000

7.8 TX Gain and Phase Mismatch Monitoring

This section contains API SBs that configure the monitors of transmitter gain and phase mismatches, and report the soft results from the monitor. The corresponding monitors are collectively named TX_GAIN_PHASE_MISMATCH_MONITOR.

This monitor needs the operation of at least one RX channel. It also needs to use the RX in complex mode. Therefore, if all channels are disabled as per AWR_CHAN_CONF_SET_SB, this monitor automatically enables one RX channel. Further, this monitor automatically uses both I and Q channels of the receiver, irrespective of the ADC settings given by AWR_ADCOUT_CONF_SET_SB.

7.8.1 Sub block 0x01CD – AWR_MONITOR_TX_GAIN_PHASE_MISMATCH_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX gain and phase mismatch monitoring. The report is sent as an async event AWR_MONITOR_TX_GAIN_PHASE_REPORT_AE_SB.

Table 7.14 – AWR_MONITOR_TX_GAIN_PHASE_MISMATCH_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01CD
SBLKLEN	2	Value = 56
PROFILE_INDX	1	This field indicates the Profile Index for which this monitoring configuration applies. The TX settings corresponding to this profile index are used during the monitoring. The RX gain used in this measurement may differ from the given profile's RX gain.
RF_FREQ_BITMASK	1	This field indicates the exact RF frequencies inside the profile's RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled wrt the profile's RF band. Bit number | RF frequency | RF name 0 | Lowest RF frequency in profile's sweep bandwidth | RF1 1 | Center RF frequency in profile's sweep bandwidth | RF2 2 | Highest RF frequency in profile's sweep bandwidth | RF3 The RF Name column is mentioned here to set the convention for the purpose of reporting and describing many monitoring packets.
TX_EN	1	This field indicates the TX channels that should be compared for gain and phase balance. Setting the corresponding bit to 1 enables that channel for imbalance measurement. Bit number | TX Channel 0 | TX0 1 | TX1 2 | TX2

RX_EN	1	This field indicates the RX channels that should be enabled for TX to RX loopback measurement. Setting the corresponding bit to 1 enables that channel for imbalance measurement.

Bit number	RX Channel
0	RX0
1	RX1
2	RX2
3	RX3

REPORTING_MODE	1	

Value	**Definition**
0	Report is sent every monitoring period without threshold check
1	Report is send only upon a failure (after checking for thresholds)
2	Report is sent every monitoring period with threshold check

RESERVED	1	0x00
TX_GAIN_ MISMATCH_ THRESH	2	The magnitude of difference between measured TX powers across the enabled channels at each enabled RF frequency is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if the measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting the offset given in the TX_GAIN_MISMATCH_OFFSET_VALUE field. 1 LSB = 0.1dB, signed number Valid range: TBD to 60 Recommended value = TBD

TX_PHASE_ MISMATCH_ THRESH	2	The magnitude of measured TX phase mismatch across the enabled channels at each enabled RF frequency is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting the offset given in the TX_PHASE_MISMATCH_OFFSET_VALUE field. 1 LSB = $360^{\circ} / 2^{16}$, unsigned number Valid range: corresponding to 2^{o} (TBD) to 20^{o}. Recommended value = TBD
TX_GAIN_ MISMATCH_ OFFSET_VALUE	18	The offsets to be subtracted from the measured TX gain for each TX and RF before the relevant threshold comparisons are given here. Byte numbers corresponding to different RX and RF, in this field are here: (see table below) 1 LSB = 0.1 dB, signed number Only the entries of enabled RF Frequencies and enabled TX channels are considered.
TX_PHASE_ MISMATCH_ OFFSET_VALUE	18	The offsets to be subtracted from the measured TX phase for each TX and RF before the relevant threshold comparisons are given here. Byte numbers corresponding to different RX and RF, in this field are here: (see table below) 1 LSB = $360°/2^{16}$. Only the entries of enabled RF Frequencies and enabled TX channels are considered.

TX_GAIN_MISMATCH_OFFSET_VALUE byte numbers:

	RF1	RF2	RF3
TX0	1:0	7:6	11:12
TX1	3:2	9:8	15:14
TX2	5:4	11:10	17:16

TX_PHASE_MISMATCH_OFFSET_VALUE byte numbers:

	RF1	RF2	RF3
TX0	1:0	7:6	11:12
TX1	3:2	9:8	15:14
TX2	5:4	11:10	17:16

RESERVED	2	0x0000
RESERVED	4	0x00000000

7.9 TX BPM Phase Monitor

This section contains API SBs that configure the monitors of transmitter binary phase modulation, and report the soft results from the monitor, for various TX channels. The corresponding monitors are collectively named TX0_BPM_MONITOR, TX1_ BPM_MONITOR and TX2_BPM_MONITOR for the respective TX channels.

7.9.1 *Sub block 0x01CE – AWR_MONITOR_TX0_BPM_CONF_SB*

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX0 BPM monitoring.

The report is sent as an async event AWR_MONITOR_TX0_BPM_REPORT_AE_SB.

Table 7.15 – AWR_MONITOR_TX0_BPM_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01CE
SBLKLEN	2	Value = 20
PROFILE_INDX	1	This field indicates the Profile Index for which this monitoring configuration applies.
RESERVED	3	0x000000
REPORTING_MODE	1	**Value** / **Definition**: 0 – Report is sent every monitoring period without threshold check; 1 – Report is send only upon a failure (after checking for thresholds); 2 – Report is sent every monitoring period with threshold check

RX_EN	1	This field indicates the RX channels that should be enabled for TX to RX loopback measurement. Setting the corresponding bit to 1 enables that channel for imbalance measurement. Bit number / RX Channel: 0 – RX0 1 – RX1 2 – RX2 3 – RX3
TX_BPM_PHASE_ ERROR_THRESH	2	The deviation of the TX output phase difference between the two BPM settings from the ideal 180° is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is higher than this threshold, with the units of both quantities being the same). 1 LSB = $360°/2^{16}$. Valid range: TBD Recommended value = TBD
TX_BPM_ AMPLITUDE_ ERROR_THRESH	2	The deviation of the TX output amplitude difference between the two BPM settings is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is higher than this threshold, with the units of both quantities being the same). 1 LSB = 0.1 dB Valid range: TBD Recommended value = TBD
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.9.2 Sub block 0x01CF – AWR_MONITOR_TX1_BPM_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX1 BPM monitoring. The report is sent as an async event AWR_MONITOR_TX1_BPM_REPORT_AE_SB.

Table 7.16 – AWR_MONITOR_TX1_BPM_CONF_SB contents

Field Name	**Number of bytes**	**Description**

SBLKID	2	Value = 0x01CF
SBLKLEN	2	Value = 20
PROFILE_INDX	1	This field indicates the Profile Index for which this monitoring configuration applies.
RESERVED	3	0x000000
REPORTING_MODE	1	**Value** / **Definition**: 0 – Report is sent every monitoring period without threshold check; 1 – Report is send only upon a failure (after checking for thresholds); 2 – Report is sent every monitoring period with threshold check
RX_EN	1	This field indicates the RX channels that should be enabled for TX to RX loopback measurement. Setting the corresponding bit to 1 enables that channel for imbalance measurement. Bit number / RX Channel: 0 – RX0; 1 – RX1; 2 – RX2; 3 – RX3
TX_BPM_PHASE_ ERROR_THRESH	2	The deviation of the TX output amplitude difference between the two BPM settings is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is higher than this threshold, with the units of both quantities being the same). 1 LSB = 0.1 dB Valid range: TBD Recommended value = TBD

REPORTING_MODE values:

Value	Definition
0	Report is sent every monitoring period without threshold check
1	Report is send only upon a failure (after checking for thresholds)
2	Report is sent every monitoring period with threshold check

RX_EN bits:

Bit number	RX Channel
0	RX0
1	RX1
2	RX2
3	RX3

TX_BPM_ AMPLITUDE_ ERROR_THRESH	2	The deviation of the TX output phase difference between the two BPM settings from the ideal 180° is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is higher than this threshold, with the units of both quantities being the same). 1 LSB = 360°/2^{16}. Valid range: TBD Recommended value = TBD
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.9.3 Sub block 0x01D0 – AWR_MONITOR_TX2_BPM_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX2 BPM monitoring. The report is sent as an async event AWR_MONITOR_TX2_BPM_REPORT_AE_SB.

Table 7.17 – AWR_MONITOR_TX2_BPM_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01D0
SBLKLEN	2	Value = 20
PROFILE_INDX	1	This field indicates the Profile Index for which this monitoring configuration applies.
RESERVED	3	0x000000
REPORTING_MODE	1	**Value** / **Definition** 0 – Report is sent every monitoring period without threshold check 1 – Report is send only upon a failure (after checking for thresholds) 2 – Report is sent every monitoring period with threshold check

RX_EN	1	This field indicates the RX channels that should be enabled for TX to RX loopback measurement. Setting the corresponding bit to 1 enables that channel for imbalance measurement. Bit number / RX Channel: 0 – RX0 1 – RX1 2 – RX2 3 – RX3
TX_BPM_PHASE_ ERROR_THRESH	2	The deviation of the TX output phase difference between the two BPM settings from the ideal 180° is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is higher than this threshold, with the units of both quantities being the same). 1 LSB = 360°/2^{16}. Valid range: TBD Recommended value = TBD
TX_BPM_ AMPLITUDE_ ERROR_THRESH	2	The deviation of the TX output amplitude difference between the two BPM settings is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is higher than this threshold, with the units of both quantities being the same). 1 LSB = 0.1 dB Valid range: TBD Recommended value = TBD
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.10 Synthesizer Frequency Monitoring

This section contains API SBs that configure the monitors of synthesizer chirp frequency, and report the soft results from the monitor. The corresponding monitor is named SYNTH_FREQ_MONITOR.

7.10.1 Sub block 0x01D1 – AWR_MONITOR_SYNTHESIZER_FREQUENCY_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to synthesizer frequency monitoring during chirping. The report is sent as an async event AWR_MONITOR_SYNTH_FREQUENCY_REPORT_AE_SB.

Table 7.18 – AWR_MONITOR_SYNTHESIZER_FREQUENCY_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01D1
SBLKLEN	2	Value = 16
PROFILE_INDX	1	This field indicates the Profile Index for which this monitoring configuration applies.
REPORTING_MODE	1	**Value** | **Definition** 0 | Report is sent every monitoring period without threshold check 1 | Report is send only upon a failure (after checking for thresholds) 2 | Report is sent every monitoring period with threshold check
FREQ_ERROR_THRESH	2	During the chirp, the error of the measured instantaneous chirp frequency w.r.t. the desired value is continuously compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if the measurement is above this threshold, ever during the previous monitoring period). 1 LSB = 10 kHz Valid range: TBD Recommended value = TBD
MONITOR_START_TIME	1	This field determines when the monitoring starts in each chirp relative to the start of the ramp. 1 LSB = 0.2us, unsigned number Valid range: 0 to 25us
RESERVED	3	0x000000
RESERVED	4	0x00000000

7.11 External Analog Signals Monitor

This section contains API SBs that configure the monitors of external analog signals which are input to the device through pins ANALOGTEST1-4, ANAMUX and VSENSE (also called ADC1-6) and report the soft results from the monitor. The corresponding monitors are collectively named EXTERNAL_ANALOG_SIGNALS_MONITOR. These monitors observe various analog signals input on the pins ADC1-6 using a GPADC and compare them against internally fixed thresholds.

7.11.1 Sub block 0x01D2 – AWR_MONITORING_EXTERNAL_ANALOG_SIGNALS_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to external DC signals monitoring (available only in AWR1642). The report is sent as an async event AWR_MONITOR_EXTERNAL_ANALOG_SIGNALSREPORT_AE_SB.

Table 7.19 describes the content of this sub block.

Table 7.19 – AWR_MONITORING_EXTERNAL_ANALOG_SIGNALS_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01D2
SBLKLEN	2	Value = 36
REPORTING_MODE	1	**Value** / **Definition** 0: Report is sent every monitoring period without threshold check 1: Report is send only upon a failure (after checking for thresholds) 2: Report is sent every monitoring period with threshold check
RESERVED	1	0x00

SIGNAL_INPUT_ ENABLES	1	This field indicates the sets of externally fed DC signals which are to be monitored using GPADC. When each bit in this field is set, the corresponding signal is monitored. The monitored signals are compared against programmed limits. The comparison result is part of the monitoring report message. **Bit Location** | **SIGNAL** 0 | ANALOGTEST1 1 | ANALOGTEST2 2 | ANALOGTEST3 3 | ANALOGTEST4 4 | ANAMUX 5 | VSENSE Others | RESERVED
SIGNAL_BUFFER_ ENABLES	1	This field indicates the sets of externally fed DC signals which are to be buffered before being fed to the GPADC. When each bit in this field is set, the corresponding signal is buffered before the GPADC. The monitored signals are compared against programmed limits. The comparison result is part of the monitoring report message. **Bit** | **SIGNAL** 0 | ANALOGTEST1 1 | ANALOGTEST2 2 | ANALOGTEST3 3 | ANALOGTEST4 4 | ANAMUX Others | RESERVED

SIGNAL_SETTLING_TIME	6	After connecting an external signal to the GPADC, the amount of time to wait for it to settle before taking GPADC samples is programmed in this field. For each signal, after that settling time, GPADC measurements take place for 6.4us (averaging 4 samples of the GPADC output). The byte locations of the settling times for each signal are tabulated here: **Byte Location** | **SIGNAL** 0 | ANALOGTEST1 1 | ANALOGTEST2 2 | ANALOGTEST3 3 | ANALOGTEST4 4 | ANAMUX 5 | VSENSE 1 LSB = 0.8us Valid range: 0 to 12us Valid programming condition: all the signals that are enabled should take a total of <100us, including the programmed settling times and a fixed 6.4us of measurement time per enabled signal.

Byte Location	SIGNAL
0	ANALOGTEST1
1	ANALOGTEST2
2	ANALOGTEST3
3	ANALOGTEST4
4	ANAMUX
5	VSENSE

SIGNAL_THRESH	12	The external DC signals measured on GPADC are compared against these minimum and maximum thresholds. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). **Byte Location** | **Threshold** | **SIGNAL** 0 | Minimum | ANALOGTEST1 1 | Minimum | ANALOGTEST2 2 | Minimum | ANALOGTEST3 3 | Minimum | ANALOGTEST4 4 | Minimum | ANAMUX 5 | Minimum | VSENSE 6 | Maximum | ANALOGTEST1 7 | Maximum | ANALOGTEST2 8 | Maximum | ANALOGTEST3 9 | Maximum | ANALOGTEST4 10 | Maximum | ANAMUX 11 | Maximum | VSENSE 1 LSB = 1.8V / 256 Valid range: 0 to 255
RESERVED	2	0x0000
RESERVED	4	0x00000000
RESERVED	4	0x00000000

7.12 Internal Analog Signals Monitor

This section contains API SBs that configure the monitors of internal analog signals in the RF analog modules and report the soft results from the monitor. The corresponding monitors are collectively named INTERNAL_ANALOG_SIGNALS_MONITOR. These monitors observe various analog nodes in the RF and analog modules using a GPADC and compare them against internally fixed thresholds.

The configuration API SBs are organized to address various analog circuits as follows:

1. TX0 Internal Analog Signals Monitoring
    a. This monitor is called INTERNAL_TX0_SIGNALS_MONITOR.
    b. Signal sets that are monitored: (SUPPLY_TX, PWRDET_TX)
2. TX1 Internal Analog Signals Monitoring
    a. This monitor is called INTERNAL_TX1_SIGNALS_MONITOR
    b. Signal sets that are monitored: (SUPPLY_TX, PWRDET_TX)
3. TX2 Internal Analog Signals Monitoring
    a. This monitor is called INTERNAL_TX2_SIGNALS_MONITOR
    b. Signal sets that are monitored: (SUPPLY_TX, PWRDET_TX)
4. RX Internal Analog Signals Monitoring
    a. This monitor is called INTERNAL_RX_SIGNALS_MONITOR
    b. Signal sets that are monitored: (SUPPLY_RX, PWRDET_RX, DCBIAS_RX)
5. PM CLK LO Internal Analog Signals Monitoring
    a. This monitor is called INTERNAL_PMCLKLO_SIGNALS_MONITOR
    b. Signal sets that are monitored: (SUPPLY_PMCLKLO, PWRDET_PMCLKLO, DCBIAS_PMCLKLO)
6. GPADC Internal Analog Signals Monitoring
    a. This monitor is called INTERNAL_GPADC_SIGNALS_MONITOR
    b. Signal sets that are monitored: (GPADC_REF1, GPADC_REF2)

The results are reported in the corresponding REPORT API SBs in this section.

7.12.1 Sub block 0x01D3 – AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AR device, containing information related to TX0 Internal Analog Signals monitoring. The report is sent as an async event AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB.

Table 7.20 – AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01D3
SBLKLEN	2	Value = 12
PROFILE_INDEX	1	The RF analog settings corresponding to this profile are used for monitoring the enabled signals, using test chirps (static frequency, at the center of the profile’s RF frequency band).

REPORTING_MODE	1	**Value** / **Definition** 0 / RESERVED 1 / Report is send only upon a failure (after checking for thresholds) 2 / Report is sent every monitoring period with threshold check
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.12.2 Sub block 0x01D4 – AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX1 Internal Analog Signals monitoring. The report is sent as an async event AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB.

Table 7.21 – AWR_MONITOR_TX1_INTERNAL_ANALOG_SIGNALS_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01D4
SBLKLEN	2	Value = 12
PROFILE_INDEX	1	The RF analog settings corresponding to this profile are used for monitoring the enabled signals, using test chirps (static frequency, at the center of the profile's RF frequency band).
REPORTING_MODE	1	**Value** / **Definition** 0 / RESERVED 1 / Report is send only upon a failure (after checking for thresholds) 2 / Report is sent every monitoring period with threshold check
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.12.3 Sub block 0x01D5 – AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to TX2 Internal Analog Signals monitoring. The report is sent as an async event AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB.

Table 7.22 – AWR_MONITOR_TX2_INTERNAL_ANALOG_SIGNALS_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x01D5
SBLKLEN	2	Value = 16
PROFILE_INDEX	1	The RF analog settings corresponding to this profile are used for monitoring the enabled signals, using test chirps (static frequency, at the center of the profile's RF frequency band).
REPORTING_MODE	1	**Value** / **Definition**: 0 – RESERVED; 1 – Report is send only upon a failure (after checking for thresholds); 2 – Report is sent every monitoring period with threshold check
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.12.4 Sub block 0x01D6 – AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to RX Internal Analog Signals monitoring. The report is sent as an async event AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB.

Table 7.23 – AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x01D6
SBLKLEN	2	Value = 12

PROFILE_INDEX	1	The RF analog settings corresponding to this profile are used for monitoring the enabled signals, using test chirps (static frequency, at the center of the profile's RF frequency band).
REPORTING_MODE	1	**Value** / **Definition**: 0 – RESERVED; 1 – Report is send only upon a failure (after checking for thresholds); 2 – Report is sent every monitoring period with threshold check
RESERVED	2	0x0000
RESERVED	4	0x00000000

7.12.5 Sub block 0x01D7 – AWR_MONITOR_PMCLKLO_INTERNAL_ANALOG_SIGNALS_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to Power Management, Clock generation and LO distribution circuits' Internal Analog Signals monitoring. The report is sent as an async event AWR_MONITOR_PMCLKLO_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB.

Table 7.24 – AWR_MONITOR_PMCLKLO_INTERNAL_ANALOG_SIGNALS_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01D7
SBLKLEN	2	Value = 12
PROFILE_INDEX	1	The RF analog settings corresponding to this profile are used for monitoring the enabled signals, using test chirps (static frequency, at the center of the profile's RF frequency band).
REPORTING_MODE	1	**Value** / **Definition**: 0 – RESERVED; 1 – Report is send only upon a failure (after checking for thresholds); 2 – Report is sent every monitoring period with threshold check
RESERVED	2	0x0000

RESERVED	4	0x00000000

7.12.6 Sub block 0x01D8 – AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to GPADC Internal Analog Signals monitoring. During this monitor, only the relevant circuits are ensured to be ON. The monitored signals are compared against internally chosen valid limits. The comparison result is part of the consolidated monitoring report message (Error bit for any signal set is set to 1 if any measurement in that signal set is beyond valid limits). The report is sent as an async event AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_REPORT_AE_SB.

Table 7.25 – AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x01D8
SBLKLEN	2	Value = 12
REPORTING_MODE	1	<table><tr><th>Value</th><th>Definition</th></tr><tr><td>0</td><td>Report is sent every monitoring period without threshold check</td></tr><tr><td>1</td><td>Report is send only upon a failure (after checking for thresholds)</td></tr><tr><td>2</td><td>Report is sent every monitoring period with threshold check</td></tr></table>
RESERVED	3	0x000000
RESERVED	4	0x00000000

7.13 PLL Control Voltage Monitor

This section contains API SBs that configure the monitors of APLL and Synthesizer VCO control voltages and report the soft results from the monitor. The corresponding monitors are collectively named PLL_CONTROL_VOLTAGE_MONITOR. These monitors observe the VCO control voltages under various conditions using the GPADC and compare them against internally fixed thresholds. The transmitters are kept in OFF state during these measurements to avoid external emission.

7.13.1 Sub block 0x01D9 – AWR_MONITOR_PLL_CONTROL_VOLTAGE_SIGNALS_CONF_SB

This is a monitoring configuration API which the host sends to the AWR device, containing information related to APLL and Synthesizer's control voltage signals monitoring. The report is sent as an async event AWR_MONITOR_PLL_CONTROL_VOLTAGE_REPORT_AE_SB.

Table 7.26 – AWR_MONITOR_PLL_CONTROL_VOLTAGE_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01D9
SBLKLEN	2	Value = 12
REPORTING_MODE	1	**Value** **Definition** 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds) 2 Report is sent every monitoring period with threshold check
RESERVED	1	0x00

SIGNAL_ENABLES	2	This field indicates the sets of signals which are to be monitored. When each bit in this field is set, the corresponding signal set is monitored using test chirps. Rest of the RF analog may not be ON during these test chirps. The APLL VCO control voltage can be monitored. The Synthesizer VCO control voltage for both VCO1 and VCO2 can be monitored, while operating at their respective minimum and maximum frequencies, and their respective VCO slope (Hz/V) can be monitored if both frequencies are enabled for that VCO. The monitored signals are compared against internally chosen valid limits. The comparison results are part of the monitoring report message. **Bit Location** / **SIGNAL**: 0 – APLL_VCTRL 1 – SYNTH_VCO1_VCTRL 2 – SYNTH_VCO2_VCTRL 15:3 – RESERVED The synthesizer VCO extreme frequencies are: **Synthesizer VCO** / **Frequency Limits (Min, Max)**: VCO1 – (76GHz, 78GHz) VCO2 – (77GHz, 81GHz) Synthesizer measurements are done with TX switched off to avoid emissions. For the failure reporting, the internally chosen valid limits are (tentative): for the measured control voltage levels: 0.15V to 1.25V; for the synthesizer VCO slope: ±20% of 1.1GHz/V for VCO2 and 0.55GHz/V for VCO1.
RESERVED	4	0x00000000

Nested tables within SIGNAL_ENABLES:

Bit Location	SIGNAL
0	APLL_VCTRL
1	SYNTH_VCO1_VCTRL
2	SYNTH_VCO2_VCTRL
15:3	RESERVED

Synthesizer VCO	Frequency Limits (Min, Max)
VCO1	(76GHz, 78GHz)
VCO2	(77GHz, 81GHz)

7.14 Dual Clock Comparator Based Clock Frequency Monitor

This section contains API SBs that configure the Dual Clock Comparator based monitors of clocks in the BSS digital modules and report the soft results from the monitor. The corresponding monitors are collectively named DCC_CLOCK_FREQ_MONITOR. These monitors observe the relative frequency of various clock pairs and compare the measured relative frequency errors against internally fixed thresholds.

The various clock pairs that are monitored are defined here:

CLOCK PAIR	REFERENCE CLOCK	MEASURED CLOCK	ERROR THRESHOLD (Tentative)
0	XTAL	BSS_600M	±0.25%
1	BSS_600M	BSS_200M	±0.25%
2	BSS_600M	BSS_100M	±0.25%
3	BSS_600M	GPADC_10M	±2.5%
4	BSS_600M	RCOSC_10M	±17.5%
5	BSS_600M	RAMPGEN_100M	±0.25%
RSVD	RSVD	RSVD	RSVD

The ideal frequencies of clocks involved in this monitor are given here:

CLOCK NAME	CLOCK FREQUENCY (MHz)	COMMENTS
XTAL	40	Crystal clock
BSS_600M	600	BSS root clock
BSS_200M	200	BSS processor clock
BSS_100M	100	BSS internal clock
GPADC_10M	10	GPADC clock used in monitoring and calibrations
RCOSC_10M	10 (±10%)	RC Oscillator clock
RAMPGEN_100M	100	Clock for Ramp Generator (timing engine) and Digital Front End.

7.14.1 *Sub block 0x01DA – AWR_MONITOR_DUAL_CLOCK_COMP_CONF_SB*

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to the DCC based clock frequency monitoring. The report is sent as an async event AWR_MONITOR_DCC_DUAL_CLOCK_COMP_REPORT_AE_SB.

Table 7.27 – AWR_MONITOR_DUAL_CLOCK_COMP_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x01DA
SBLKLEN	2	Value = 12

REPORTING_MODE	1	<table><tr><th>Value</th><th>Definition</th></tr><tr><td>0</td><td>Report is sent every monitoring period without threshold check</td></tr><tr><td>1</td><td>Report is send only upon a failure (after checking for thresholds)</td></tr><tr><td>2</td><td>Report is sent every monitoring period with threshold check</td></tr></table>
RESERVED	1	0x00
DCC_PAIR_ ENABLES	2	This field indicates which pairs of clocks to monitor. When a bit in the field is set to 1, the firmware monitors the corresponding clock pair by deploying the hardware's Dual Clock Comparator in the corresponding DCC mode. <table><tr><th>Bit</th><th>CLOCK PAIR</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr><tr><td>2</td><td>2</td></tr><tr><td>3</td><td>3</td></tr><tr><td>4</td><td>4</td></tr><tr><td>5</td><td>5</td></tr><tr><td>15:6</td><td>RESERVED</td></tr></table> The comparison results are part of the monitoring report message. The definition of the clock pairs and their error thresholds for failure reporting are given in the table below the message definition.
RESERVED	4	0x00000000

7.15 RX Saturation Detection Monitor

This section contains API SBs that configure the monitoring of RX analog saturation detectors, and report the results from the monitor. The corresponding monitors are collectively named RX_SATURATION_DETECTOR_MONITOR and RX_SIG_IMG_BAND_MONITOR. The report is available in CQ RAM.

7.15.1 Sub block 0x01DB – AWR_MONITOR_RX_SATURATION_DETECTOR_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to RX saturation detector monitoring. The report is available as CQ2 (part of CQ) in CQ RAM every chirp. The application should transfer the report from CQ RAM every chirp.

Table 7.28 – AWR_MONITOR_RX_SATURATION_DETECTOR_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01DB
SBLKLEN	2	Value = 24
PROFILE_INDX	1	This field indicates the profile Index for which this monitoring configuration applies.
SAT_MON_SELECT	1	01 ⇒ Enable only the ADC saturation monitor 10 ⇒ Enable only the IFA1 saturation monitor 11 ⇒ Enable both the ADC and IFA1 saturation monitors
RESERVED	1	0x00
RESERVED	1	0x00
SAT_MON_PRIMARY_TIME_SLICE_DURATION	2	It specifies the duration of each (primary) time slice. 1 LSB = 0.16us. Valid range: 4 to floor(ADC sampling time us/0.16 us) NOTES: The minimum allowed duration of each (primary) time slice is 4 LSBs = 0.64us. Also, the maximum number of (primary) time slices that will be monitored in a chirp is 64 so the recommendation is to set this value to correspond to (ADC sampling time / 64). If the slice is smaller, such that the ADC sampling time is longer than 64 primary slices, some regions of the valid duration of a chirp may go un-monitored.
SAT_MON_NUM_SLICES	2	Number of (primary + secondary) time slices to monitor. Valid range: 1 to 127 NOTE: Together with SAT_MON_PRIMARY_TIME_SLICE_DURATION, this determines the full duration of the ADC valid time that gets covered by the monitor

SAT_MON_RX_ CHANNEL_MASK	1	Masks RX channels used for monitoring. In every slice, saturation counts for all unmasked channels are added together, and the total is capped to 127. The 8 bits are mapped (MSB->LSB) to: [RX3Q, RX2Q, RX1Q, RX0Q, RX3I, RX2I, RX1I, RX0I] 00000000 ⇒ All channels unmasked 11111111 ⇒ All channels masked
RESERVED	1	0
RESERVED	1	0
RESERVED	1	0
RESERVED	4	0x00000000
RESERVED	4	0x00000000

7.15.2 Sub block 0x01DC – AWR_MONITOR_SIG_IMG_MONITOR_CONF_SB

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to signal and image band energy. The report is available as CQ1 (part of CQ) in CQ RAM. The application should transfer the report every chirp.

Table 7.29 – AWR_MONITOR_SIG_IMG_MONITOR_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x01DC
SBLKLEN	2	Value = 16
PROFILE_INDX	1	This field indicates the profile index for which this monitoring configuration applies.
SIG_IMG_MON_NUM_ SLICES	1	Number of (primary + secondary) slices to monitor Valid range: 1 to 127

NUM_SAMPLES_PER_PRIMARY_TIME_SLICE	2	This field specifies the number of samples constituting each time slice. The minimum allowed value for this parameter is 4. Valid range: 4 to NUM_ADC_SAMPLES (see NOTE2 below) **NOTE1:** The maximum number of (primary) time slices that will be monitored in a chirp is 64, so our recommendation is that this value should at least equal (NUM_ADC_SAMPLES / 64). If the slice is smaller, such that the number of ADC samples per chirp is larger than 64 primary slices, some regions of the valid duration of a chirp may go un-monitored. **NOTE2:** In Complex1x mode, the minimum number of samples per slice is 4 and for other modes it is 8. Also note that number of samples should be an even number
RESERVED	4	0x00000000
RESERVED	4	0x00000000

7.16 RX mixer input power monitor

7.16.1 *Sub block 0x01DD – AWR_MONITOR_RX_MIXER_IN_POWER_CONF_SB*

This API is a Monitoring Configuration API which the host sends to the AWR device, containing information related to RX mixer input power monitoring. The report is sent as an async event AWR_MONITOR_RX_MIXER_IN_POWER_REPORT_AE_SB.

Table 7.30 – AWR_MONITOR_RX_MIXER_IN_POWER_CONF_SB contents

Field Name	Number of bytes	Description
SBLKID	2	Value = 0x1DD
SBLKLEN	2	Value = 16
REPORTING_MODE	1	**Value** / **Definition** 0: Report is sent every monitoring period without threshold check 1: Report is send only upon a failure (after checking for thresholds) 2: Report is sent every monitoring period with threshold check

PROFILE_INDEX	1	The RF analog settings corresponding to this profile are used for monitoring RX mixer input power using test chirps (static frequency, at the center of the profile's RF frequency band).
TX_EN	1	This field indicates if and which TX channels should be enabled while measuring RX mixer input power. Setting a bit to 1 enables the corresponding TX channel. Enabling a TX channel may help find reflection power while disabling may help find interference power. Bit number: TX Channel 0: TX0 1: TX1 2: TX2
RESERVED	1	0x00
THRESHOLDS	2	The measured RX mixer input voltage swings during this monitoring is compared against the minimum and maximum thresholds configured in this field. The comparison result is part of the monitoring report message (Status bit is cleared if any measurement is outside this (minimum, maximum) range). **Byte number**: **Threshold** 0: Minimum Threshold 1: Maximum Threshold Only the RX channels enabled in the static configuration APIs are monitored. 1 LSB = 1800mV/256, unsigned number Valid range: TBD, maximum threshold ≥ minimum threshold Recommended value = TBD
RESERVED	2	0x00000000
RESERVED	4	0x00000000

7.17 Sub block 0x01DE – RESERVED

7.18 Analog Fault injection

7.18.1 Sub block 0x1DF – AWR_ANALOG_FAULT_INJECTION_CONF_SB

This API is a Fault Injection API which the host sends to the AWR device. It can be used to inject faults in the analog circuits to test the corresponding monitors. After the faults are injected, the regular monitors, when enabled will indicate the faults in their associated reports.

NOTE 1: This API should be issued when no frames and on-going.

NOTE 2: The fault injection should be tested by injecting one fault at a time

Table 7.31 – AWR_ANALOG_FAULT_INJECTION_CONF_SB contents

Field Name	**Number of bytes**	**Description**
SBLKID	2	Value = 0x01DF
SBLKLEN	2	Value = 24
RESERVED	1	0x00
RX_GAIN_DROP	1	Primary Fault: RX Gain This field indicates which RX RF sections should have fault injected. If the fault is enabled, the RX RF gain drops significantly. The fault can be used to cause significant gain change, inter-RX gain imbalance and an uncontrolled amount of inter-RX phase imbalance. Bit number | RX Channel 0 | RX0 1 | RX1 2 | RX2 3 | RX3 Others | RESERVED For each bit, 1 = inject fault, 0 = remove injected fault

RX_PHASE_INV	1	Primary Fault: RX Phase This field indicates which RX channels should have fault injected. If the fault is enabled, the RX phase gets inverted. The fault can be used to cause a controlled amount (180o) of inter-RX phase imbalance. Bit number | RX Channel 0 | RX0 1 | RX1 2 | RX2 3 | RX3 Others | RESERVED For each bit, 1 = inject fault, 0 = remove injected fault
RX_HIGH_NOISE	1	Primary Fault: RX Noise This field indicates which RX channels should have fault injected. If the fault is enabled, the RX IFA square wave loopback paths are engaged to inject high noise at RX IFA input. The fault can be used to cause significant RX noise floor elevation. Bit number | RX Channel 0 | RX0 1 | RX1 2 | RX2 3 | RX3 Others | RESERVED For each bit, 1 = inject fault, 0 = remove injected fault

RX_IF_STAGES_ FAULT	1	Primary Fault: Cutoff frequencies of RX IFA HPF & LPF, IFA Gain This field indicates which RX channels should have fault injected. If the fault is enabled, the RX IFA HPF cutoff frequency becomes very high (about 15MHz). The fault can be used to cause the measured inband IFA gain, HPF and LPF attenuations to vary from ideal expectations. Bit number | RX Channel 0 | RX0 1 | RX1 2 | RX2 3 | RX3 Others | RESERVED For each bit, 1 = inject fault, 0 = remove injected fault **Note:** During the execution of RX_IFSTAGE_MONITOR, the RX_HIGH_NOISE faults are temporarily removed.
RX_LO_AMP_FAULT	1	Primary Fault: RX Mixer LO input swing reduction This field indicates which RX channels should have fault injected. If the fault is enabled, the RX mixer LO input swing is significantly reduced. The fault is primarily expected to be detected by RX_INTERNAL_ANALOG_SIGNALS_MONITOR (under PWRDET_RX category). Bit number | Channel 0 | RX0 and RX1 1 | RX2 and RX3 Others | RESERVED For each bit, 1 = inject fault, 0 = remove injected fault

TX_LO_AMP_FAULT	1	Primary Fault: TX PA input signal generator turning off. This field indicates which TX channels should have fault injected. If the fault is enabled, the amplifier generating TX power amplifier's LO input signal is turned off. The fault is primarily expected to be detected by TX<n>_INTERNAL_ANALOG_SIGNALS_MONITOR (under DCBIAS category). Bit number | Channel 0 | TX0 and TX1 1 | TX2 (applicable only if available in the device) Others | RESERVED For each bit, 1 = inject fault, 0 = remove injected fault
TX_GAIN_DROP	1	Primary Fault: TX Gain (power) This field indicates which TX RF sections should have fault injected. If the fault is enabled, the TX RF gain drops significantly. The fault can be used to cause significant TX output power change, inter-TX gain imbalance and an uncontrolled amount of inter-TX phase imbalance. Bit number | TX Channel 0 | TX0 1 | TX1 2 | TX2 Others | RESERVED For each bit, 1 = inject fault, 0 = remove injected fault

TX_PHASE_INV	1	Primary Fault: TX Phase This field indicates if TX channels should have fault injected, along with some further programmability. If the fault is enabled, the TX BPM polarity (phase) is forced to a constant value as programmed. The fault can be used to cause a controlled amount (180 degree) of inter-TX phase imbalance as well as BPM functionality failure.

Bit number	TX Channel
0	TX_FAULT (Common for all TX channels)
1	RESERVED
2	RESERVED
3	TX0_BPM_VALUE
4	TX1_BPM_VALUE
5	TX2_BPM_VALUE
Others	RESERVED

For each TXn_BPM_VALUE:

Applicable only if TX_FAULT is enabled.

Value = 0: force TX<n> BPM polarity to 0

Value = 1: force TX<n> BPM polarity to 1.

Note: The TXn_BPM_VALUE takes effect only when TX_FAULT value is changed

SYNTH_FAULT	1	Primary Fault: Synthesizer Frequency This field indicates which Synthesizer faults should be injected. SYNTH_VCO_OPENLOOP: If the fault is enabled, the synthesizer is forced in open loop mode with the VCO control voltage forced to a constant. In order to avoid out of band emissions in this faulty state, this fault is injected just before the PLL_CONTROL_VOLTAGE_MONITOR is executed and released just after its completion. SYNTH_FREQ_MON_OFFSET: If the fault is enabled, the synthesizer frequency monitor's ideal frequency ramp waveform is forced to be offset from the actual ramp waveform by a constant, causing monitoring to detect failures. Bit number | Enable Fault 0 | SYNTH_VCO_OPENLOOP 1 | SYNTH_FREQ_MON_OFFSET Others | RESERVED For each bit, 1 = inject fault, 0 = remove injected fault
SUPPLY_LDO_ FAULT	1	This field indicates whether some LDO output voltage faults should be injected or not. Bit number | Enable Fault 0 | SUPPLY_LDO_RX_LODIST_ FAULT Others | RESERVED SUPPLY_LDO_RX_LODIST_FAULT: if enabled, the RX LO distribution sub system's LDO output voltage is slightly changed compared to normal levels to cause INTERNAL_PMCLKLO_SIGNALS_MONITOR to detect failure (under SUPPLY category). For each bit, 1 = inject fault, 0 = remove injected fault

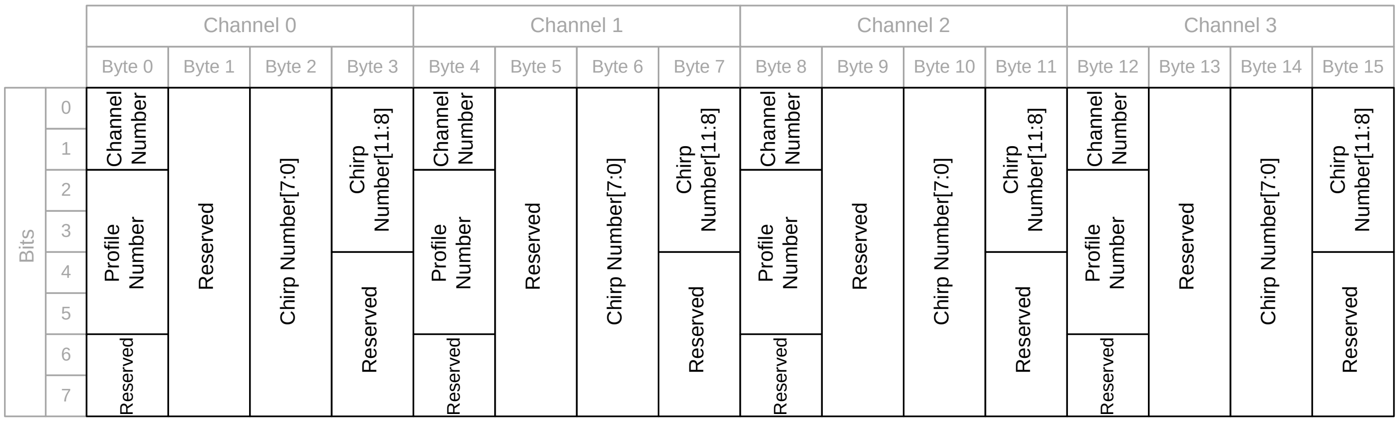
MISC_FAULT	1	This field indicates whether a few miscellaneous faults should be injected or not. Bit number | Enable Fault 0 | GPADC_CLK_FREQ_FAULT Others | RESERVED GPADC_CLK_FREQ_FAULT: if enabled, the GPADC clock frequency is slightly increased compared to normal usage to cause BSS DCC_CLOCK_FREQ_MONITOR to detect failure. For each bit, 1 = inject fault, 0 = remove injected fault
MISC_THRESH_ FAULT	1	This field indicates whether faults should be forced in the threshold comparisons in the software layer of some monitors. If a fault is enabled, the logic in the min-max threshold comparisons used for failure detection is inverted, causing a fault to be reported. During these faults, no hardware fault condition is injected in the device. Bit number | Enable Fault 0 | GPADC_INTERNAL_SIGNALS_ MONITOR Others | RESERVED For each bit, 1 = inject fault, 0 = remove injected fault
RESERVED	3	0x00000000
RESERVED	4	0x00000000

8. Chirp Parameters (CP) and Chirp Quality (CQ) data

8.1 Chirp Parameters data

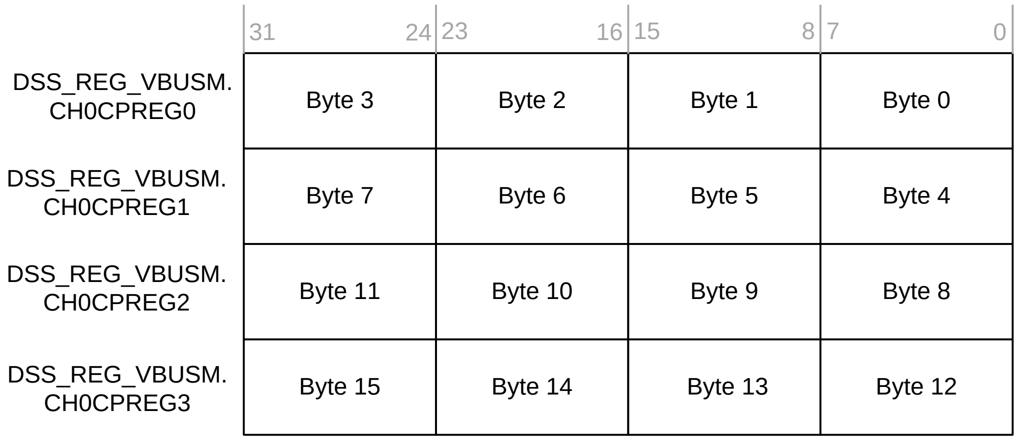
Chirp parameter information is always updated in the CP registers DSS_REG_VBUSM__CPREG[0-3] for single chirp use case.

NOTE: Chirp Number is always reset every burst by the hardware



	Channel 0				Channel 1				Channel 2				Channel 3			
Bits	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
0	Channel Number	Reserved	Chirp Number[7:0]	Chirp Number[11:8]	Channel Number	Reserved	Chirp Number[7:0]	Chirp Number[11:8]	Channel Number	Reserved	Chirp Number[7:0]	Chirp Number[11:8]	Channel Number	Reserved	Chirp Number[7:0]	Chirp Number[11:8]
1																
2	Profile Number				Profile Number				Profile Number				Profile Number			
3																
4				Reserved				Reserved				Reserved				Reserved
5																
6	Reserved				Reserved				Reserved				Reserved			
7																

Figure 8.1 – Chirp parameter information fields



	31 – 24	23 – 16	15 – 8	7 – 0
DSS_REG_VBUSM. CH0CPREG0	Byte 3	Byte 2	Byte 1	Byte 0
DSS_REG_VBUSM. CH0CPREG1	Byte 7	Byte 6	Byte 5	Byte 4
DSS_REG_VBUSM. CH0CPREG2	Byte 11	Byte 10	Byte 9	Byte 8
DSS_REG_VBUSM. CH0CPREG3	Byte 15	Byte 14	Byte 13	Byte 12

Figure 8.2 – Chirp parameter information from DSS registers

For multichip use case, the CP data is available for up to 8 chirps in DSS_REG_VBUSM.CH[0-7]CPREG[0-3].

8.2 Chirp Quality data

Chirp quality information is divided into 3 parts

1. CQ0 – Wideband signal and image energy information (Reserved for future use)
2. CQ1 – RX signal and image band energy statistics
3. CQ2 – RX ADC and IF saturation information

CQ data will be available in CQ RAM which is a ping-pong memory when the CQ monitors are enabled. Currently supported CQ monitors are AWR_MONITOR_RX_SATURATION_DETECTOR_CONF_SB for CQ2 and AWR_MONITOR_SIG_IMG_MONITOR_CONF_SB for CQ1. CQ data will be refreshed every chirp by the hardware. User has to ensure that before the next chirp finishes, the current chirps' CQ data is either processed or transferred to a local memory for further processing.

The starting location (on 128 bit boundary) of each CQ data within the CQ memory can be configured by programming DSS_REG.CQCFG1[12:4] for CQ0, DSS_REG.CQCFG1[21:13] for CQ1 and DSS_REG.CQCFG1[30:22] for CQ2.

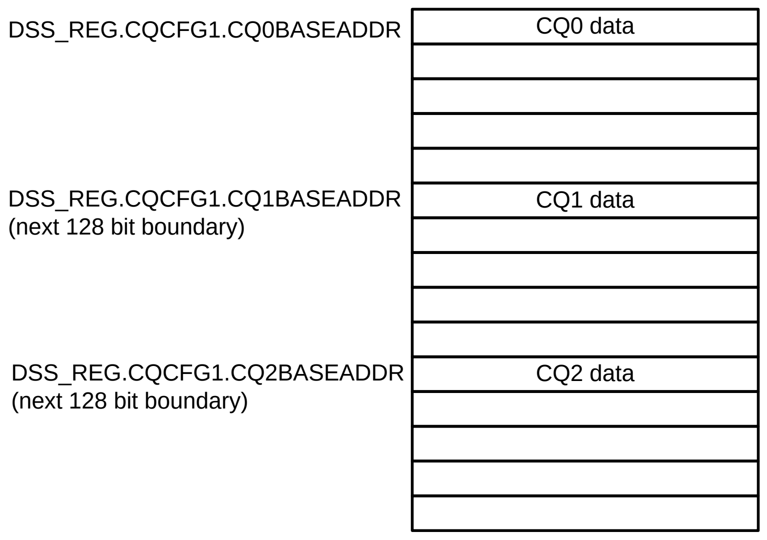

DSS_REG.CQCFG1.CQ0BASEADDR
CQ0 data
DSS_REG.CQCFG1.CQ1BASEADDR (next 128 bit boundary)
CQ1 data
DSS_REG.CQCFG1.CQ2BASEADDR (next 128 bit boundary)
CQ2 data

Figure 8.3 – CQ data start address configuration in single chirp use case

For N-chirp use case, when user wishes to process N chirps simultaneously, then CQ0 for all N chirps will be concatenated together in memory. Similarly CQ1 and CQ2 for all N chirps will also be concatenated together.

NOTE: When CQ data is concatenated in N-chirp use case, the CQ data for new chirp starts on the next 128 bit boundary

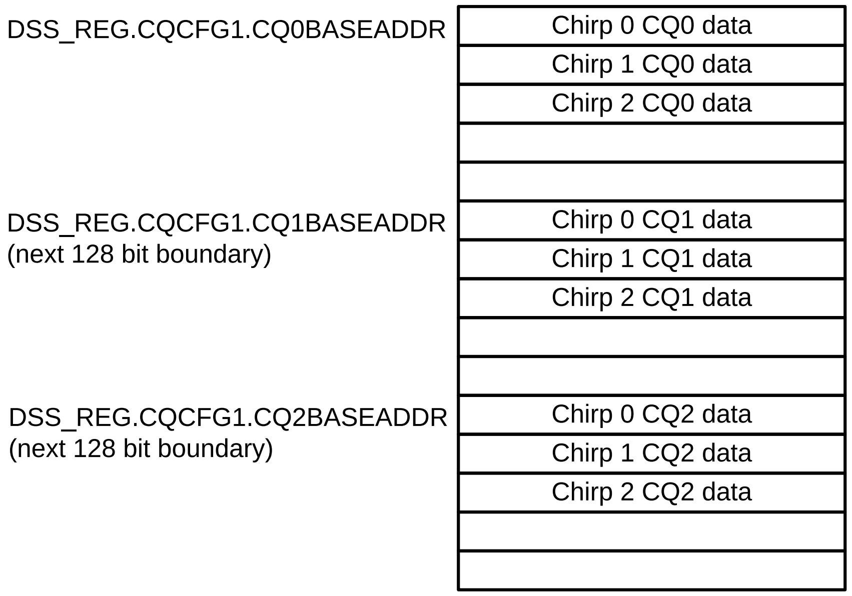


Figure 8.4 – CQ data start address configuration in multi chirp use case

The CQDATAWIDTH parameter in DSS_REG.CQCFG1 defines the packing of the CQ data in the CQ memory in either 16-bit mode, 12-bit mode or in 14-bit mode.

8.2.1 CQ1

The signal band and image band are separated using a two-channel filter bank and the ADC sampling time duration is monitored in terms of primary and secondary time slices, as shown below.

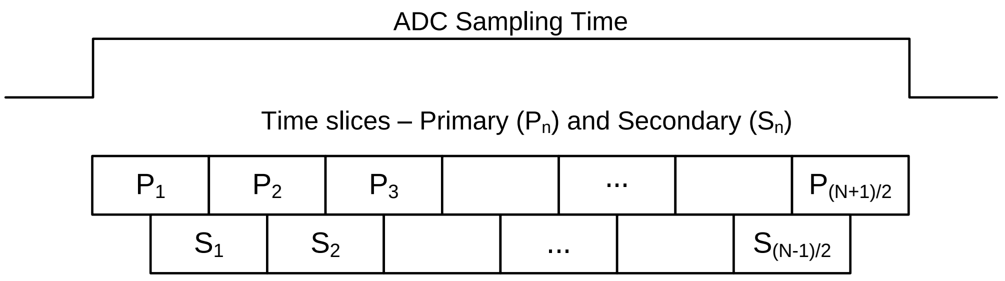


Figure 8.5 – Time slices during RX signal and image band monitor and saturation monitor

For each of the two bands (signal and image), for each time slice, the input-referred average power in the slice in negative dBm is recorded as an 8-bit unsigned number, with 1 LSB = -0.5 dBm

CQ1 data is stored in memory as shown below (in 16-bit mode)

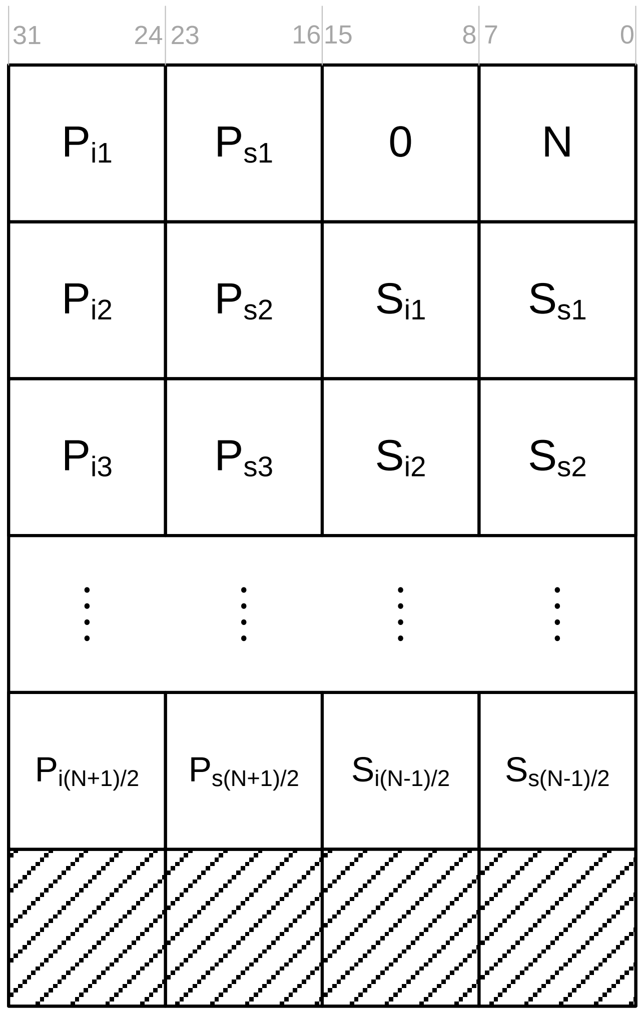


Figure 8.6 – CQ1 data format in memory in 16-bit mode

N indicates the total number of primary and secondary slices which are monitored (maximum value of N is 127). $P_{\{s,i\}n}$ indicates the power of primary slice n for {signal, image} band and $S_{\{s, i\}n}$ indicates the power of secondary slice n for {signal, image} band. Each power is encoded in 8 bit unsigned number with each LSB representing -0.5 dBm.

Since maximum value of N is 127, the maximum size of CQ1 data in 16-bit mode is 256 bytes

NOTE: In real output mode, since there is no image band visibility, only the signal band statistics will be meaningful

Similarly, in 12-bit and 14-bit modes, the CQ1 data in CQ memory will be packed as shown below. Only the relevant bits in each 16 bits of memory (either 12 bits or 14 bits) are useful and other bits and not written by hardware.

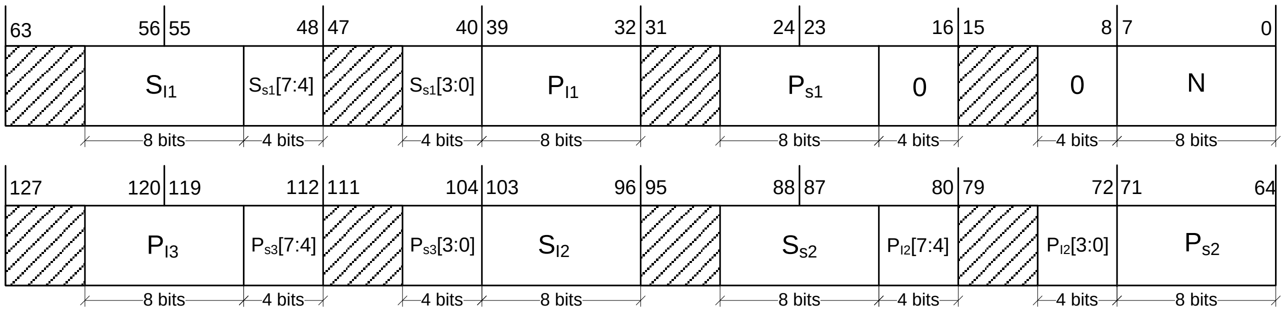


Figure 8.7 – CQ1 data format in memory in 12-bit mode

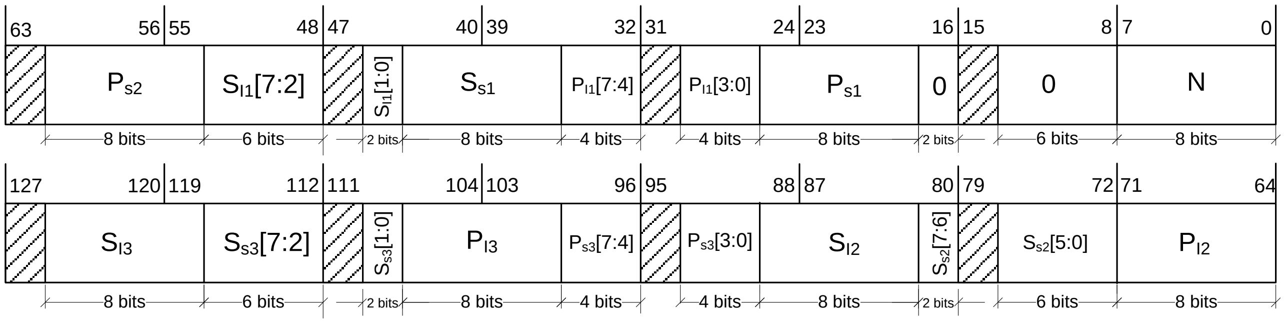


Figure 8.8 – CQ1 data format in memory in 14-bit mode

8.2.2 CQ2

The analog to digital interface includes a 100 MHz bit stream indicating saturation events in the ADC/IF sections, for each channel. This one-bit indicator for each channel is monitored during the ADC sampling time duration in a time-sliced manner, as shown in Figure 8.5.

For each time slice, a saturation event count is recorded. This count is the sum of saturation event counts across all RX channels selected for monitoring, capped to a maximum count of 255 (8 bits). The saturation counts are stored in memory as shown below

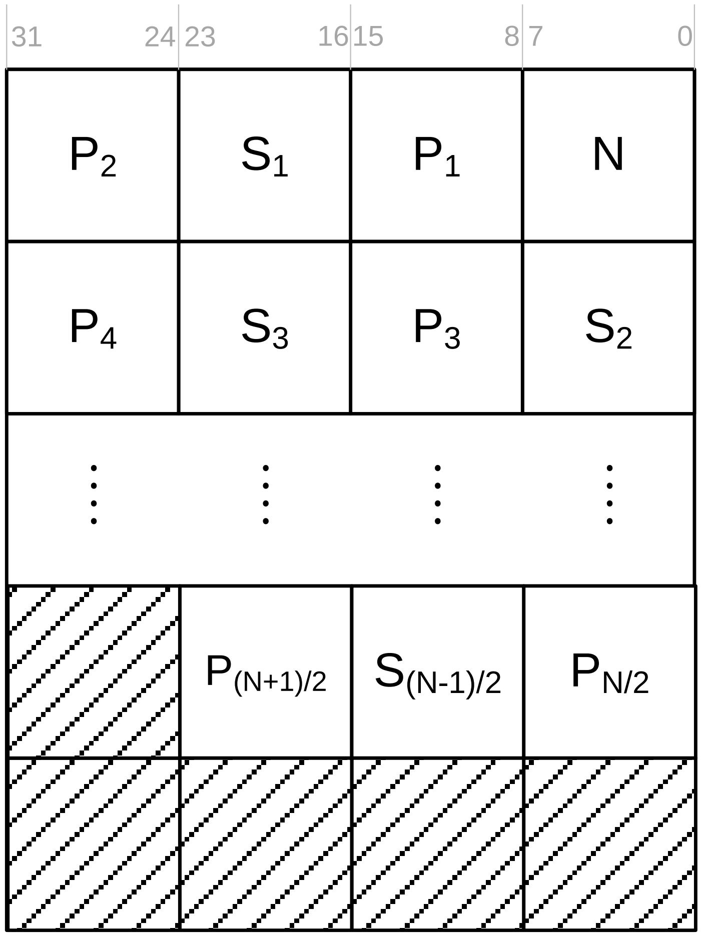


Figure 8.9 – CQ2 data format in memory in 16-bit mode

N indicates the total number of primary and secondary slices which are monitored (maximum value of N is 127). P_n indicates the accumulated saturation count for all enabled RX channels in primary slice n, S_n indicates the accumulated saturation count for all enabled RX channels in secondary slice n.

Since maximum value of N is 127, the maximum size of CQ2 data in 16-bit mode is 128 bytes.

Similarly, in 12-bit and 14-bit modes, the CQ2 data in CQ memory will be packed as shown below. Only the relevant bits in each 16 bits of memory (either 12 bits or 14 bits) are useful and other bits and not written by hardware.

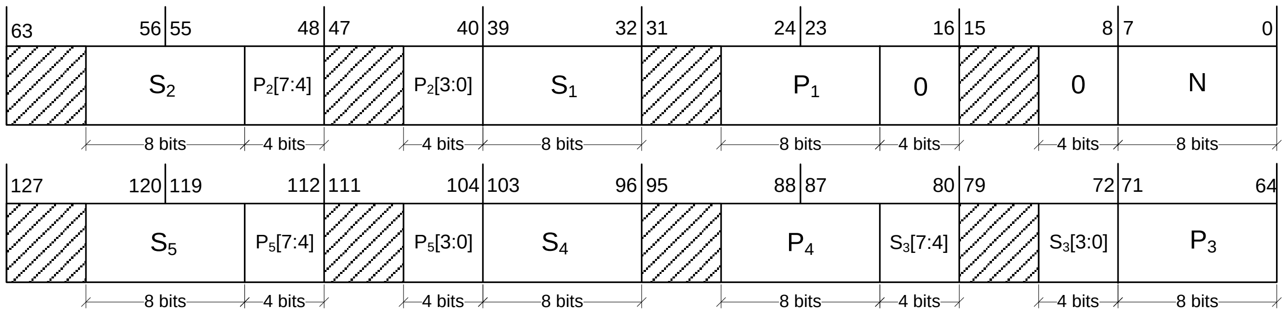


Figure 8.10 – CQ2 data format in memory in 12-bit mode

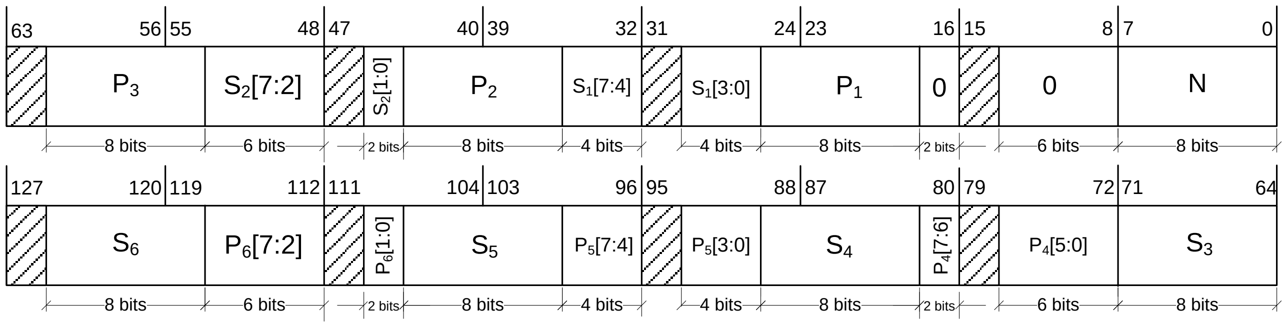


Figure 8.11 – CQ2 data format in memory in 14-bit mode

9. Calibration and monitoring durations

9.1 Boot time calibration durations

Table 9.1 – Duration of boot time calibrations

Sl. No	Calibration	Duration (us)
1	APLL	330
2	Synth VCO	1300
3	LO DIST	12
4	ADC DC	600
5	HPF cutoff	3500
6	LPF cut off	3200
7	Peak detector	4200
8	TX power (assumes 2 TX use-case)	6000
9	RX gain	2300
10	TX phase (not enabled in firmware)	0
11	RX IQMM	32000

9.2 Run time calibration durations

Table 9.2 – Duration of run time calibrations

Sl. No	Calibration	Duration (us)
1	APLL	150
2	Synth VCO	300
3	LO DIST	30
4	Peak detector	500
5	TX power (assumes 1 TX, 1 profile)	800
6	RX gain	30
7	Application of calibration to hardware (This needs to be included always)	150

To configure CALIB_MON_TIME_UNIT, user has to calculate the total available IDLE time in the frame and subtract 100us for every frame to allow for preparion of frame. The duration for all the enabled calibrations should be included and following software overheads should be added to that number

9.3 Monitoring duration

Table 9.3 – Duration of monitors

Sl. No	Monitors	Duration (us)
1	RX gain phase (assumes 1 RF frequency)	1250
2	RX noise figure (assumes 1 RF frequency)	250
3	RX IF stage (assumes 1 RF frequency)	1000
4	TX power (assumes 1 TX, 1 RF frequency)	200
5	TX ballbreak (assumes 1 TX)	250
6	TX gain phase mismatch (assumes 1 TX, 1 RF frequency)	400
7	TX BPM (assumes 1 TX)	575
8	Synthesizer frequency	0
9	External analog signals (all 6 GPADC channels enabled)	150
10	TX Internal analog signals (assumes 1 TX)	200
11	RX internal analog signals	1700
12	PMCLKLO internal analog signals	400
13	GPADC internal signals	50
14	PLL control voltage	210
15	Dual clock comparator (assumes 6 clock comparators)	110
16	RX saturation detector	0
17	RX signal and image band monitor	0
18	RX mixer input power	350

9.4 Software overheads

When the calibrations or monitorings are enabled, the software needs certain time for reading the temperature sensors, reading the DFE statistics, preparing the calibration or monitoring reports and to clear the watch dog. All these time durations should also be accounted when computing the CALIB_MON_TIME_UNIT. The details of the software overheards are given in the table below

Table 9.4 – Software overheads every FTTI that should be accounted to program CALIB_MON_TIME_UNIT and CALIBRATION_PERIODICITY

Sl. No	Software overhead	Duration (us)
1	Periodic monitoring of stack usage	20
2	Minimum monitoring duration (report formation, digital energy monitor at the end of FTTI, temperature read every FTTI)	1000
3	Minimum calibration duration (report formation, temperature read every FTTI)	500
4	Idle time needed per FTTI for windowed watchdog	Frame period*CALIB_MON_TIME_UNIT/16 ~12.5% of FTTI is reserved for watchdog

Example 1

A user has enabled 2 TX, uses only 1 profile, frame configuration consists of 64 chirps, each chirp is of duration is 66 µs (56 µs ramp time and 10 µs chirp idle time) and frame periodicity is 10 ms. User has enabled all run time calibrations. None of the analog monitoring is enabled.

Frame duty cycle is 42.24%

Idle time per frame is 5.776 ms.

Idle time available for calibration/monitoring per frame is 5.676 ms (100 µs is for frame preparation)

Time needed for all run time calibrations is 150 + 300 + 30 + 500 + (800 * 2) + 30 + 150 = 2760 µs

Minimum time for software overheads is 20 + 1000 + 500 + 10000*1/16 = 2145 µs

Total time needed per frame for calibration is 2760 + 2145 = 4905 µs which is less than 5.676 ms and hence this configuration will be honored by the MMIC device.

User can set CALIB_MON_TIME_UNIT to 1 and CALIBRATION_PERIODICITY as 100. With this setting calibrations are triggered once every 100 frames (i.e. once every 1 s)

Example 2

Consider another example where the frame configuration remains the same as in example 1, but frame periodicity is reduced to 8 ms.

Frame duty cycle is 52.8%

Idle time per frame is 3.776 ms.

Idle time available for calibration/monitoring per frame is 3.676 ms (100 µs is for frame preparation)

Time needed for all run time calibrations is 150 + 300 + 30 + 500 + (800 * 2) + 30 + 150 = 2760 µs

Minimum time for software overheads is 20 + 1000 + 500 + 8000*1/16 = 2020 µs

Total time needed per frame for calibration is 2760 + 2020 = 4780 µs which is more than 3.676 ms and hence this configuration will be **not** be honored by the MMIC device.

User can set CALIB_MON_TIME_UNIT to 2 and CALIBRATION_PERIODICITY as 63. With this setting calibrations are triggered once every 124 frames (i.e. once every 1.006 s)

Example 3

A user has enabled 2 TX, uses 2 profiles, frame configuration consists of 32 chirps, each chirp is of duration is 90 µs (80 µs ramp time and 10 µs chirp idle time) and frame periodicity is 6 ms. User has enabled all run time calibrations. None of the analog monitoring is enabled

Frame duty cycle is 48%

Idle time per frame is 3.120 ms.

Idle time available for calibration/monitoring per frame is 3.020 ms (100 µs is for frame preparation)

Time needed for all run time calibrations is 150 + 300 + 30 + 500 + (800 * 2 * 2) + 30 + 150 = 4360 µs

Minimum time for software overheads is 20 + 1000 + 500 + 6000*1/16 = 1895 µs if CALIB_MON_TIME_UNIT is set as 1

Total time needed per frame for calibration is 4360 + 1895 = 6255 µs which is more than 3.020 ms and hence CALIB_MON_TIME_UNIT = 1 configuration will be **not** be honored by the MMIC device.

User can set CALIB_MON_TIME_UNIT to 3 and CALIBRATION_PERIODICITY as 56. With this setting, minimum required time is 7005 µs and available idle time for calibration/monitoring is 9.06 ms and calibrations are triggered once every 168 frames (i.e. once every 1.008 s)

Example 4

A user has enabled 2 TX, uses 2 profiles, frame configuration consists of 32 chirps, each chirp is of duration is 90 µs (80 µs ramp time and 10 µs chirp idle time) and frame periodicity is 6 ms. User has enabled all run time calibrations. Analog monitorings which are enabled are (a) TX output power monitor for TX0 and TX1 (b) TX BPM monitor for TX0 and TX1 (c) RX gain phase monitor and (d) RX noise figure monitor. Each of the monitors are configured to be run for 1 profile and 3 RF frequencies (low, mid and high) as defined by the profile.

Frame duty cycle is 48%

Idle time per frame is 3.120 ms.

Idle time available for calibration/monitoring per frame is 3.020 ms (100 µs is for frame preparation)

Time needed for all run time calibrations is 150 + 300 + 30 + 500 + (800 * 2 * 2) + 30 + 150 = 4360 µs

Time needed for all monitoring is 1250*3 + 250*3 + 200*3*2 + 575*2 = 6850 µs

Minimum time for software overheads is 20 + 1000 + 500 + 6000*1/16 = 1895 µs if CALIB_MON_TIME_UNIT is set as 1

Total time needed per frame for calibration and monitoring is 4360 + 6850 + 1895 = 13105 µs which is more than 3.020 ms and hence CALIB_MON_TIME_UNIT = 1 configuration will be **not** be honored by the MMIC device.

User can set CALIB_MON_TIME_UNIT to 5 and CALIBRATION_PERIODICITY as 34. With this setting, minimum required time for calibration and monitoring is 14605 µs and available idle time for calibration/monitoring is 15.1 ms. Monitoring is triggered once in 5 frames and calibration is triggered once in 165 frames (i.e. once every 1.020 s)

10. mmWave Application Framework

TI's mmWaveLink framework

- Is a portable framework which abstracts the services provided by the Radar device.
- Provides APIs to control the Radar device and RF modules
- Abstracts communication protocol with the Radar device
- Handles communication errors, Notifies exceptions
- Is platform independent
- Is OS agnostic
- Can work in single threaded (non OS) environment
- For AWR1642 will reside in the internal Cortex R4F and will provide same APIs for Radar Controls.

mmWaveLink APIs have been divided into following modules depending upon its functionalities:

- Device manager
- Radar RF control
- Radar data control
- Safety Monitoring
- Platform adaptation
  - Communication channel (SPI, UART, I2C, Mailbox) adaptation
  - Operating system adaptation
  - Driver adaptation (CRC, Interrupt, Timer etc.)

10.1 Device Manager APIs

The device manage module has the interface for enabling and controlling the Radar device. This module configures the callbacks (SPI, OS, and Interrupt etc.) for communication with the device. This module also has interface to perform the firmware patch to the device.

10.1.1 rlDevicePowerOn

This function initializes the driver and does the necessary resource allocation for the driver. It initializes the host protocol driver by creating the necessary OS services like semaphore, mutex, queues etc. It also brings multiple radar devices out of reset and opens communication channel (SPI, Mailbox etc.) with these devices. mmWaveLink Framework internally does some handshake with each connected radar device and completion is indicated by asynchronous event (AWR_AE_DEV_MSSPOWERUPDONE_SB). User application should wait for this event before invoking any other APIs.

```
rlReturnVal_t rlDevicePowerOn(rlUInt8_t deviceMap, rlClientCbs_t clientCb)
```

10.1.2 rlDevicePowerOff

This function de-initializes the device and de-allocates all the resources allocated in the initialization process.

```
rlReturnVal_t rlDevicePowerOff(void)
```

10.1.3 rlDeviceRfStart

This function initializes the RF (BIST) subsystem in the AWR1XXX device. The function returns immediately and RF initialization completion is indicated by asynchronous event (AWR_AE_DEV_RFPOWERUPDONE_SB). User application should wait for this event before invoking any Radar Sensor Control APIs

```
rlReturnVal_t rlDeviceRfStart(rlUInt8_t deviceMap)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_RFPOWERUP_MSG
SB Id(s): AWR_DEV_RFPOWERUP_SB

10.1.4 rlDeviceFileDownload

This function downloads a binary file to the AWR1XXX device. This file could be a firmware patch file, Application code, calibration data or configuration data. Large files needs to be sent in multiple chunks of smaller size. Remaining chunks should contain the number of remaining chunks to be sent. For the last chunk, remaining chunks should be 0.

```
rlReturnVal_t rlDeviceFileDownload(rlUInt8_t deviceMap, rlFileData_t *data,
rlUInt16_t remChunks)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_FILE_DOWNLOAD_MSG
SB Id(s): AWR_DEV_FILE_DOWNLOAD_SB

Note1: This API downloads the file only in the internal RAM and not on the FLASH.

Note2: Max Chunk size on SPI communication channel is 256 bytes including SYNC (4 Bytes), Header (12 Bytes) and CRC (0, 2, 4 or 8 Bytes). So Max Payload chunk should be 256 – (SYNC + HEADER + CRC)

10.1.5 rlDeviceGetVersion

This function is used to get the AWR1XXX hardware and firmware version. This provides the device variant (AWR1243, AWR1642, etc.), device version, ROM version, ROM build date, firmware patch version and the build date.

```
rlReturnVal_t rlDeviceGetVersion(rlUInt8_t deviceMap, rlVersion_t *ver)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_STATUS_GET_MSG, AWR_RF_STATUS_GET_MSG
SB Id(s): AWR_MSSVERSION_GET_SB, AWR_RFVERSION_GET_SB

10.1.6 rlDeviceGetMssVersion

This function is used to get the AWR1XXX Master SS firmware version which includes ROM version, ROM build date, firmware patch version and the build date.

```
rlReturnVal_t rlDeviceGetMssVersion(rlUInt8_t deviceMap, rlFwVersionParam_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_STATUS_GET_MSG
SB Id(s): AWR_MSSVERSION_GET_SB

10.1.7 rlDeviceGetRfVersion

This function is used to get the AWR1XXX Radar SS firmware version which includes ROM version, ROM build date, firmware patch version and the build date.

```
rlReturnVal_t rlDeviceGetRfVersion(rlUInt8_t deviceMap, rlFwVersionParam_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATUS_GET_MSG
SB Id(s): AWR_RFVERSION_GET_SB

10.1.8 rlDeviceGetMmWaveLinkVersion

This function is used to get version of mmWaveLink.

```
rlReturnVal_t rlDeviceGetMmWaveLinkVersion(rlSwVersionParam_t *data)
```

10.1.9 rlDeviceSetContStreamingModeConfig

This function configures the transfer of captured ADC samplesfor continuous streaming mode.

```
rlReturnVal_t rlDeviceSetContStreamingModeConfig(rlUInt8_t deviceMap,
rlDevContStreamingModeCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AWR_DEV_RX_CONTSTREAMING_MODE_CONF_SET_SB

10.1.10 rlDeviceGetContStreamingModeConfig

This function reads the ADC samples configuration for Continuous streaming mode.

```
rlReturnVal_t rlDeviceGetContStreamingModeConfig(rlUInt8_t deviceMap,
rlDevContStreamingModeCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_GET_MSG
SB Id(s): AWR_DEV_RX_CONTSTREAMING_MODE_CONF_GET_SB

10.1.11 rlDeviceConfigureCrc

This function is used to configure CRC type for mmWaveLink driver.

```
rlReturnVal_t rlDeviceConfigureCrc(rlCrcType_t rlCrcType)
```

10.1.12 rlDeviceConfigureAckTimeout

This function is used to configure the acknowledgement timeout for mmWaveLink driver.

```
rlReturnVal_t rlDeviceConfigureAckTimeout(rlUInt32_t ackTimeout)
```

10.1.13 rlDeviceAddDevices

This function is used to add cascaded mmwave radar devices. User needs to give connected device ID.

```
rlReturnVal_t rlDeviceAddDevices(rlUInt8_t deviceMap)
```

10.1.14 rlDeviceRemoveDevices

This function is used to remove cascaded mmwave radar devices. User needs to give connected device ID.

```
rlReturnVal_t rlDeviceRemoveDevices(rlUInt8_t deviceMap)
```

10.1.15 rlDeviceMcuClkConfig

This function contains the configurations to setup the desired frequency of the MCU Clock.

```
rlReturnVal_t rlDeviceMcuClkConfig(rlUInt8_t deviceMap, rlMcuClkCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AR_DEV_MCUCLOCK_CONF_SET_SB

10.1.16 rlDevicePmicClkConfig

This function contains the configurations for the PMIC clock.

```
rlReturnVal_t rlDevicePmicClkConfig(rlUInt8_t deviceMap, rlPmicClkCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AR_DEV_PMICCLOCK_CONF_SET_SB

10.1.17 rlDeviceLatentFaultTests

This function contains the configurations for latent fault test for the master subsystem.

```
rlReturnVal_t rlDeviceLatentFaultTests(rlUInt8_t deviceMap, rllatentFault_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AR_MSS_LATENTFAULT_TEST_CONF_SB

10.1.18 rlDeviceEnablePeriodicTests

This function contains the configurations for periodic test for the master subsystem.

```
rlReturnVal_t rlDeviceEnablePeriodicTests(rlUInt8_t deviceMap, rlperiodicTest_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AR_MSS_PERIODICTESTS_CONF_SB

10.1.19 rlDeviceSetTestPatternConfig

This function contains the configurations to setup the test pattern to be generated and transferred over the selected high speed interface (LVDS/CSI2).

```
rlReturnVal_t rlDeviceSetTestPatternConfig(rlUInt8_t deviceMap, rltestPattern_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AR_DEV_TESTPATTERN_GEN_SET_SB

10.1.20 rlDeviceSetMiscConfig

Set misc. device configurations of MasterMSS, where currently it set CRC type for async event message sent by MSS to Host. This API is valid only for AWR1243 mmWave device when mmWaveLink instance is running on External Host Processor.

```
rlReturnVal_t rlDeviceSetMiscConfig(rlUInt8_t deviceMap,  rlDevConfig_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): **AWR_DEV_CONFIGURATION_SET_SB**

10.2 Radar Sensor Control APIs

The Radar Sensor Control APIs are responsible for the configuration of the chirp for radar operation. This includes configuring the profile parameter, associating defined profile to chirp index, frame configuration, TX/RX channel configuration, ADC output configuration, Noise figure vs Linearity configuration, BPM configuration and other RF parameters.

10.2.1 rlSetChannelConfig

This function sets the static device configurations of how many RX and TX channel are needed for operation of the device. It also defines static configurations related to whether the sensor uses a single AWR1xxx or multiple AWR1xxx chips to realize a larger antenna array (multiple is applicable only in AWR12xx). This is applicable for given power cycle.

```
rlReturnVal_t rlSetChannelConfig(rlUInt8_t deviceMap, rlChanCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_CHAN_CONF_SET_SB

10.2.2 rlSetAdcOutConfig

This function sets the static device configuration for the data format of the ADC output. This would include the digital filtering output. This is applicable for given power cycle.

```
rlReturnVal_t rlSetAdcOutConfig(rlUInt8_t deviceMap, rlAdcOutCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_ADCOUT_CONF_SET_SB

10.2.3 rlSetLowPowerModeConfig

This function sets the static device configurations of low power options in the Analog filters (using I only instead of IQ), and Sigma Delta ADC root sampling clock rate (reducing rate to half to save power in small IF bandwidth applications). This setting is applicable for given power cycle.

```
rlReturnVal_t rlSetLowPowerModeConfig(rlUInt8_t deviceMap, rlLowPowerModeCfg_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_LOWPOWERMODE_CONF_SET_SB

10.2.4 rlRfInit

This function initializes the RF or the Analog subsystem for control later on. It performs the boot time calibrations related to clock, transmitter and receiver.

```
rlReturnVal_t rlRfInit(rlUInt8_t deviceMap);
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_INIT_MSG
SB Id(s): AWR_RF_INIT_SB

Once API returns, the Application should wait for RF Initialization/Calibration Completiion Asynchronous event AWR_AE_RF_INITCALIBSTATUS_SB

10.2.5 rlSetProfileConfig

This function sets the FMCW radar chirp properties like FMCW slope, chirp duration, TX power etc. The API allows multiple profiles to be set together by passing the array of profile data along with count of profiles

```
rlReturnVal_t rlSetProfileConfig(rlUInt8_t deviceMap, rlUInt16_t cnt,
rlProfileCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_PROFILE_CONF_SET_SB

10.2.6 rlGetProfileConfig

This function gets the FMCW radar chirp properties like FMCW slope, chirp duration, TX power etc. from the device.

```
rlReturnVal_t rlGetProfileConfig(rlUInt8_t deviceMap, rlUInt16_t profileId,
rlProfileCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_GET_MSG
SB Id(s): AWR_PROFILE_CONF_GET_SB

10.2.7 rlSetChirpConfig

This function sets the chirp to chirp variations on top of the chirp profile defined in `rlSetProfileConfig()`. This function indicates the profile to be used by each chirp in frame in addition to fine tuning parameters to the FMCW start frequency and idle time for each chirp.

```
rlReturnVal_t rlSetChirpConfig(rlUInt8_t deviceMap, rlUInt16_t cnt,
rlChirpCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_CHIRP_CONF_SET_SB

10.2.8 rlGetChirpConfig

This function gets the chirp configuration from the device.

```
rlReturnVal_t rlGetChirpConfig(rlUInt8_t deviceMap, rlUInt16_t chirpStartIdx,
rlUInt16_t chirpEndIdx, rlChirpCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_CHIRP_CONF_GET_SB

10.2.9 rlSetFrameConfig

This function defines the frame properties like the sequence of chirps to be transmitted, number of frames to be transmitted, periodicity of the frame and the trigger method.

```
rlReturnVal_t rlSetFrameConfig(rlUInt8_t deviceMap, rlFrameCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG, AWR_DEV_FRAME_CONFIG_APPLY_MSG
SB Id(s): AWR_FRAME_CONF_SET_SB, AWR_DEV_FRAME_CONFIG_APPLY_SB

10.2.10 rlGetFrameConfig

This function reads the frame properties of the device.

```
rlReturnVal_t rlGetFrameConfig(rlUInt8_t deviceMap, rlFrameCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_GET_MSG
SB Id(s): AWR_FRAME_CONF_GET_SB

10.2.11 rlSetAdvFrameConfig

This function defines the advance frame properties like the sequence of chirps to be transmitted, number of sub-frames, number of frames to be transmitted, periodicity of the frame and the trigger method.

```
rlReturnVal_t rlSetAdvFrameConfig (rlUInt8_t deviceMap, rlAdvFrameCfg_t* data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG, AWR_DEV_FRAME_CONFIG_APPLY_MSG
SB Id(s): AWR_ADVANCED_FRAME_CONF_SB, AWR_DEV_FRAME_CONFIG_APPLY_SB

10.2.12 rlGetAdvFrameConfig

This function reads the advance frame properties of the device.

```
rlReturnVal_t rlGetAdvFrameConfig(rlUInt8_t deviceMap, rlAdvFrameCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_GET_MSG
SB Id(s): AWR_ADVANCED_FRAME_CONF_SB

10.2.13 *rlSetBpmCommonConfig*

This API defines static configurations related to BPM (Binary Phase Modulation) feature in each of the TXs. E.g. the source of the BPM pattern (one constant value for each chirp as defined, or intra-chirp pseudo random BPM pattern as found by a programmable LFSR or a programmable sequence inside each chirp), are defined here.

```
rlReturnVal_t rlSetBpmCommonConfig(rlUInt8_t deviceMap, rlBpmCommonCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_ADVANCED_FEATURES_CONF_SET_MSG
SB Id(s): AWR_BPM_COMMON_CONF_SET_SB

10.2.14 *rlSetBpmChirpConfig*

This API defines static configurations related to BPM (Binary Phase Modulation) feature in each of the TXs

```
rlReturnVal_t rlSetBpmChirpConfig(rlUInt8_t deviceMap, rlBpmChirpCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_ADVANCED_FEATURES_CONF_SET_MSG
SB Id(s): AWR_BPM_CHIRP_CONF_SET_SB

10.2.15 *rlSensorStart*

This function triggers the transmission of the frames as per the frame and chirp configuration.

```
rlReturnVal_t rlSensorStart(rlUInt8_t deviceMap)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_FRAME_TRIG_MSG
SB Id(s): AWR_FRAMESTARTSTOP_CONF_SB

10.2.16 *rlSensorStop*

This function stops the transmission of the frames.

```
rlReturnVal_t rlSensorStop(rlUInt8_t deviceMap)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_FRAME_TRIG_MSG
SB Id(s): AWR_FRAMESTARTSTOP_CONF_SB

10.2.17 *rlSetTestSourceConfig*

This function enables the test source.

```
rlReturnVal_t rlSetTestSourceConfig(rlUInt8_t deviceMap, rlTestSource_t* data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_ RF_ MISC_CONF_SET_MSG
SB Id(s): AWR_RF_TEST_SOURCE_CONFIG_SET_SB

10.2.18 *rlTestSourceEnable*

This function enables the test source.

```
rlReturnVal_t rlTestSourceEnable(rlUInt8_t deviceMap, rlTestSourceEnable_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_ MISC_CONF_SET_MSG
SB Id(s): AWR_RF_TEST_SOURCE_ENABLE_SET_SB

10.2.19 *rlRfGetTemperatureReport*

This function reads the temperature from the device.

```
rlReturnVal_t rlRfGetTemperatureReport(rlUInt8_t deviceMap, rlRfTempData_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MISC_CONF_GET_MSG
SB Id(s): AWR_RF_TEMPERATURE_GET_SB

10.2.20 rlSetContModeConfig

This function sets continuous mode properties like start frequency, TX power etc. for radar device.

```
rlReturnVal_t rlSetContModeConfig(rlUInt8_t deviceMap,  rlContModeCfg_t* data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_CONT_STREAMING_MODE_CONF_SET_SB

10.2.21 rlEnableContMode

This function enables/disables the FMCW radar continuous mode.

```
rlReturnVal_t rlEnableContMode(rlUInt8_t deviceMap,  rlContModeEn_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_CONT_STREAMING_MODE_EN_SB

10.2.22 rlRfDfeRxStatisticsReport

This function will get all DFE statistics report for all receivers and profiles.

```
rlReturnVal_t rlRfDfeRxStatisticsReport(rlUInt8_t deviceMap,
rlDfeStatReport_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_REPORT_GET_MSG
SB Id(s): AWR_RF_DFE_STATISTICS_REPORT_GET_SB

10.2.23 rlRfDynamicPowerSave

This function configures whether to enable dynamic power saving during inter-chirp IDLE times by turning off various circuits e.g. TX, RX, LO Distribution blocks.

```
rlReturnVal_t rlRfDynamicPowerSave(rlUInt8_t deviceMap,  rlDynPwrSave_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_DYNAMICPOWERSAVE_CONF_SET_SB

10.2.24 *rlRfSetDeviceCfg*

This function will set the direction of asynchronous event.

```
rlReturnVal_t rlRfSetDeviceCfg(rlUInt8_t deviceMap,  rlRfDevCfg_t  *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_RF_DEVICE_CFG_SB

10.2.25 *rlSetGpAdcConfig*

This function will set all relevant parameters for GPADC. This is applicable only in xWR1642 and is not supported in AWR1243 and xWR1443.

```
rlReturnVal_t rlSetGpAdcConfig(rlUInt8_t deviceMap,  rlGpAdcCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MISC_CONF_SET_MSG
SB Id(s): AWR_RF_GPADC_CFG_SET_SB

10.2.26 *rlRfSetLdoBypassConfig*

This function Enables/Disables LDO bypass mode

```
rlReturnVal_t rlRfSetLdoBypassConfig(rlUInt8_t deviceMap,  rlRfLdoBypassCfg_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MISC_CONF_SET_MSG
SB Id(s): AWR_RF_LDO_BYPASS_SB

10.2.27 *rlRfSetPhaseShiftConfig*

This function configures the per-chirp phase shift for each TX,

```
rlReturnVal_t rlRfSetPhaseShiftConfig(rlUInt8_t deviceMap, rlUInt16_t cnt,
rlRfPhaseShiftCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_PERCHIRPPHASESHIFT_CONF_SB

10.2.28 rlRfSetPALoopbackConfig

This function Enables/Disables PA loopback for all enabled profiles

```
rlReturnVal_t rlRfSetPALoopbackConfig(rlUInt8_t deviceMap,
rlRfPALoopbackCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MISC_CONF_SET_MSG
SB Id(s): AWR_RF_PALOOPBACK_CFG_SB

10.2.29 rlRfSetPSLoopbackConfig

This function Enables/Disables phase shift loopback for all enabled profiles

```
rlReturnVal_t rlRfSetPSLoopbackConfig(rlUInt8_t deviceMap,
rlRfPSLoopbackCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MISC_CONF_SET_MSG
SB Id(s): AWR_RF_PSLOOPBACK_CFG_SB

10.2.30 rlRfSetIFLoopbackConfig

This function Enables/Disables RF IF loopback for all enabled profiles

```
rlReturnVal_t rlRfSetIFLoopbackConfig(rlUInt8_t deviceMap,
rlRfIFLoopbackCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MISC_CONF_SET_MSG
SB Id(s): AWR_RF_IFLOOPBACK_CFG_SB

10.2.31 rlRfSetProgFiltCoeffRam

This function sets Programmable Filter coefficient RAM. This API is applicable only on xWR1642.

```
rlReturnVal_t rlRfSetProgFiltCoeffRam(rlUInt8_t deviceMap,
rlRfProgFiltCoeff_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_PROG_FILT_COEFF_RAM_SET_SB

10.2.32 rlRfSetProgFiltConfig

This function selects programmable filter coefficient RAM and maps it to configured profile ID. This API is applicable only on xWR1642.

```
rlReturnVal_t rlRfSetProgFiltConfig(rlUInt8_t deviceMap,  rlRfProgFiltConf_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_PROG_FILT_CONF_SET_SB

10.2.33 rlRfSetMiscConfig

This function configures miscellaneous feature such as per chirp phase shifter

```
rlReturnVal_t rlRfSetMiscConfig(rlUInt8_t deviceMap,  rlRfMiscConf_t* data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_RF_RADAR_MISC_CTL_SB

10.2.34 rlRfSetCalMonTimeUnitConfig

This function configures the calibration monitoring time unit

```
rlReturnVal_t rlRfSetCalMonTimeUnitConfig(rlUInt8_t deviceMap,
rlRfCalMonTimeUntConf_t* data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_CALIB_MON_TIME_UNIT_CONF_SB

10.2.35 rlRfSetCalMonFreqLimitConfig

This function configures calibration/monitoring frequency limit

```
rlReturnVal_t rlRfSetCalMonFreqLimitConfig(rlUInt8_t deviceMap,
rlRfCalMonFreqLimitConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_CAL_MON_FREQUENCY_LIMITS_SB

10.2.36 rlRfInitCalibConfig

This function configures RF Init calibration mask bits and report type

```
rlReturnVal_t rlRfInitCalibConfig(rlUInt8_t deviceMap,  rlRfInitCalConf_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_RF_INIT_CALIBRATION_CONF_SB

10.2.37 rlRfRunTimeCalibConfig

This function configures RF periodic calibration to be triggered autonomously and/or triggers a one-time calibration instantaneously.

```
rlReturnVal_t rlRfRunTimeCalibConfig(rlUInt8_t deviceMap,
rlRunTimeCalibConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_RUN_TIME_CALIBRATION_CONF_AND_TRIGGER_SB

10.2.38 rlRfInterRxGainPhaseConfig

This function allows configuration of the correction for inter-RX gain and phase mismatch in digital

```
rlReturnVal_t rlRfInterRxGainPhaseConfig(rlUInt8_t deviceMap,
rlInterRxGainPhConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_INTER_RX_GAIN_PHASE_CONTROL_SB

10.2.39 rlRxGainTempLutGet

This function gets the temperature based RX gain LUT for a given profile

```
rlReturnVal_t rlRxGainTempLutGet(rlUInt8_t deviceMap,
rlRxGainTempLutReadReq_t *data, rlRxGainTempLutData_t *outData)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_RX_GAIN_TEMPLUT_GET_SB

10.2.40 rlTxGainTempLutGet

This function gets the temperature based TX gain LUT for a given profile

```
rlReturnVal_t rlTxGainTempLutRead(rlUInt8_t deviceMap,
rlTxGainTempLutReadReq_t *data, rlTxGainTempLutData_t *outData)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_TX_GAIN_TEMPLUT_GET_SB

10.2.41 rlRxGainTempLutSet

This function sets the temperature based RX gain LUT for a given profile

```
rlReturnVal_t rlRxGainTempLutSet(rlUInt8_t deviceMap, rlRxGainTempLutData_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_RX_GAIN_TEMPLUT_SET_SB

10.2.42 rlTxGainTempLutSet

This function sets the temperature based TX gain LUT for a given profile.

```
rlReturnVal_t rlTxGainTempLutSet(rlUInt8_t deviceMap, rlTxGainTempLutData_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_TX_GAIN_TEMPLUT_SET_SB

10.2.43 rlSetLoopBckBurstCfg

This function configures the loopback burst which will be used within a sub-frame as part of advanced frame configuration.

```
rlReturnVal_t rlSetLoopBckBurstCfg (rlUInt8_t deviceMap, rlLoopbackBurst_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_LOOPBACK_BURST_CONF_SB

10.2.44 rlSetDynChirpCfg

This function configures the chirps dynamically i.e. while frames are on-going. The HW is updated at the end of the current frame.

```
rlReturnVal_t rlSetDynChirpCfg(rlUInt8_t deviceMap, rlDynChirpCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_DYN_CHIRP_CONF_SET_SB

10.2.45 rlSetDynChirpEn

This function enables the Dynamic chirp configured using rlSetDynChirpCfg

```
rlReturnVal_t rlSetDynChirpEn(rlUInt8_t deviceMap, rlDynChirpEnCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_DYN_CHIRP_ENABLE_SB

10.2.46 rlDynPerChirpPhShifterCfgSet

This function configures the per-chirp phase shifter dynamically i.e. while frames are on-going. The HW is updated at the end of the current frame.

```
rlReturnVal_t rlDynPerChirpPhShifterCfgSet(rlUInt8_t deviceMap,
rlDynPerChirpPhShftCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_DYN_PERCHIRP_PHASESHIFTER_CONF_SET_SB

10.2.47 rlSetMultiChirpCfg

This function sets the chirp to chirp variations on top of the chirp profile.

NOTE - One can set upto 512 unique chirps which can be stored in dedicated memory inside mmWave front end. Hence user doesn't need to program the chirps during run time. Also these chirps can be sequenced in a frame using rlSetFrameConfig to create a larger FMCW signal This API is similar to rlSetChirpConfig but gives the flexibility to pass the array of chirp configuration pointers, so chirp configuration memory need not be contiguous.

```
rlReturnVal_t rlSetMultiChirpCfg(rlUInt8_t deviceMap, rlChirpCfg_t **data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_CHIRP_CONF_SET_SB

10.2.48 rlRfCalibDataStore

This function reads the calibration data from the device which can be injected later using the rlCalibDataRestore command. RadarSS will return three chunks of calibration data.

```
rlReturnVal_t rlRfCalibDataStore(rlUInt8_t deviceMap,rlCalDataStore_t data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_STATIC_CONFIG_GET_SB
SB Id(s): AWR_CAL_DATA_SAVE_SB

10.2.49 rlRfCalibDataRestore

This API restores the calibration data which was stored previously using the rlCalibDataStore command. Application needs to feed in 3 chunks of calibration data.

NOTE - Once the calibration data is restored properly in radarSS SW RAM and validated, mmWave Front end would send asynchronous event RL_RF_AE_INITCALIBSTATUS_SB indicating the result of the calibrations based on Calib data sent by the application

```
rlReturnVal_t rlRfCalibDataRestore(rlUInt8_t deviceMap,rlCalDataStore_t
data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_CAL_DATA_RESTORE_SB

10.2.50 rlGetRfBootupStatus

This function gets RadarSS/BSS bootup (Boot time monitoring tests) status

```
rlReturnVal_t rlGetRfBootupStatus(rlUInt8_t deviceMap, rlRfBootStatusCfg_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATUS_GET_MSG
SB Id(s): AWR_RF_BOOTUPBIST_STATUS_GET_SB

10.2.51 rlSetInterChirpBlkCtrl

This function programs the Inter-chip turn on and turn off times or various RF blocks

```
rlReturnVal_t rlSetInterChirpBlkCtrl(rlUInt8_t deviceMap,
rlInterChirpBlkCtrlCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_INTERCHIRP_BLOCKCONTROLS_SB

10.2.52 rlRfInterRxGainPhaseConfig

This function programs the Inter-chip turn on and turn off times or various RF blocks

```
rlReturnVal_t rlRfInterRxGainPhaseConfig(rlUInt8_t deviceMap,
rlInterRxGainPhConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_DYNAMIC_CONF_SET_MSG
SB Id(s): AWR_INTER_RX_GAIN_PHASE_CONTROL_SB

10.2.53 rlRfTxFreqPwrLimitConfig

This function sets the limits for RF frequency transmission for each TX and also TX power limits.

```
rlReturnVal_t rlRfTxFreqPwrLimitConfig(rlUInt8_t deviceMap,
rlRfTxFreqPwrLimitMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_CAL_MON_FREQUENCY_TX_POWER_LIMITS_SB

10.3 Radar Data Control APIs

The Radar Data Control APIs controls the format of the output data from the device and also controls the interface in which data needs to be transmitted. The data can be transmitted over Low Voltage Differential signaling (LVDS) or Camera Serial Interface (CSI)-2. Output data contains radar receiver ADC output data and may optionally contain Chirp parameters (CP) and Chirp Quality (CQ) data.

10.3.1 rlDeviceSetHsiConfig

This function sets the High Speed Interface (LVDS/CSI2) clock, lane, and data rate and data format

```
rlReturnVal_t rlDeviceSetHsiConfig(rlUInt8_t deviceMap, rlDevHsiCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
AWR_RF_STATIC_CONF_SET_MSG
SB Id(s): AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB
AWR_DEV_RX_DATA_PATH_CONF_SET_SB
AWR_DEV_RX_DATA_PATH_CLK_SET_SB

10.3.2 rlDeviceSetLvdsConfig

This function sets the LVDS lane configuration

```
rlReturnVal_t rlDeviceSetLvdsConfig(rlUInt8_t deviceMap, rlDevLvdsCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AWR_DEV_LVDS_CFG_SET_SB

10.3.3 *rlDeviceSetCsi2Config*

This function sets the CSI-2 lane configuration

```
rlReturnVal_t rlDeviceSetCsi2Config(rlUInt8_t deviceMap, rlDevCsi2Cfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AWR_DEV_CSI2_CFG_SET_SB

10.3.4 *rlDeviceSetDataFmtConfig*

This function sets the LVDS/CSI2 data output format for the device.

```
rlReturnVal_t rlDeviceSetDataFmtConfig(rlUInt8_t deviceMap, rlDevDataFmtCfg_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AWR_DEV_RX_DATA_FORMAT_CONF_SET_SB

10.3.5 *rlDeviceSetDataPathConfig*

This function sets the LVDS/CSI2 data path configuration for the device.

```
rlReturnVal_t rlDeviceSetDataPathConfig(rlUInt8_t deviceMap,
rlDevDataPathCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AWR_DEV_RX_DATA_PATH_CONF_SET_SB

10.3.6 rlDeviceSetDataPathClkConfig

This function sets the LVDS/CSI2 data path clock configuration for the device.

```
rlReturnVal_t rlDeviceSetDataPathClkConfig(rlUInt8_t deviceMap,
rlDevDataPathClkCfg_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AWR_DEV_RX_DATA_PATH_CLK_SET_SB

10.3.7 rlDeviceSetLaneConfig

This function sets the LVDS/CSI2 lane configuration for the device.

```
rlReturnVal_t rlDeviceSetLaneConfig(rlUInt8_t deviceMap,  rlDevLaneEnable_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_DEV_CONF_SET_MSG
SB Id(s): AWR_DEV_RX_DATA_PATH_LANEEN_SET_SB

10.4 Safety Monitoring

TI mmWave device monitoring can be configured through a set of APIs defined in this section. Note that these APIs cover the RF/Analog related monitoring mechanisms. There are separate monitoring mechanisms for the digital logic (including the processor, memory, etc.) which are internal to the device and not explicitly enabled through these APIs.

The monitoring APIs are structured as follows:

- There are common configuration APIs that control the overall periodicity of monitoring, as well as, enable/disable control for each monitoring mechanism.
- For each monitoring mechanism there is an individual API to allow the customer to set an appropriate threshold for declaring failure from that monitoring.
- For each monitoring mechanism, there is an individual API to report soft (raw) values from that monitoring.
- The raw, periodic or failure Monitoring report is sent to application as asynchronous events.

10.4.1 rlRfDigMonEnableConfig

This function contains the consolidated configuration of all digital monitoring within the radar sub-system.

The enabled monitoring functions are executed when the API is issued. The scheduling of these monitoring should be handled in the application.

```
rlReturnVal_t rlRfDigMonEnableConfig(rlUInt8_t deviceMap, rlMonDigEnables_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_RF_DIG_LATENTFAULT_CONF_SB

10.4.2 rlRfDigMonPeriodicConfig

This function contains the consolidated configuration of all periodic digital monitoring within radar sub-system.

```
rlReturnVal_t rlRfDigMonPeriodicConfig(rlUInt8_t deviceMap,
rlDigMonPeriodicConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_RF_DIG_PERIODIC_CONF_SB

10.4.3 rlRfAnaMonConfig

This function contains the consolidated configuration of all analog monitoring. The enabled monitoring functions are executed with a periodicity of CAL_MON_TIME_UNITS (rlRfCalMonTimeUntConf_t.calibMonTimeUnit) number of logical frames. The host should ensure that all the enabled monitors can be completed in the available inter-frame times, based on the monitoring durations. The time taken for each monitoring is not defined in this document.

```
rlReturnVal_t rlRfAnaMonConfig(rlUInt8_t deviceMap, rlMonAnaEnables_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_ANALOG_ENABLES_CONF_SB

10.4.4 *rlRfTempMonConfig*

This API configures the on chip temperature monitors and report the soft results from the monitor. The corresponding monitors are collectively named TEMPERATURE_MONITOR. These monitors observe the temperature near various RF analog and digital modules using temperature sensors and GPADC and compare them against configurable thresholds. The report is sent as an async event AWR_MONITOR_TEMPERATURE_REPORT_AE_SB.

```
rlReturnVal_t rlRfTempMonConfig(rlUInt8_t deviceMap, rlTempMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_TEMPERATURE_CONF_SB

10.4.5 *rlRfRxGainPhMonConfig*

This function contains the configuration related to RX gain and phase monitoring.

```
rlReturnVal_t rlRfRxGainPhMonConfig(rlUInt8_t deviceMap,
rlRxGainPhaseMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_RX_GAIN_PHASE_CONF_SB

10.4.6 *rlRfRxNoiseMonConfig*

This function contains configuration related to RX noise figure monitoring.

```
rlReturnVal_t rlRfRxNoiseMonConfig(rlUInt8_t deviceMap, rlRxNoiseMonConf_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_RX_NOISE_FIGURE_CONF_SB

10.4.7 *rlRfRxIfStageMonConfig*

This function contains configuration related to RX IF filter stage monitoring.

```
rlReturnVal_t rlRfRxIfStageMonConfig(rlUInt8_t deviceMap, rlRxIfStageMonConf_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_RX_IFSTAGE_CONF_SB

10.4.8 *rlRfTxPowMonConfig*

This function contains configuration related to TX0, TX1 and TX2 power monitoring.

```
rlReturnVal_t rlRfTx0PowMonConfig(rlUInt8_t deviceMap, rlAllTxPowMonConf_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_TX0_POWER_CONF_SB
AWR_MONITOR_TX1_POWER_CONF_SB
AWR_MONITOR_TX2_POWER_CONF_SB

10.4.9 *rlRfTxBallbreakMonConfig*

This function contains configuration related to TX0, TX1 and TX2 ball break detection.

```
rlReturnVal_t rlRfTx0BallbreakMonConfig(rlUInt8_t deviceMap,
rlTxBallbreakMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_TX0_BALLBREAK_CONF_SB
AWR_MONITOR_TX1_BALLBREAK_CONF_SB
AWR_MONITOR_TX2_BALLBREAK_CONF_SB

10.4.10 *rlRfTxGainPhaseMismatchMonConfig*

This function contains configuration related to TX gain and phase mismatch monitoring.

```
rlReturnVal_t rlRfTxGainPhaseMismatchMonConfig(rlUInt8_t deviceMap,
rlTxGainPhaseMismatchMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_TX_GAIN_PHASE_MISMATCH_CONF_SB

10.4.11 *rlRfTxBpmMonConfig*

This function contains configuration related to TX0, TX1 and TX2 BPM monitoring.

```
rlReturnVal_t rlRfTxBpmMonConfig(rlUInt8_t deviceMap, rlAllTxBpmMonConf_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_TX0_BPM_CONF_SB
AWR_MONITOR_TX1_BPM_CONF_SB
AWR_MONITOR_TX2_BPM_CONF_SB

10.4.12 *rlRfSynthFreqMonConfig*

This function contains configuration related to synthesizer frequency monitoring during chirping.

```
rlReturnVal_t rlRfSynthFreqMonConfig(rlUInt8_t deviceMap, rlSynthFreqMonConf_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_SYNTHESIZER_FREQUENCY_CONF_SB

10.4.13 *rlRfExtAnaSignalsMonConfig*

This function contains configuration related to external DC signals monitoring (available only in xWR1642).

```
rlReturnVal_t rlRfExtAnaSignalsMonConfig(rlUInt8_t deviceMap,
rlExtAnaSignalsMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITORING_EXTERNAL_ANALOG_SIGNALS_CONF_SB

10.4.14 rlRfTxIntAnaSignalsMonConfig

This function contains configuration related to TX0, TX1 and TX2 Internal Analog Signals monitoring.

```
rlReturnVal_t rlRfTxIntAnaSignalsMonConfig(rlUInt8_t deviceMap,
rlAllTxIntAnaSignalsMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_CONF_SB
AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_CONF_SB
AWR_MONITOR_TX0_INTERNAL_ANALOG_SIGNALS_CONF_SB

10.4.15 rlRfRxIntAnaSignalsMonConfig

This function contains configuration related to RX Internal Analog Signals monitoring.

```
rlReturnVal_t rlRfRxIntAnaSignalsMonConfig(rlUInt8_t deviceMap,
rlRxIntAnaSignalsMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_RX_INTERNAL_ANALOG_SIGNALS_CONF_SB

10.4.16 rlRfPmClkLoIntAnaSignalsMonConfig

This function contains configuration related to Power Management, Clock generation and LO distribution circuits’ Internal Analog Signals monitoring.

```
rlReturnVal_t rlRfPmClkLoIntAnaSignalsMonConfig(rlUInt8_t deviceMap,
rlPmClkLoIntAnaSignalsMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_PMCLKLO_INTERNAL_ANALOG_SIGNALS_CONF_SB

10.4.17 rlRfGpadcIntAnaSignalsMonConfig

This function contains configuration related to GPADC Internal Analog Signals monitoring.

```
rlReturnVal_t rlRfGpadcIntAnaSignalsMonConfig(rlUInt8_t deviceMap,
rlGpadcIntAnaSignalsMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_GPADC_INTERNAL_ANALOG_SIGNALS_CONF_SB

10.4.18 rlRfPllContrlVoltMonConfig

This function contains configuration related to APLL and Synthesizer's control voltage signals monitoring

```
rlReturnVal_t rlRfPllContrlVoltMonConfig(rlUInt8_t deviceMap,
rlPllContrVoltMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_PLL_CONTROL_VOLTAGE_SIGNALS_CONF_SB

10.4.19 rlRfDualClkCompMonConfig

This function contains configuration related to the DCC based clock frequency monitoring.

```
rlReturnVal_t rlRfDualClkCompMonConfig(rlUInt8_t deviceMap,
rlDualClkCompMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_DUAL_CLOCK_COMP_CONF_SB

10.4.20 rlRfRxIfSatMonConfig

This function contains configuration related to RX saturation detector monitoring.

```
rlReturnVal_t rlRfRxIfSatMonConfig(rlUInt8_t deviceMap, rlRxSatMonConf_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_RX_SATURATION_DETECTOR_CONF_SB

10.4.21 rlRfRxSigImgMonConfig

This function contains configuration related to RX signal and image band energy.

```
rlReturnVal_t rlRfRxSigImgMonConfig(rlUInt8_t deviceMap, rlSigImgMonConf_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_SIG_IMG_MONITOR_CONF_SB

10.4.22 rlRfRxMixerInPwrConfig

This function contains configuration related to RX mixer input power monitor.

```
rlReturnVal_t rlRfRxMixerInPwrConfig(rlUInt8_t deviceMap,
rlRxMixInPwrMonConf_t *data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_MONITOR_RX_MIXER_IN_POWER_CONF_SB

10.4.23 rlRfAnaFaultInjConfig

This function is a fault injection configuration API.

NOTE - Not supported in current xWR1xxx Silicon.

```
rlReturnVal_t rlRfAnaFaultInjConfig(rlUInt8_t deviceMap, rlAnaFaultInj_t
*data)
```

The API invokes below message(s) and Sub block Id(s) to the Radar Device

MSG Id: AWR_RF_MONITORING_CONF_SET_MSG
SB Id(s): AWR_ANALOG_FAULT_INJECTION_CONF_SB

10.5 Platform Abstraction

The porting of the mmWaveLink driver to any new platform is based on few simple steps. This section takes you through this process step by step. Please follow the instructions carefully to avoid any problems during this process and to enable efficient and proper work with the device. Please notice that all modifications and porting adjustments of the driver should be made in the application only and driver should not be modified. Changes in the application file will ensure smoothly transaction to new versions of the driver at the future!

Step 1. *Define mmWaveLink client callback structure*: The mmWaveLink framework is ported to different platforms using mmWaveLink client callbacks. These callbacks are grouped as different structures such as OS callbacks, Communication Interface callbacks and others. Application needs to define these callbacks and initialize the mmWaveLink framework with the structure

Refer to rlClientCbs_t for interface details

Step 2. *Implement Communication Interface Callbacks*: The mmWave device supports several standard communication protocols among SPI and Mailbox. Depending on device variant, one needs to choose the communication channel. For e.g. xWR1443/xWR1642 requires Mailbox interface and AWR1243 supports SPI interface. The interface for this communication channel should include 4 simple access functions:

1. rlComIfOpen
2. rlComIfClose
3. rlComIfRead
4. rlComIfWrite

Refer to rlComIfCbs_t for interface details

Step 3. *Implement Device Control Interface*: The mmWaveLink driver internally powers on/off the AWR1xxx device. The exact implementation of these interfaces is platform dependent, hence one need to implement below functions:

1. rlDeviceEnable
2. rlDeviceDisable
3. rlRegisterInterruptHandler

Refer to rlDeviceCtrlCbs_t for interface details

Step 4. *Implement Event Handlers*: The mmWaveLink driver reports asynchronous events indicating AWR1xxx device status, exceptions etc. Application can register this callback to receive these notifications and take appropriate actions

Refer to rlEventCbs_t for interface details

Step 5. *Implement OS Interface*: The mmWaveLink driver can work in both OS and NonOS environment. If Application prefers to use operating system, it needs to implement basic OS routines such as tasks, mutex and Semaphore

Refer to rlOsiCbs_t for interface details

10.5.1 Platform Abstraction Callbacks

10.5.1.1 rlInt32_t (* rlComIfClose)(rlComIfHdl_t fd)

This API closes the communication interface.

Parameters:

In	*fd*	Handle to access the communication interface

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.2 rlComIfHdl_t (* rlComIfOpen)(rlUInt8_t deviceIndex, rlUInt32_t flags)

This API opens the communication interface (SPI/UART/Mailbox/I2C) with the AWR1xxx device.

Parameters:

In	*deviceIndex*	Device Index to identify the communication interface
In	*flags*	Flags to configure the interface **(reserved for future use)**

Returns:

rlComIfHdl_t Handle to access the communication interface

10.5.1.3 rlInt32_t (* rlComIfRead)(rlComIfHdl_t fd, rlUInt8_t *pBuff, rlUInt16_t len)

This API reads data from the communication interface.

Parameters:

In	*fd*	Handle to access the communication interface
In	*len*	Read size in bytes
Out	*pBuff*	Buffer to store data from communication interface

Returns:

rlInt32_t Length of received data

10.5.1.4 rlInt32_t (* rlComIfWrite)(rlComIfHdl_t fd, rlUInt8_t *pBuff, rlUInt16_t len)

This API writes data over the communication interface.

Parameters:

In	*fd*	Handle to access the communication interface
In	*pBuff*	Buffer containing data to write over communication interface
In	*len*	write size in bytes

Returns:

rlInt32_t Length of data sent

10.5.1.5 rlInt32_t (* rlComputeCRC)(rlInt8_t *data, rlUInt32_t dataLen, rlUInt8_t crcType, rlInt8_t *crc)

This API computes CRC (of a given CRC type) of the data for a given length.

Parameters:

In	*data*	Data buffer on which CRC needs to be computed
In	*dataLen*	Length of data in bytes
In	*crcType*	CRC type 0 – 16 bit CRC CCITT 1 – CRC-32 (used in Ethernet) 2 – CRC-64-ISO (HDLC)
Out	*crc*	Computed CRC

Returns:

rlInt32_t Success – 0, Failure – Error code

Note: Currently only 16 bit CRC CCITT is verified

10.5.1.6 rlInt32_t (* rlDeviceDisable)(rlUInt8_t deviceIndex)

This API powers off AWR1xxx device. Host should implement this function to de-assert the NRESET pin in AWR1xxx board.

Parameters:

In	*deviceIndex*	Device Index to identify the communication interface

Returns:

rlInt32_t Success – 0, Failure – Error code

10.5.1.7 rlInt32_t (* rlDeviceEnable)(rlUInt8_t deviceIndex)

This API brings AWR1xxx device out of reset. Host should implement this function to assert the NRESET pin in AWR1xxx board. Optionally it might require Sense on Power (SOP) pins to be asserted.

Parameters:

In	*deviceIndex*	Device Index to identify the communication interface

Returns:

rlInt32_t Success – 0, Failure – Error code

10.5.1.8 rlInt32_t(*rlDeviceWaitIrqStatus)(rlComIfHdl_t fd, int Level)

This callback polls the host Interrupt GPIO status. AWR1xxx device asserts host IRQ pin to get the host attention. After receiving the host interrupt, host polls the host interrupt status, low on host IRQ GPIO indicates that AWR1xxx has written data on communication interface.

Parameters:

In	*fd*	Handle to access the communication interface
In	*level*	Level of Host Irq to wait for

Returns:

rlInt32_t Success – 0, Failure – Error code

Note: This is currently reserved for future and is optional to implement

10.5.1.9 void (* rlDeviceMaskHostIrq)(rlComIfHdl_t fd)

This API masks the host interrupt. If GPIO interrupt is level triggered, host should mask the interrupt until the interrupt is serviced

Parameters:

In	*fd*	Handle to access the communication interface

10.5.1.10 void (* rlDeviceUnMaskHostIrq)(rlComIfHdl_t fd)

This API unmasks the host interrupt. If GPIO interrupt is level triggered, host should unmask the interrupt once interrupt is processed

Parameters:

In	*fd*	Handle to access the communication interface

10.5.1.11 rlInt32_t (* rlRegisterInterruptHandler)(rlUInt8_t deviceIndex, RL_P_EVENT_HANDLER pHandler, void *pValue)

This API registers the host interrupt handler. The application should store this function handler and invoke when it receives host interrupt. The application should enable the GPIO interrupt in this callback. Event handler callback does not process the interrupt in the same context so it is safe to call this handler from ISR directly.

Parameters:

In	*deviceIndex*	Device Index to identify source of Host Interrupt
In	*pHandler*	Interrupt Handler routine
In	*pValue*	To pass any additional data **(reserved for future use)**

Returns:

rlInt32_t Success – 0, Failure – Error code

10.5.1.12 void (*rlAsyncEvent)(rlUInt8_t deviceIndex, rlUInt16_t sbId, rlUInt16_t sbLen, rlUInt8_t *payload)

This API reports asynchronous events from radar device such as device status, exceptions etc**.**

Parameters:

In	*deviceIndex*	Device Index to identify source of event
In	*sbId*	RF Asynchronous event sub-blocks AWR_AE_RF_CPUFAULT_SB AWR_AE_RF_ESMFAULT_SB AWR_AE_RF_INITCALIBSTATUS_SB AWR_AE_RF_FRAME_TRIGGER_RDY_SB AWR_AE_RF_GPADC_RESULT_DATA_SB Master SS Asynchronous event sub-blocks AWR_AE_DEV_MSSPOWERUPDONE_SB AWR_AE_DEV_RFPOWERUPDONE_SB AWR_AE_MSS_CPUFAULT_SB AWR_AE_MSS_ESMFAULT_SB AWR_AE_MSS_BOOTERRORSTATUS_SB
In	*buf*	Event Data

In	*len*	Length of Event Data

10.5.1.13 rlInt32_t (* rlOsiMutexCreate)(rlOsiMutexHdl_t *mutexHdl, rlInt8_t *name)

This API creates a mutex object.

Parameters:

In	*name*	Name to associate with Mutex Object
Out	*mutexHdl*	Pointer to Mutex object

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.14 rlInt32_t (* rlOsiMutexDelete)(rlOsiMutexHdl_t *mutexHdl)

This API destroys a mutex object.

Parameters:

In	*mutexHdl*	Pointer to Mutex object

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.15 rlInt32_t (* rlOsiMutexLock)(rlOsiMutexHdl_t *mutexHdl, rlOsiTime_t timeout)

This API locks the mutex object.

Parameters:

In	*mutexHdl*	Pointer to Mutex object
In	*timeout*	Maximum Time to wait for Mutex Lock

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.16 rlInt32_t (* rlOsiMutexUnLock)(rlOsiMutexHdl_t *mutexHdl)

This API unlocks the mutex object.

Parameters:

In	*mutexHdl*	Pointer to Mutex object

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.17 rlInt32_t (* rlOsiSemCreate)(rlOsiSemHdl_t *semHdl, rlInt8_t *name)

This API creates a semaphore object.

Parameters:

In	*name*	Name to associate with Semaphore Object
Out	*semHdl*	Pointer to Semaphore object

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.18 rlInt32_t (* rlOsiSemDelete)(rlOsiSemHdl_t *semHdl)

This API destroys the semaphore object.

Parameters:

In	*semHdl*	Pointer to Semaphore object

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.19 rlInt32_t (* rlOsiSemSignal)(rlOsiSemHdl_t *semHdl)

This API releases the semaphore.

Parameters:

In	*semHdl*	Pointer to Semaphore object

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.20 rlInt32_t (* rlOsiSemWait)(rlOsiSemHdl_t *semHdl, rlOsiTime_t timeout)

This API waits for the semaphore.

Parameters:

In	*semHdl*	Pointer to Semaphore object
In	*timeout*	Maximum Time to wait for Semaphore

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.21 rlInt32_t (* rlOsiSpawn)(RL_P_OSI_SPAWN_ENTRY pEntry, void *pValue, rlUInt32_t flags)

This API calls a function in a different context. mmWaveLink Driver Interrupt handler function invokes this interface to switch context so that Interrupt Service Routine is executed immediately.

Parameters:

In	*pEntry*	Pointer to Entry Function
In	*pValue*	Pointer to data passed to function
In	*flags*	Flag to indicate preference **(for future use)**

Returns:

rlInt32_t Success - 0, Failure - Error code

10.5.1.22 rlInt32_t (*rlPrint) (const rlInt8_t* format, ...)

This API will print input message as per the format in the input arguments.

Parameters:

In	*format*	Formatted input message
In	*Variable number of argument*	Data which needs to be printed on console.

Returns:

rlInt32_t Success - Length of the message written in user's output console in bytes, Failure - Negative value

10.6 Data Type definitions

10.6.1 rlInt32_t

mmWaveLink integer Data type

Type	Name	Description	Units
int	rlInt32_t	mmWaveLink integer data type	-

10.6.2 rlUInt32_t

mmWaveLink unsigned integer Data type

Type	Name	Description	Units
Unsigned int	rlUInt32_t	mmWaveLink unsigned integer data type	-

10.6.3 rlInt16_t

mmWaveLink short Data type

Type	Name	Description	Units
short	rlInt16_t	mmWaveLink short data type	-

10.6.4 rlUInt16_t

mmWaveLink unsigned short data type

Type	Name	Description	Units
Unsigned short	rlUInt16_t	mmWaveLink unsigned short data type	-

10.6.5 rlInt8_t

mmWaveLink character data type

Type	Name	Description	Units
char	rlInt8_t	mmWaveLink character data type	-

10.6.6 rlUInt8_t

mmWaveLink Integer Data type

Type	Name	Description	Units
Unsigned char	rlUInt8_t	mmWaveLink unsigned char data type	-

10.6.7 rlReturnVal_t

mmWaveLink API return type

Type	Name	Description
rlInt32_t	rlReturnVal_t	mmWaveLink Error Codes #define RL_RET_CODE_OK (0) #define RL_RET_CODE_PROTOCOL_ERROR (-1) #define RL_RET_CODE_INVALID_INPUT (-2) #define RL_RET_CODE_SELF_ERROR (-3) #define RL_RET_CODE_RADAR_IF_ERROR (-4) #define RL_RET_CODE_MALLOC_ERROR (-5) #define RL_RET_CODE_CRC_FAILED (-6) #define RL_RET_CODE_CHKSUM_FAILED (-7) #define RL_RET_CODE_RESP_TIMEOUT (-8) #define RL_RET_CODE_FATAL_ERROR (-9) #define RL_RET_CODE_RADAR_OSIF_ERROR (-10) #define RL_RET_CODE_INVALID_STATE_ERROR (-11) #define RL_RET_CODE_API_NOT_SUPPORTED (-12) #define RL_RET_CODE_MSGID_MISMATCHED (-13) #define RL_RET_CODE_NULL_PTR (-14) #define RL_RET_CODE_INTERFACE_CB_NULL (-15) #define RL_RET_CODE_INVLD_OPCODE (1U) #define RL_RET_CODE_INVLD_NUM_SB (2U) #define RL_RET_CODE_INVLD_SB_ID (3U) #define RL_RET_CODE_INVLD_SB_LEN (4U) #define RL_RET_CODE_SB_INVL_DATA (5U) #define RL_RET_CODE_SB_PROCESS_ERR (6U) #define RL_RET_CODE_MISMATCH_FILE_CRC (7U) #define RL_RET_CODE_MISMATCH_FILE_TYPE (8U) #define RL_RET_CODE_FRAME_ALREADY_STARTED (20U) #define RL_RET_CODE_FRAME_ALREADY_ENDED (21U) #define RL_RET_CODE_FRAME_CFG_NOT_RECVD (22U) #define RL_RET_CODE_FRAME_TRIG_INVL_IN (23U) #define RL_RET_CODE_CH_CFG_RX_INVAL_IN (24U) #define RL_RET_CODE_CH_CFG_TX_INVAL_IN (25U) #define RL_RET_CODE_CH_CFG_CASC_INVAL_IN (26U) #define RL_RET_CODE_ADC_BITS_INVAL_IN (27U) #define RL_RET_CODE_ADC_FORM_INVAL_IN (28U) #define RL_RET_CODE_LP_ADC_INVAL_IN (29U) #define RL_RET_CODE_DYN_PS_INVAL_IN (30U) #define RL_RET_CODE_HSI_DIV_INVAL_IN (31U) #define RL_RET_CODE_RESERVED0 (32U)

		#define RL_RET_CODE_HSI_DIV_INVAL_1IN (33U) #define RL_RET_CODE_HSI_DIV_INVAL_2IN (34U) #define RL_RET_CODE_PF_IND_INVAL_IN (35U) #define RL_RET_CODE_PF_START_FREQ_INVAL_IN (36U) #define RL_RET_CODE_PF_IDLE_TIME_INVAL_IN (37U) #define RL_RET_CODE_PF_IDLE_TIME_1INVAL_IN (38U) #define RL_RET_CODE_PF_ADC_START_INVAL_IN (39U) #define RL_RET_CODE_PF_RAMP_END_INVAL_IN (40U) #define RL_RET_CODE_PF_RAMP_END_1INVAL_IN (41U) #define RL_RET_CODE_PF_TX0_INVAL_IN (42U) #define RL_RET_CODE_PF_TX1_INVAL_IN (43U) #define RL_RET_CODE_PF_TX2_INVAL_IN (44U) #define RL_RET_CODE_RESERVED1 (45U) #define RL_RET_CODE_PF_FREQ_SLOPE_1INVAL_IN (46U) #define RL_RET_CODE_PF_TX_START_INVAL_IN (47U) #define RL_RET_CODE_PF_NUM_ADC_SMAP_INVAL_IN (48U) #define RL_RET_CODE_PF_DFE_SAMP_RATE_INVAL_IN (49U) #define RL_RET_CODE_PF_HPF1_CF_INVAL_IN (50U) #define RL_RET_CODE_PF_HPF2_CF_INVAL_IN (51U) #define RL_RET_CODE_PF_RX_GAIN_INVAL_IN (52U) #define RL_RET_CODE_RESERVED2 (53U) #define RL_RET_CODE_RESERVED3 (54U) #define RL_RET_CODE_RESERVED4 (55U) #define RL_RET_CODE_RESERVED5 (56U) #define RL_RET_CODE_RESERVED6 (57U) #define RL_RET_CODE_RESERVED7 (58U) #define RL_RET_CODE_CHIRP_START_INVAL_IN (59U) #define RL_RET_CODE_CHIRP_END_INVAL_IN (60U) #define RL_RET_CODE_CHIRP_END_1INVAL_IN (61U) #define RL_RET_CODE_CHIRP_PF_IND_INVAL_IN (62U) #define RL_RET_CODE_CHIRP_PF_IND_1INVAL_IN (63U) #define RL_RET_CODE_CHIRP_START_FREQ_INVAL_IN (64U) #define RL_RET_CODE_CHIRP_SLOPE_INVAL_IN (65U) #define RL_RET_CODE_CHIRP_SLOPE_1INVAL_IN (66U) #define RL_RET_CODE_CHIRP_IDLE_TIME_INVAL_IN (67U) #define RL_RET_CODE_CHIRP_ADC_START_INVAL_IN (68U) #define RL_RET_CODE_CHIRP_ADC_START_1INVAL_IN

		(69U) #define RL_RET_CODE_CHIRP_TX_ENA_INVAL_IN (70U) #define RL_RET_CODE_CHIRP_TX_ENA_1INVAL_IN (71U) #define RL_RET_CODE_FRAME_CHIRP_STR_INVAL_IN (72U) #define RL_RET_CODE_FRAME_CHIRP_END_INVAL_IN (73U) #define RL_RET_CODE_FRAME_CHIRP_END_1INVAL_IN (74U) #define RL_RET_CODE_FRAME_CHIRP_END_2INVAL_IN (75U) #define RL_RET_CODE_FRAME_CHIRP_PF_INVAL_IN (76U) #define RL_RET_CODE_FRAME_CHIRP_LOOPS_INVAL_IN (77U) #define RL_RET_CODE_RESERVED8 (78U) #define RL_RET_CODE_FRAME_PERIOD_INVAL_IN (79U) #define RL_RET_CODE_FRAME_PERIOD_1INVAL_IN (80U) #define RL_RET_CODE_FRAME_TRIG_SEL_INVAL_IN (81U) #define RL_RET_CODE_FRAME_TRIG_DELAY_INVAL_IN (82U) #define RL_RET_CODE_FRAME_IS_ONGOING (83U) #define RL_RET_CODE_AFRAME_NUM_SUBF_INVAL_IN (84U) #define RL_RET_CODE_AFRAME_FORCE_PF_INVAL_IN (85U) #define RL_RET_CODE_AFRAME_PF_IND_INVAL_IN (86U) #define RL_RET_CODE_AFRAME_PF_IND_1INVAL_IN (87U) #define RL_RET_CODE_AFRAME_CHIRP_STR_INVAL_IN (88U) #define RL_RET_CODE_AFRAME_NCHIRP_INVAL_IN (89U) #define RL_RET_CODE_AFRAME_NCHIRP_1INVAL_IN (90U) #define RL_RET_CODE_AFRAME_CHIRP_PF_INVAL_IN (91U) #define RL_RET_CODE_AFRAME_CHIRP_LOOPS_INVAL_IN (92U) #define RL_RET_CODE_AFRAME_BURST_PERIOD_INVAL_IN (93U) #define RL_RET_CODE_AFRAME_BURST_PER_1INVAL_IN (94U) #define RL_RET_CODE_AFRAME_BURST_STIND_INVAL_IN (95U) #define RL_RET_CODE_AFRAME_BURST_SIND_1INVAL_IN (96U) #define RL_RET_CODE_AFRAME_NUM_BURSTS_INVAL_IN (97U) #define RL_RET_CODE_AFRAME_BURST_LOOPS_INVAL_IN

		(98U) #define RL_RET_CODE_AFRAME_SF_PERIOD_INVAL_IN (99U) #define RL_RET_CODE_AFRAME_SF_PERIOD_1INVAL_IN (100U) #define RL_RET_CODE_RESERVED9 (101U) #define RL_RET_CODE_AFRAME_TRIG_SEL_INVAL_IN (102U) #define RL_RET_CODE_AFRAME_TRIG_DELAY_INVAL_IN (103U) #define RL_RET_CODE_AFRAME_IS_ONGOING (104U) #define RL_RET_CODE_TS_POS_VECY_INVAL_IN (105U) #define RL_RET_CODE_RESERVED10 (106U) #define RL_RET_CODE_TS_VEL_VECXYZ_INVAL_IN (107U) #define RL_RET_CODE_TS_SIG_LEVEL_INVAL_IN (108U) #define RL_RET_CODE_TS_RX_ANT_POS_INVAL_IN (109U) #define RL_RET_CODE_RESERVED11 (110U) #define RL_RET_CODE_PROG_FILT_STARTINDX_INVALID (111U) #define RL_RET_CODE_PROG_FILT_PROFILE_INVALID (112U) #define RL_RET_CODE_PROG_FILT_UNSUPPORTED_DEV (113U) #define RL_RET_CODE_PERCHIRPPHSHIFT_UNSUPPORTED_DEV (114U) #define RL_RET_CODE_PERCHIRPPHSHIFT_STIND (115U) #define RL_RET_CODE_PERCHIRPPHSHIFT_ENIND (116U) #define RL_RET_CODE_PERCHIRPPHSHIFT_WRONG_STIND (117U) #define RL_RET_CODE_RF_INIT_NOT_DONE (118U) #define RL_RET_CODE_FREQ_LIMIT_OUT_RANGE (119U) #define RL_RET_CODE_CAL_MON_TIME_INVALID (120U) #define RL_RET_CODE_RUN_CAL_PERIOD_INVALID (121U) #define RL_RET_CODE_CONT_STREAM_MODE_EN (122U) #define RL_RET_CODE_RX_GAIN_BOOT_CAL_NOT_DONE (123U) #define RL_RET_CODE_LO_DIST_BOOT_CAL_NOT_DONE (124U) #define RL_RET_CODE_TX_PWR_BOOT_CAL_NOT_DONE (125U) #define RL_RET_CODE_PROG_FILTR_UNSUPPORTED_DFEMODE (126U) #define RL_RET_CODE_ADC_BITS_FULL_SCALE_REDUC_INVAL (127U)

		#define RL_RET_CODE_CH_CFG_DEV_VRNT_CASC_INVAL_IN (130U) #define RL_RET_CODE_INVAL_LOOPBACK_TYPE (132U) #define RL_RET_CODE_INVAL_LOOPBACK_BURST_IND (133U) #define RL_RET_CODE_INVAL_LOOPBACK_CONFIG (134U) #define RL_RET_CODE_DYN_CHIRP_INVAL_SEG (135U) #define RL_RET_CODE_DYN_PERCHIRP_PHSHFT_INVA_SEG (136U) #define RL_RET_CODE_INVALID_CAL_CHUNK_ID (137U) #define RL_RET_CODE_INVALID_CAL_CHUNK_DATA (138U) #define RL_RET_CODE_RX02_RF_TURN_OFF_TIME_INVALID (139U) #define RL_RET_CODE_RX13_RF_TURN_OFF_TIME_INVALID (140U) #define RL_RET_CODE_RX02_BB_TURN_OFF_TIME_INVALID (141U) #define RL_RET_CODE_RX13_BB_TURN_OFF_TIME_INVALID (142U) #define RL_RET_CODE_RX02_RF_PREENABLE_TIME_INVALID (143U) #define RL_RET_CODE_RX13_RF_PREENABLE_TIME_INVALID (144U) #define RL_RET_CODE_RX02_BB_PREENABLE_TIME_INVALID (145U) #define RL_RET_CODE_RX13_BB_PREENABLE_TIME_INVALID (146U) #define RL_RET_CODE_RX02_RF_TURN_ON_TIME_INVALID (147U) #define RL_RET_CODE_RX13_RF_TURN_ON_TIME_INVALID (148U) #define RL_RET_CODE_RX02_BB_TURN_ON_TIME_INVALID (149U) #define RL_RET_CODE_RX13_BB_TURN_ON_TIME_INVALID (150U) #define RL_RET_CODE_RX_LO_TURN_OFF_TIME_INVALID (151U) #define RL_RET_CODE_TX_LO_TURN_OFF_TIME_INVALID (152U) #define RL_RET_CODE_RX_LO_TURN_ON_TIME_INVALID (153U) #define RL_RET_CODE_TX_LO_TURN_ON_TIME_INVALID (154U) #define RL_RET_CODE_DEVICE_NOT_ASILB_TYPE (250U) #define RL_RET_CODE_FRAME_ONGOING (251U) #define RL_RET_CODE_INVLD_REPO_MODE (252U) #define RL_RET_CODE_INVLD_PROFILE_ID (253U) #define RL_RET_CODE_INVLD_PROFILE (254U) #define RL_RET_CODE_INVLD_EXTSIG_SETLTIME (255U) #define RL_RET_CODE_INVLD_NO_RX_ENABLED (256U) #define RL_RET_CODE_INVLD_TX0_NOT_ENABLED

```
       (257U)
#define RL_RET_CODE_INVLD_TX1_NOT_ENABLED
       (258U)
#define RL_RET_CODE_INVLD_TX2_NOT_ENABLED
       (259U)
#define RL_RET_CODE_MON_INVALID_RF_BIT_MASK
       (260U)
#define RL_RET_CODE_RESERVED12    (261U)
#define RL_RET_CODE_RESERVED13    (262U)
#define RL_RET_CODE_MON_TX_EN_CHK_FAIL  (263U)
#define RL_RET_CODE_MON_RX_CH_EN_CHK_FAIL
       (264U)
#define RL_RET_CODE_MON_TX_CH_PS_LB     (265U)
#define RL_RET_CODE_INVLD_SAT_MON_SEL   (266U)
#define RL_RET_CODE_INVLD_SAT_MON_PRI_SLICE_DUR
       (267U)
#define RL_RET_CODE_INVLD_SAT_MON_NUM_SLICES
       (268U)
#define RL_RET_CODE_INVLD_SIG_IMG_SLICENUM
       (269U)
#define RL_RET_CODE_INVLD_SIG_IMG_NUMSAMPPERSLICE
       (270U)
#define RL_RET_CODE_INVLD_SYNTH_L1_LIN  (271U)
#define RL_RET_CODE_INVLD_SYNTH_L2_LIN  (272U)
#define RL_RET_CODE_INVLD_SYNTH_N_LIN   (273U)
#define RL_RET_CODE_INVLD_SYNTH_MON_START_TIME
       (274U)
#define RL_RET_CODE_INVLD_SYNTH_MON_LIN_RAM_ADDR
       (275U)
#define RL_RET_CODE_LDO_BYPASSED (279U)
#define RL_RET_CODE_INVLD_SIG_IMG_BAND_MONTR
       (280U)
#define RL_RET_CODE_RX_CHAN_EN_OOR      (1001U)
#define RL_RET_CODE_NUM_ADC_BITS_OOR    (1002U)
#define RL_RET_CODE_ADC_OUT_FMT_OOR     (1003U)
#define RL_RET_CODE_IQ_SWAP_SEL_OOR     (1004U)
#define RL_RET_CODE_CHAN_INTERLEAVE_OOR (1005U)
#define RL_RET_CODE_DATA_INTF_SEL_OOR   (1006U)
#define RL_RET_CODE_DATA_FMT_PKT0_INVALID
       (1007U)
#define RL_RET_CODE_DATA_FMT_PKT1_INVALID
       (1008U)
#define RL_RET_CODE_LANE_ENABLE_OOR     (1009U)
#define RL_RET_CODE_LANE_ENABLE_INVALID (1010U)
#define RL_RET_CODE_LANE_CLK_CFG_OOR    (1011U)
#define RL_RET_CODE_LANE_CLK_CFG_INVALID
       (1012U)
#define RL_RET_CODE_DATA_RATE_OOR (1013U)
#define RL_RET_CODE_LANE_FMT_MAP_OOR    (1014U)
#define RL_RET_CODE_LANE_PARAM_CFG_OOR  (1015U)
#define RL_RET_CODE_CONT_STREAM_MODE_OOR
       (1016U)
#define RL_RET_CODE_CONT_STREAM_MODE_INVALID
       (1017U)
#define RL_RET_CODE_LANE0_POS_POL_OOR   (1018U)
```

		#define RL_RET_CODE_LANE1_POS_POL_OOR (1019U) #define RL_RET_CODE_LANE2_POS_POL_OOR (1020U) #define RL_RET_CODE_LANE3_POS_POL_OOR (1021U) #define RL_RET_CODE_CLOCK_POS_OOR (1022U) #define RL_RET_CODE_HALF_WORDS_PER_CHIRP_OOR (1023U) #define RL_RET_CODE_NUM_SUBFRAMES_OOR (1024U) #define RL_RET_CODE_SF1_TOT_NUM_CHIRPS_OOR (1025U) #define RL_RET_CODE_SF1_NUM_ADC_SAMP_OOR (1026U) #define RL_RET_CODE_SF1_NUM_CHIRPS_OOR (1027U) #define RL_RET_CODE_SF2_TOT_NUM_CHIRPS_OOR (1028U) #define RL_RET_CODE_SF2_NUM_ADC_SAMP_OOR (1029U) #define RL_RET_CODE_SF2_NUM_CHIRPS_OOR (1030U) #define RL_RET_CODE_SF3_TOT_NUM_CHIRPS_OOR (1031U) #define RL_RET_CODE_SF3_NUM_ADC_SAMP_OOR (1032U) #define RL_RET_CODE_SF3_NUM_CHIRPS_OOR (1033U) #define RL_RET_CODE_SF4_TOT_NUM_CHIRPS_OOR (1034U) #define RL_RET_CODE_SF4_NUM_ADC_SAMP_OOR (1035U) #define RL_RET_CODE_SF4_NUM_CHIRPS_OOR (1036U) #define RL_RET_CODE_MCUCLOCK_CTRL_OOR (1040U) #define RL_RET_CODE_MCUCLOCK_SRC_OOR (1041U) #define RL_RET_CODE_PMICCLOCK_CTRL_OOR (1042U) #define RL_RET_CODE_PMICCLOCK_SRC_OOR (1043U) #define RL_RET_CODE_PMICMODE_SELECT_OOR (1044U) #define RL_RET_CODE_PMICFREQ_SLOPE_OOR (1045U) #define RL_RET_CODE_PMICCLK_DITHER_EN_OOR (1046U) #define RL_RET_CODE_TESTPATTERN_EN_OOR (1047U) #define RL_RET_CODE_LFAULTTEST_UNSUPPORTED_OOR (1048U)

10.6.8 RL_P_OSI_SPAWN_ENTRY

mmWaveLink Spawn task entry function

Type	Name	Description	Units
void (*RL_P_OSI_SPAWN_ENTRY)(void* pValue)	RL_P_OSI_SPAWN_ENTRY	mmWaveLink Spawn task entry function	-

10.6.9 RL_P_EVENT_HANDLER

mmWaveLink Event Handler callback

Type	Name	Description	Units
void (*RL_P_EVENT_HANDLER)(rlUInt8_t deviceIndex, void* pValue)	RL_P_EVENT_HANDLER	mmWaveLink Event Handler callback	-

10.6.10 rlComIfHdl_t

Communication Interface Handle

Type	Name	Description	Units
rlInt32_t* rlComIfHdl_t	rlComIfHdl_t	Communication Interface Handle	-

10.6.11 rlOsiSemHdl_t

OS Semaphore Object Handle

Type	Name	Description	Units
void* rlOsiSemHdl_t	rlOsiSemHdl_t	OS Semaphore Object Handle	-

10.6.12 rlOsiMutexHdl_t

OS Mutex Object Handle

Type	Name	Description	Units
void* rlOsiSemHdl_t	rlOsiSemHdl_t	OS Mutex Object Handle	-

10.6.13 rlOsiTime_t

OS Time data type

Type	Name	Description	Units
rlUInt32_t rlOsiTime_t	rlOsiTime_t	OS Time data type	-

10.6.14 rlCrcType_t

CRC type

Type	Name	Description	Units
rlUInt32_t rlCrcType_t	rlCrcType_t	CRC type 0 – 16 bit CRC CCITT 1 – CRC-32 (used in Ethernet) 2 – CRC-64-ISO (HDLC) 3 – CRC disabled	-

10.6.15 *rlPrintFptr*

Callback to print user messages/logs

Type	Name	Description	Units
rlInt32_t (*rlPrint)(const rlInt8_t* format, ...)	rlPrintFptr	mmWaveLink print callback function prototype	-

10.6.16 *rlClientCbs_t*

This structure is the container for the callback to be registered to use various services and invoke certain asynchronous events.

struct rlClientCbs {

Type	Name	Description	Units
rlComIfCbs_t	comIfCb	Structure containing the function pointers to be used for communication channel(SPI, Mailbox) interface. Supports the open, read, write, close functionality of the communication interface driver.	-
rlOsiCbs_t	osiCb	Structure containing the function pointers to use the OS services like semaphores, mutex and message queues.	-
rlEventCbs_t	eventCb	Structure containing function pointer that would be invoked in case of any async events	

rlDeviceCtrlCbs_t	devCtrlCb	Structure containing function pointers to the device control interface function like SPI IRQ mask/unmask, get IRQ status, register interrupt handler.	
rlTimerCbs_t	timerCb	Structure containing the function pointer to use the timer services like start/stop timer, delay etc.	
rlCrcCbs_t	crcCb	Structure contains the function pointer to perform the CRC computation. This callback is optional if crcType is set to 0(No CRC)	
rlCrcType_t	crcType	CRC Type to be used with this. 0 – 16 bit CRC CCITT 1 – CRC-32 (used in Ethernet) 2 – CRC-64-ISO (HDLC) 3 – CRC disabled	
rlUInt32_t	ackTimeout	ACK timeout in milliseconds. 0 – indicates no ACK.	milliseconds
rlUInt8_t	platform	0x0: mmWaveLink runs on Ext Host 0x1: mmWaveLink runs on MSS, 0x2: mmWaveLink runs on DSS	

rlUInt8_t	arDevType	AWR12xx = 0x0,
		AWR14xx = 0x1,
		AWR16xx = 0x2	

rlDbgCb_t	dbgCb	Structure containing the function pointer to print user messages/logs and level of debugging. 0: None, 1:Error, 2: Warning 3:Info 4: Debug 5:Verbose	

};

10.6.17 rlComIfCbs_t

This structure is the container for the communication interface callback functions

struct rlComIfCbs {

Type	Name	Description	Units
rlComIfHdl_t (* rlComIfOpen) (rlInt8_t deviceIndex, rlUInt32_t flags)	rlComIfOpen	Open Communication interface (SPI, UART, Mailbox etc) with Radar device	-
rlInt32_t (* rlComIfRead) (rlComIfHdl_t fd, rlUInt8_t *pBuff, rlUInt32_t len)	rlComIfRead	Read data from Communication Interface	-
rlInt32_t (* rlComIfWrite) (rlComIfHdl_t fd, rlUInt8_t *pBuff, rlUInt32_t len)	rlComIfWrite	Write data on Communication Interface	-
rlInt32_t (* rlComIfClose) (rlComIfHdl_t fd)	rlComIfClose	Close Communication Interface	

};

10.6.18 rlDeviceCtrlCbs_t

This structure is the container for the device interface callback functions

struct rlDeviceCtrlCbs {

Type	Name	Description	Units
rlInt32_t (* rlDeviceEnable) (rlUInt8_t deviceIndex)	rlDeviceEnable	Bring AWR1xxx device out of Reset	-
rlInt32_t (* rlDeviceDisable) (rlUInt8_t deviceIndex)	rlDeviceDisable	Power down AWR1xxx device	-
void (*rlDeviceMaskHostIrq) (rlComIfHdl_t fd)	rlDeviceMaskHostIrq	Masks Host Interrupt	-
void (*rlDeviceUnMaskHostIrq) (rlComIfHdl_t fd)	rlDeviceUnMaskHostIrq	Unmasks Host Interrupt	
rlInt32_t (*rlDeviceHostIrqStatus) (rlComIfHdl_t fd, int Level)	rlDeviceHostIrqStatus	Polls Host Interrupt Status	
rlInt32_t (* rlRegisterInterruptHandler) (RL_P_EVENT_HANDLER pHandler, void *pValue)	rlRegisterInterruptHandler	Register host interrupt handler	

```
};
```

10.6.19 rlOsiMutexCbs_t

This structure is the container for the operating system mutex interface callback functions

struct rlOsiMutexCbs {

Type	Name	Description	Units
rlInt32_t(* rlOsiMutexCreate)(rlOsiMutexHdl_t *mutexHdl, rlInt8_t *name)	rlOsiMutexCreate	Create Mutex Object	-
rlInt32_t(* rlOsiMutexLock)(rlOsiMutexHdl_t *mutexHdl, rlOsiTime_t timeout)	rlOsiMutexLock	Lock Mutex Object	-
rlInt32_t(* rlOsiMutexUnLock)(rlOsiMutexHdl_t *mutexHdl)	rlOsiMutexUnLock	Unlock Mutex Object	-
rlInt32_t(* rlOsiMutexDelete)(rlOsiMutexHdl_t *mutexHdl)	rlOsiMutexDelete	Delete Mutex Object	

};

10.6.20 rlOsiSemCbs_t

This structure is the container for the operating system semaphore interface callback functions

struct rlOsiSemCbs {

Type	Name	Description	Units
rlInt32_t(* rlOsiSemCreate)(rlOsiSemHdl_t *semHdl, rlInt8_t *name)	rlOsiSemCreate	Create semaphore Object	-
rlInt32_t(* rlOsiSemWait)(rlOsiSemHdl_t *semHdl, rlOsiTime_t timeout)	rlOsiSemWait	Wait for semaphore object	-
rlInt32_t(* rlOsiSemSignal)(rlOsiSemHdl_t *semHdl)	rlOsiSemSignal	Release semaphore	-
rlInt32_t(* rlOsiSemDelete)(rlOsiSemHdl_t *semHdl)	rlOsiSemDelete	Delete semaphore Object	

};

10.6.21 rlOsiMsgQCbs_t

This structure is the container for the operating system message queue interface callback functions

struct rlOsiMsgQCbs {

Type	Name	Description	Units
rlInt32_t(* rlOsiSpawn)(RL_P_OSI_SPAWN_ENTRY pEntry, void *pValue, rlUInt32_t flags)	rlOsiSpawn	Calls a function in different context	-

};

10.6.22 rlOsiCbs_t

This structure is the container for the operating system interface callback functions

struct rlOsiMutexCbs {

Type	Name	Description	Units
rlOsiMutexCbs_t	mutex	OS Mutex interface	-
rlOsiSemCbs_t	sem	OS Semaphore interface	-
rlOsiMsgQCbs_t	queue	OS Queue interface	-

};

10.6.23 rlEventCbs_t

This structure is the container for the asynchronous event handler callback functions

struct rlEventCbs {

Type	Name	Description	Units
void(* rlAsyncEvent)((rlUInt8_t devIndex, rlUInt16_t sbId, rlUInt16_t sbLen, rlUInt8_t *payload)	rlAsyncEvent	Callback to receive asynchronous events	-

};

10.6.24 rlCrcCbs_t

This structure is the container for the CRC calculation callback functions

Struct rlCrcCbs {

Type	Name	Description	Units
rlInt32_t(*rlComputeCRC)(rlInt8_t *data, rlUInt32_t dataLen,	rlComputeCRC	Compute CRC	-

rlUInt8_t crcType, rlInt8_t *crc)			

};

10.6.25 rlTimerCbs_t

This structure is the container for the timer callback functions

struct rlTimerCbs {

Type	Name	Description	Units
rlInt32_t(* rlDelay)(rlUInt32_t delay)	rlDelay	Delay function	

};

10.6.26 rlDbgCb_t

This structure is the container for the callback function of print and level of debugging.

struct rlDbgCb {

Type	Name	Description	Units
rlPrintFptr	rlPrint	Print function	
rlUInt8_t	dbgLevel	Level of debugging	

};

10.6.27 rlFileData_t

This structure is the container for the file chunk size and payload

struct rlFileData {

Type	Name	Description	Units
rlUInt32_t	chunkLen	Length of the file/data to be sent to device	-
rlUInt8_t	fData[]	Data to be sent to the device.	-

};

10.6.28 rlVersion_t

This structure is the container for the version number information.

struct rlVersion {

Type	Name	Description	Units
rlFwVersionParam_t	master	Master Sub System version	-

rlFwVersionParam_t	rf	RF Sub System version	-
rlSwVersionParam_t	mmWaveLink	mmWaveLink version	

```
};
```

10.6.29 rlFwVersionParam_t

This structure is the container for the device firmware version information.

```
struct rlFwVersionParam {
```

Type	Name	Description	Units
rlUInt8_t	hwVarient	HW variant number	-
rlUInt8_t	hwMajor	HW version major number	-
rlUInt8_t	hwMinor	HW version minor number	-
rlUInt8_t	fwMajor	Firmware major version number	-
rlUInt8_t	fwMinor	Firmware minor version number	-
rlUInt8_t	fwBuild	Firmware build number	-
rlUInt8_t	fwDebug	Firmware debug version number	-
rlUInt8_t	fwYear	Year of Firmware release	-
rlUInt8_t	fwMonth	Month of the firmware release	-
rlUInt8_t	fwDay	Day of the firmware release	-
rlUInt8_t	patchMajor	ROM Patch major version number	-
rlUInt8_t	patchMinor	ROM Patch minor version number	-
rlUInt8_t	patchYear	Year of ROM patch release	-
rlUInt8_t	patchMonth	Month of the ROM patch release	-
rlUInt8_t	patchDay	Day of the ROM patch release	-
rlUInt8_t	reserved	Reserved	0

```
};
```

10.6.30 rlSwVersionParam_t

This structure is the container for the device firmware version information.

```
struct rlSwVersionParam {
```

Type	Name	Description	Units
rlUInt8_t	major	Software major version number	-
rlUInt8_t	minor	Software minor version number	-
rlUInt8_t	build	Software build number	-
rlUInt8_t	debug	Software debug version number	-
rlUInt8_t	year	Year of Firmware release	-

rlUInt8_t	month	Month of the Software release	-
rlUInt8_t	day	Day of the Software release	-
rlUInt8_t	reserved	Reserved	0

`};`

10.6.31 *rlGpAdcData_t*

This structure is the container of GPADC data.

`struct rlGpAdcData {`

Type	Name	Description	Units
rlUInt16_t	min	Min value of GP ADC data	-
rlUInt16_t	max	Max value of GP ADC data	-
rlUInt16_t	avg	Average value of GP ADC data	

`};`

10.6.32 *rlRecvdGpAdcData_t*

This structure is the container of GPADC data for different sensors.

`struct rlGpAdcData {`

Type	Name	Description	Units
rlGpAdcData_t	Sensor[12]	Contain all sensors GPADC data	-

`};`

10.6.33 *rlChanCfg_t*

This structure is the container for the TX/RX channel to enable or disable

`struct rlChanCfg {`

Type	Name	Description	Units
rlUInt16_t	rxChannelEn	RX Channel Enable/disable b0: 0 Disable RX channel 0; 1 Enable RX channel 0 b1: 0 Disable RX channel 1; 1 Enable RX channel 1 b2: 0 Disable RX channel 2; 1 Enable RX channel 2	-

		b3: 0 Disable RX channel 3; 1 Enable RX channel 3 b15:4: Reserved	
rlUInt16_t	txChannelEn	TX Channel Enable/disable b0: 0 Disable TX channel 0; 1 Enable TX channel 0 b1: 0 Disable TX channel 1; 1 Enable RX channel 1 b2: 0 Disable TX channel 2; 1 Enable TX channel 2 b15:3: Reserved	-
rlUInt16_t	cascading	Cascading enable 0 – Single Chip 1 – Multichip Master 2 – Multichip Slave	-
rlUInt16_t	bReserved	Reserved	0

};

10.6.34 rlAdcBitFormat_t

This structure is the container for the data format of the ADC output.

union rlAdcBitFormat {

Type	Name	Description	Units
struct bitFormat {			
rlUInt32_t	b2AdcBits:2	Number of ADC bits 00: 12 bits 01: 14 bits 10: 16 bits Other: Reserved	-
rlUInt32_t	b14Reserved1:14	Reserved	0
rlUInt32_t	b2AdcOutFmt:2	ADC output Format 00: Real 01: Complex 1x(image band filtered output) 10: Complex 2x(Image band visible) 11: Pseudo Real	
rlUInt32_t	b14Reserved2:14	Reserved	0
};			
rlUInt32_t	b32Value		

};

10.6.35 rlAdcOutCfg_t

This structure is the container for the data format of the ADC output.

struct rlAdcOutCfg {

Type	Name	Description	Units
rlAdcBitFormat_t	Fmt	ADC data format	-
rlUInt16_t	reserved	Reserved	0
rlUInt16_t	reserved1	Reserved	0

};

10.6.36 rlLowPowerModeCfg_t

This structure is the container for the low power mode configuration

struct rlLowPowerModeCfg {

Type	Name	Description	Units
rlUInt16_t	anaChannelCfg	Analog Chain Configuration 0x00 : Complex Chain 0x01 : Real Chain	-
rlUInt16_t	lpAdcMode	ADC Mode 0x00 : Regular ADC mode 0x01 : Low Power ADC mode	-

};

10.6.37 rlProfileCfg_t

This structure is the container for the FMCW radar profile properties

struct rlProfileCfg {

Type	Name	Description	Units
rlUInt16_t	profileId	Profile index for which rest of the properties are applicable	-
rlUInt8_t	pfVcoSelect	Bit Description b0 FORCE_VCO_SEL 0 - Use internal VCO selection 1 - Forced external VCO selection b1 VCO_SEL 0 - VCO1 (76 - 78 GHz) 1 - VCO2 (77 - 81 GHz) b7:2 RESERVED	-
rlUInt8_t	pfCalLutUpdate	Bit Description b0 RETAIN_TXCAL_LUT 0 - Update TX calibration LUT 1 - Do not update TX	

		calibration LUT b1 RETAIN_RXCAL_LUT 0 - Update RX calibration LUT 1 - Do not update TX calibration LUT b7:2 RESERVED If PF_TX_OUTPUT_POWER_BACKOFF is changed then set RETAIN_TXCAL_LUT to 0, else set it to 1 and if PF_RX_GAIN is changed, then set RETAIN_RXCAL_LUT to 0 else set them to 1.	
rlUInt32_t	startFreqConst	Start frequency for each profile 1 LSB = 3.6e9/2^26 = ~53.644 Hz	Hz
rlUInt32_t	idleTimeConst	Idle time for each profile 1 LSB = 10 ns	ns
rlUInt32_t	adcStartTimeConst	ADC capture time for each profile. 1 LSB = 10 ns	ns
rlUInt32_t	rampEndTime	Ramp end of each profile. 1 LSB = 10ns	ns
rlUInt32_t	txOutPowerBackoffCode	TX channel output power backoff. b7:0- TX0 output power backoff b15:8-TX1 output power backoff b23:16-TX2 output power backoff. b31:24 - Reserved 1 LSB = 1 dB	dB
rlUInt32_t	txPhaseShifter	Concatenated phase shift for TX0/1/2, b1:0 RESERVED (set it to 0b00) b7:2 TX0 phase shift value b9:8 RESERVED (set it to 0b00) b15:10 TX1 phase shift value b17:16 RESERVED (set it to 0b00) b23:18 TX2 phase shift value b31:24 Reserved 1 LSB = 360/2^6 = 5.6 degrees	degree
rlInt16_t	freqSlopeConst	Frequency slope for each profile 1LSB = 3.6e9*900/2^26	kHz/µs

		= 48.279 kHz/µs	
rlInt16_t	txStartTime	TX start time 1 LSB = 10ns	ns
rlUInt16_t	numAdcSamples	Number of ADC samples to capture in a chirp for each RX Valid range: 64 to MAX_NUM_SAMPLES, where MAX_NUM_SAMPLES is such that all the enabled RX channels' data fits into 16 kB memory, with each sample consuming 2 bytes for real ADC output case and 4 bytes for complex 1x and complex 2x ADC output cases. For example: 4 RX, Complex ADC output-> MAX_NUM_SAMPLES = 1024\n 4 RX, Real ADC output-> MAX_NUM_SAMPLES = 2048\n 2 RX, Complex ADC output-> MAX_NUM_SAMPLES = 2048\n 2 RX, Real ADC output-> MAX_NUM_SAMPLES = 4096	-
rlUInt16_t	digOutSampleRate	ADC sampling rate 1 LSB = 1 ksps	ksps
rlUInt8_t	hpfCornerFreq1	HPF1 corner frequency for each profile 0x00 - 175kHz 0x01 - 235kHz 0x02 - 350kHz 0x03 - 700kHz	-
rlUInt8_t	hpfCornerFreq2	HPF2 corner frequency for each profile 0x00 - 350kHz 0x01 - 700kHz 0x02 - 1.4MHz 0x03 - 2.8MHz	-
rlUInt16_t	Reserved0	Reserved	0
rlUInt8_t	rxGain	RX gain for each profile 1LSB = 1dB	dB
rlUInt16_t	Reserved1		0

```
};
```

10.6.38 rlChirpCfg_t

This structure is the container for the chirp to chirp variations applied on top of chirp profile.

struct rlChirpCfg {

Type	Name	Description	Units
rlUInt16_t	chirpStartIdx	Chirp start index valid range 0 to 511	-
rlUInt16_t	chirpEndIdx	Chirp start index valid range chirpStartIdx to 511	-
rlUInt16_t	profileId	Profile index valid range 0 to 3	-
rlUInt16_t	reserved	Reserved	0
rlUInt32_t	startFreqVar	Ramp start frequency variation 1 LSB = 3.6e9/2^26 = 54.644 Hz	Hz
rlInt16_t	freqSlopeVar	Ramp slope variation 1 LSB = 3.6e9*900/2^26 = 48.279 kHz/µs	kHz/µs
rlUInt16_t	idleTimeVar	Idle time for each chirp 1 LSB = 10 ns	ns
rlUInt16_t	adcStartTimeVar	ADC start time for each chirp 1 LSB = 10 ns	ns
rlUInt16_t	txEnable	TX enable bitmask b0 : TX0 Enable b1 : TX1 Enable b2 : TX2 Enable b15:3 : Reserved	-

};

10.6.39 rlFrameCfg_t

This structure is the container for the frame properties to be configured to the device.

struct rlFrameCfg {

Type	Name	Description	Units
rlUInt16_t	reserved	Reserved	0
rlUInt16_t	chirpStartIdx	Chirp start index. Valid range 0-511	-
rlUInt16_t	chirpEndIdx	Chirp end index. Valid range from chirpStartIdx to 511	
rlUInt16_t	numLoops	Number of times to repeat from chirpStartIdx to chirpEndIdx in each frame Valid range 1-255	-
rlUInt16_t	numFrames	Number of frames to transmit. Set 0 for infinite frames	-
rlUInt16_t	numAdcSamples	Number of half words per chirp that need to be transferred out over high speed interface	

		In real mode, this is same as numAdcSamples from rlProfileCfg_t In complex1x or complex2x mode, this is twice the numAdcSamples from rlProfileCfg_t	
rlUInt32_t	framePeriodicity	Frame repetition period 1LSB = 5ns Typical range 1 to 1000 ms	ns
rlUInt16_t	triggerSelect	Frame trigger method 0x0001: SW API based triggering. 0x0002: HW SYNC_IN based triggering.	
rlUInt16_t	reserved2	Reserved	0
rlUInt32_t	frameTriggerDelay	Optional time delay from the SYNC_IN trigger to the occurrence of frame chirps. 1LSB = 5ns	ns

```
};
```

10.6.40 rlSubFrameCfg_t

This structure is the container for the Sub-frame properties within the Advance frame to be configured to the device.

```
struct rlSubFrameCfg{
```

Type	Name	Description	Units
rlUInt16_t	forceProfileIdx	Force profile index. All the chirps in subframe will be forced to this profile	-
rlUInt16_t	chirpStartIdx	Valid range = 0-511	-
rlUInt16_t	numOfChirps	Num of Chirps including start index in burst 1 - 512	
rlUInt16_t	numLoops	Number of times to repeat from start chirp to end chirp in each burst, valid range = 1 to 255	
rlUInt32_t	burstPeriodicity	Burst repitition period (1 LSB = 5 ns)	-
rlUInt16_t	chirpStartIdxOffset	Chirp Start address increament for next burst, next_burst_chirp_start_idx = last_chirp_end_index + h_ChirpStartIdxIncr 0 - 511, 0 = 2nd burst = 1st burst	
rlUInt16_t	numOfBurst	Num of bursts 1 - 512	
rlUInt16_t	numOfBurstLoops	Num of bursts loops 1-512	-
rlUInt16_t	reserved1	Reserved for future use	0
rlUInt32_t	subFramePeriodicity	Sub frame period (1 LSB = 5 ns)	ns

rlUInt32_t	reserved2	Reserved for future use	0
rlUInt32_t	reserved3	Reserved for future use	0

};

10.6.41 rlAdvFrameSeqCfg_t

This structure is the container for the Advance frame sequence properties to be configured to the device.

struct rlAdvFrameSeqCfg{

Type	Name	Description	Units
rlUInt8_t	numOfSubFrames	Number of Sub frames in frames 1-4	-
rlUInt8_t	forceProfile	Profile id of all chirp forced to single profile	-
rlUInt16_t	reserved	Reserved	-
rlSubFrameCfg_t	subFrameCfg [RL_MAX_SUBFRAMES]	Subframe configuration for 4 sub frames	
rlUInt16_t	numFrames	Number of frame to transmit (0 - infinite frames)	-
rlUInt16_t	triggerSelect	Selects the mode for triggering start of transmission of frame	ns
rlUInt32_t	frameTrigDelay	Optional time delay from sync_in trigger to the actual transmission of frame chirps (1 lsb = 5 ns)	
rlUInt32_t	Reserved1	Reserved	0
rlUInt32_t	Reserved2	Reserved	0

};

10.6.42 rlSubFrameDataCfg_t

This structure is the container for Sub-frame data configuration for LVDS/CSI2.

struct rlSubFrameDataCfg{

Type	Name	Description	Units
rlUInt32_t	totalChirps	Number of Chirps in Sub-Frame = numOfChirps * numLoops * numOfBurst	-
rlUInt16_t	numAdcSamples	Number of half words of ADC samples per data packet in sub frame 1 Example 1: In real mode, if number of ADC samples per chirp in subframe1 is 256 then this value will be 256 Example 2: In complex1x or complex2x	

		modes, if number of ADC samples per chirp in subframe1 is 256 then this value will be 512 In AWR12xx: Program this as the same as number of ADC samples in each chirp of this sub frame (required to be the same) Exception: Can do #chirps based ping-pong as in AWR16xx (see below), if CP/CQ are not needed. Useful for chirp stitching use case. In AWR16xx: The ADC samples corresponding to one or more chirps can be grouped and sent to the DSP as a single packet. Program this as the number of half words of ADC samples per packet. Ensure that in one sub frame, there is integer number of such packets. Maximum size of a data packet: (16384 - 1) half words	
rlUInt8_t	numChirpsInDataPacket	Number of Chirps Per Data Packet to process at a time in sub frame 1. In AWR12xx: Program this as 1. Exception: Can be > 1 as in 16xx if CP/CQ is not needed. Useful for chirp stitching use case. In AWR16xx: The ADC samples corresponding to one or more chirps can be grouped and sent to the DSP as a single packet. Program this as the corresponding number of chirps per packet. Maximum value = 8. Note on maximum size: 8 chirps for CP and BPM.	
rlUInt8_t	reserved	Reserved	0

```
};
```

10.6.43 rlAdvFrameDataCfg_t

This structure is the container for Sub-frame data configuration for LVDS/CSI2.

struct rlAdvFrameDataCfg{

Type	Name	Description	Units
rlUInt8_t	numSubFrames	Number of Sub Frames, Valid Range (1 - 4)	-
rlUInt8_t	reserved1	reserved	0
rlUInt16_t	Reserved2	reserved	0
rlSubFrameDataCfg_t	subframeDataCfg[RL_MAX_SUBFRAMES]	Sub Frame data configuration array	-

};

10.6.44 *rlAdvFrameCfg_t*

This structure is the container for Advance Frame configuration API

struct rlAdvFrameCfg{

Type	Name	Description	Units
rlAdvFrameSeqCfg_t	frameSeq	Advance Frame sequence and Subframe configuration	-
rlAdvFrameDataCfg_t	frameData	Advance Frame data configuration	

};

10.6.45 *rlBpmCommonCfg_t*

This structure is the container for the binary phase modulation common configuration.

Type	Name	Description	Units
struct rlBpmModeCfg {			
rlUInt16_t	b2SrcSel:2	BPM_SRC_SEL (select source of BPM pattern) 00 CHIRP_CONFIG_BPM (refer to rlBpmChirpCfg_t) 01 RESERVED 10 RESERVED 11 RESERVED	-
rlUInt16_t	b1Reserved1:1	Reserved	0
rlUInt16_t	b13Reserved2:13	Reserved	0
};			
rlUInt16_t	reserved0	Reserved	0
rlUInt16_t	Reserved1	Reserved	0
rlUInt16_t	reserved2	Reserved	0
rlUInt16_t	Reserved3	Reserved	0
rlUInt16_t	Reserved4	Reserved	0

10.6.46 *rlBpmChirpCfg_t*

This structure is the container for binary phase modulation per chirp configuration.

struct rlBpmChirpCfg {

Type	Name	Description	Units
rlUInt16_t	chirpStartIdx	Chirp start index. Valid range 0-511	-
rlUInt16_t	chirpEndIdx	Chirp end index. Valid range from chirpStartIdx to 511	
rlUInt16_t	constBpmVal	b0 - CONST_BPM_VAL_TX0_TXOFF Value of Binary Phase Shift value for TX0, when during idle time b1 - CONST_BPM_VAL_TX0_TXON Value of Binary Phase Shift value for TX0, during chirp b2 - CONST_BPM_VAL_TX1_TXOFF For TX1 b3 - CONST_BPM_VAL_TX1_TXON For TX1 b4 - CONST_BPM_VAL_TX2_TXOFF For TX2 b5 - CONST_BPM_VAL_TX2_TXON	

		For TX2 b15:6 Reserved	
rlUInt16_t	reserved	Reserved	0

};

10.6.47 *rlDevDataFmtCfg_t*

This structure is the container for device data format configuration.

struct rlDevDataFmtCfg {

Type	Name	Description	Units
rlUInt16_t	rxChannelEn	Bitmap for each of the RX channel b0: 0 – Disable RX channel 0; 1 – Enable RX channel 0 b1: 0 – Disable RX channel 1; 1 – Enable RX channel 1 b2: 0 – Disable RX channel 2; 1 – Enable RX channel 2 b3: 0 – Disable RX channel 3; 1 – Enable RX channel 3 b15:4 Reserved	-
rlUInt16_t	adcBits	ADC out bits - 0(12 Bits), 1(14 Bits), 2(16 Bits)	-
rlUInt16_t	adcFmt	ADC out format 0 - Real 1 - Complex 2 - Complex with Image band 3- Pseudo Real	
rlUInt8_t	iqSwapSel	IQ bit swap selection b1:0 – Bitmap to swap IQ samples if complex rep 00 – Sample Interleave Mode - I first 01 – Sample Interleave Mode – Q first b7:2 Reserved	-
rlUInt8_t	chInterleave	Channel interleaving of the samples stored in the ADC buffer to be transferred out on the	-

		data path b1:0 00 – Interleaved mode of storage 01 – Non-Interleaved mode b7:2 Reserved	
rlUInt32_t	Reserved	0x00000000	0

};

10.6.48 rlDevDataPathCfg_t

This structure is the container for device data path configuration.

struct rlDevDataPathCfg {

Type	Name	Description	Units
rlUInt8_t	intfSel	Data path interface select 0 – CSI2 interface selected. 1 – LVDS interface selected. 2 – SPI interface selected (applicable when a large inter-frame time is provided to transfer the data over the SPI), data will be chunked into multiple async event message where message SB-ID is RL_DEV_AE_MSS_ADC_DATA_SB	
rlUInt8_t	transferFmtPkt0	Data Output format b5:0 Packet 0 content selection 000001b ADC_DATA_ONLY 000110b CP_ADC_DATA 001001b ADC_CP_DATA 110110b CP_ADC_CQ_DATA Others Reserved b7:6 Packet 0 virtual channel number (valid only for CSI2) 00 Virtual channel number 0 (Default) 01 Virtual channel number 1 10 Virtual channel number 2 11 Virtual channel number 3	-
rlUInt8_t	transferFmtPkt1	b5:0 Packet 1 content selection 000000 – Suppress Packet 1 001110 – CP_CQ_DATA	-

		001011 - CQ_CP_DATA b7:6 Packet 1 virtual channel number (valid only for CSI2) 00 Virtual channel number 0 (Default) 01 Virtual channel number 1 10 Virtual channel number 2 11 Virtual channel number 3	
rlUInt8_t	cqConfig	This field specifies the data size of CQ samples on the lanes. b1:0 Data size 00 12 bits 01 14 bits 10 16 bits b7:2 Reserved	-
rlUInt8_t	cq0TransSize	Number of samples (in 16 bit halfwords) of CQ0 data to be transferred. Valid range [32 halfwords to 128 halfwords] Note: Ensure that the number of halfwords specified are a multiple of the number of lanes selected.	-
rlUInt8_t	cq1TransSize	Number of samples (in 16 bit halfwords) of CQ1 data to be transferred. Valid range [32 halfwords to 128 halfwords] Note: Ensure that the number of halfwords specified are a multiple of the number of lanes selected.	-
rlUInt8_t	cq2TransSize	Number of samples (in 16 bit halfwords) of CQ2 data to be transferred. Valid range [32 halfwords to 128 halfwords] Note: Ensure that the number of halfwords specified are a multiple of the number of lanes selected.	-
rlUInt8_t	Reserved	Reserved	0

};

10.6.49 rlDevDataPathClkCfg_t

This structure is the container for the HSI clock configuration

struct rlDevLvdsClkCfg {

Type	Name	Description	Units
rlUInt8_t	laneClkCfg	Bitmap for the lane clock selection b0 0 : SDR Clock 1 : DDR Clock b7:1 reserved	-
rlUInt8_t	dataRate	Bitmap for data rate selection 0000b 900 Mbps (DDR only) 0001b 600 Mbps (DDR only) 0010b 450 Mbps (SDR, DDR) 0011b 400 Mbps (DDR only) 0100b 300 Mbps (SDR, DDR) 0101b 225 Mbps (DDR only) 0110b 150 Mbps (SDR, DDR)	-
rlUInt16_t	Reserved	-	0

};

10.6.50 rlDevHsiClk_t

This structure is the container for mmwave radar high speed clock configuration.

struct rlDevHsiClk {

Type	Name	Description	Units
rlUInt16_t	hsiClk	SDR - 0x5(900), 0xA(600), 0x6(450), 0x2(400), 0xB(300), 0x7(225) DDR - 0xD(900), 0x9(600), 0x5(450), 0x1(400), 0xA(300), 0x6(225), 0xB(150)	-
rlUInt16_t	reserved	-	0

};

10.6.51 rlDevHsiCfg_t

This structure is the container for High Speed Interface (HSI) clock, number of lanes, data rate and data format configuration.

Struct rlDevHsiCfg {

Type	Name	Description	Units
rlDevDataFmtCfg_t	datafmt	HSI data format configuration	-
rlDevDataPathCfg_t	dataPath	HSI data path	

		configuration	
rlDevDataPathClkCfg_t	dataPathClk	Data path clock configuration	

};

10.6.52 rlDevLaneEnable_t

This structure is the container for the High Speed Interface (HSI) lane configuration

struct rlDevLaneEnable_t {

Type	Name	Description	Units
rlUInt16_t	laneEn	Lane enable bitmap. b0 - LANE0_EN 0 Disable lane 0 1 Enable lane 0 b1 - LANE1_EN 0 Disable lane 1 1 Enable lane 1 b2 - LANE2_EN 0 Disable lane 2 1 Enable lane 2 b3 - LANE3_EN 0 Disable lane 3 1 Enable lane 3 b15:4 Reserved	
rlUInt16_t	Reserved	Reserved	0

};

10.6.53 rlDevCsi2Cfg_t

This structure is the container for the CSI2 lane configuration

struct rlDevCsi2Cfg_t {

Type	Name	Description	Units
rlUInt32_t	lanePosPolSel	Valid values (Should be a unique position if lane 0 is enabled, ignored if lane 0 is not enabled): b2:0 DATA_LANE0_POS 000 Unused 001 Position 1 (default) 010 Position 2 011 Position 3 100 Position 4 101 Position 5 b3 DATA_LANE0_POL 0 PLUSMINUS pin	-

	order
1	MINUSPLUS pin order

b6:4 DATA_LANE1_POS
Valid values (Should be a unique position if lane 1 is enabled, ignored if lane 1 is not enabled)

000	Unused
001	Position 1
010	Position 2 (default)
011	Position 3
100	Position 4
101	Position 5

b7 DATA_LANE1_POL

0	PLUSMINUS pin order
1	MINUSPLUS pin order

b10:8 DATA_LANE2_POS
Valid values (Should be a unique position if lane 2 is enabled, ignored if lane 2 is not enabled)

000	Unused
001	Position 1
010	Position 2
011	Position 3
100	Position 4 (default)
101	Position 5

b11 DATA_LANE2_POL

0	PLUSMINUS pin order
1	MINUSPLUS pin order

b14:12 DATA_LANE3_POS
Valid values (Should be a unique position if lane 3 is enabled, ignored if lane 3 is not enabled)

000	Unused
001	Position 1
010	Position 2
011	Position 3
100	Position 4

		101 — Position 5 (default) b15 DATA_LANE3_POL 0 — PLUSMINUS pin order 1 — MINUSPLUS pin order b18:16 CLOCK_POS Valid values (Should be a unique position) 000 — Unused 001 — Unused 010 — Position 2 011 — Position 3 (default) 100 — Position 4 101 — Unused b19 CLOCK_POL 0 — PLUSMINUS pin order 1 — MINUSPLUS pin order b31:20 RESERVED	
rlUInt32_t	reserved1	0000	0

};

10.6.54 *rlDevLvdsCfg_t*

This structure is the container for the LVDS Lane configuration

struct rlDevLvdsLaneCfg {

Type	Name	Description	Units
rlUInt16_t	laneFmtMap	Lane format 0 – Format map 0 (Rx0,Rx1,...) 1 – Format map 1 (Rx3,Rx2,...)	-
rlUInt16_t	laneParamCfg	b0: 0 LSB First 1 MSB First b1: Packet End Pulse Enable 0 Disable 1 Enable	

		b2: CRC Enable 0 Disable 1 Enable b15:3 Reserved	

};

10.6.55 rlTestSource_t

This structure is the container for the test source configuration

struct rlTestSource {

Type	Name	Description	Units
rlTestSourceObject_t	testObj[RL_MAX_TST_SRC_OBJECTS]	Array of test Objects	-
rlTestSourceAntPos_t	rxAntPos[4]	Simulated Position of Rx Antennas	-
rlTestSourceAntPos_t	txAntPos[RL_TX_CNT]	Simulated Position of Tx Antennas	-
rlUInt16_t	reserved		0

};

10.6.56 rlTestSourceObject_t

This structure is the container for the test source position, velocity and boundary grid.

struct rlTestSourceObject {

Type	Name	Description	Units
rlInt16_t	posX	x-coordinate of the object in the xyz-space	1LSB = 1cm
rlInt16_t	posY	y-coordinate of the object in the xyz-space	1LSB = 1cm
rlInt16_t	posZ	z-coordinate of the object in the xyz-space	1LSB = 1cm
rlInt16_t	velX	velocity of the object in the direction parallel to x-axis Valid range: -5000 to 5000	1LSB = 1 cm/s
rlInt16_t	velY	velocity of the object in the direction parallel to y-axis Valid range: -5000 to 5000	1LSB = 1 cm/s
rlInt16_t	velZ	velocity of the object in the direction parallel to z-axis Valid range: -5000 to 5000	1LSB = 1 cm/s

rlInt16_t	posXMin	Boundary minimum limit for x-coordinate When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Valid Range: x: 0 to 32767	1LSB = 1cm
rlInt16_t	posYMin	Boundary minimum limit for y-coordinate When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Valid Range: -32767 to 32767	1LSB = 1cm
rlInt16_t	posZMin	Boundary minimum limit for z-coordinate When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Valid Range: -32767 to 32767	1LSB = 1cm
rlInt16_t	posXMax	Boundary maximum limit for x-coordinate When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Valid Range: x: 0 to 32767 cm, y & z: +/-32767 cm	1LSB = 1cm
rlInt16_t	posYMax	Boundary maximum limit for y-coordinate When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Valid Range: x: 0 to 32767 cm, y & z: +/-32767 cm	1LSB = 1cm
rlInt16_t	posZMax	Boundary maximum limit for z-coordinate When the current position crosses this boundary, the emulator returns the corresponding coordinate to the originally programmed value. Valid Range: x: 0 to 32767 cm,	1LSB = 1cm

		y & z: +/-32767 cm	

};

10.6.57 rlTestSourceAntPos_t

This structure is the container for test source antenna position.

struct rlTestSourceAntPos {

Type	Name	Description	Units
rlInt8_t	antPosX	x-coordinate of receiver antenna position in the xyz-space Only x and z coordinates to be provided as radar is assumed to be on (y=0 plane). Valid range = -120 to 120 Note: RX0 antenna coordinates should be set to (0,0,0)	1LSB = Wavelength/8
rlInt8_t	antPosZ	z-coordinate of receiver antenna position in the xyz-space Only x and z coordinates to be provided as radar is assumed to be on (y=0 plane). Valid range = -120 to 120 Note: RX0 antenna coordinates should be set to (0,0,0)	1LSB = Wavelength/8

};

10.6.58 rlTestSourceEnable_t

This structure is the container to enable the test source

struct rlTestSourceEnable {

Type	Name	Description	Units
rlUInt16_t	tsEnable	b0 enable 0 Test source Disable 1 Test source Enable b15:1 reserved	-
rlUInt16_t	tsMode	reserved	-

};

10.6.59 rlRfTempData_t

This structure is the container for the device temperature data.

struct rlRfTempData {

Type	Name	Description	Units
rlInt32_t	time	BSS local Time from device powerup.	1 LSB = 1 ms
rlInt16_t	tmpRx0Sens	RX0 temperature sensor reading (signed value)	1 LSB = 1 °C
rlInt16_t	tmpRx1Sens	RX1 temperature sensor reading (signed value)	1 LSB = 1 °C
rlInt16_t	tmpRx2Sens	RX2 temperature sensor reading (signed value)	1 LSB = 1 °C
rlInt16_t	tmpRx3Sens	RX3 temperature sensor reading (signed value)	1 LSB = 1 °C
rlInt16_t	tmpTx0Sens	TX0 temperature sensor reading (signed value)	1 LSB = 1 °C
rlInt16_t	tmpTx1Sens	TX1 temperature sensor reading (signed value)	1 LSB = 1 °C
rlInt16_t	tmpTx2Sens	TX2 temperature sensor reading (signed value)	1 LSB = 1 °C
rlInt16_t	tmpPmSens	PM temperature sensor reading (signed value)	1 LSB = 1 °C
rlInt16_t	tmpDig0Sens	Digital temp sensor reading (signed value)	1 LSB = 1 °C
rlInt16_t	reserved	reserved	0

};

10.6.60 rlDfeRxStatReport_t

This structure is container of DFE statistic report for a receiver.

struct rlDfeRxdStatReport {

Type	Name	Description	Units
rlInt16_t	iAvgDC	DC value in I chain for profile x, RX channel x	1 LSB = 1V/2^15
rlInt16_t	qAvgDC	DC value in Q chain for profile x, RX channel x	1 LSB = 1V/2^15
rlUInt16_t	iAvgPwr	Power in I chain for profile x, RX channel x	1 LSB = 1V^2/2^15
rlUInt16_t	qAvgPwr	Power in Q chain for profile x, RX channel x	1 LSB = 1V^2/2^15
rlInt32_t	iqAvgCroCorrel	Cross correlation between I and Q chains for profile x, RX channel	1 LSB = 1V^2/2^30

};

10.6.61 rlContModeCfg_t

This structure is container of Continuous Mode configuration API.

struct rlContModeCfg{

Type	Name	Description	Units
rlUInt32_t	startFreqConst	Start frequency for each profile Valid range: 0x5471C71B to 0x5A000000	1 LSB = 3.6e9 / 2^26 Hz = 53.644 Hz
rlUInt32_t	txOutPowerBackoffCode	Concatenated code for output power backoff for TX0, TX1, TX2 b7:0 TX0 output power back off b15:8 TX1 output power back off b23:16 TX2 output power back off b31:24 Reserved	-
rlUInt32_t	txPhaseShifter	Concatenated phase shift for TX0/1/2, b1:0 RESERVED (set it to 0b00) b7:2 TX0 phase shift value b9:8 RESERVED (set it to 0b00) b15:10 TX1 phase shift value b17:16 RESERVED (set it to 0b00) b23:18 TX2 phase shift value b31:24 Reserved	1 LSB = 360/2^6 = 5.6 degrees
rlUInt16_t	digOutSampleRate	ADC Sampling rate for each profile is encoded in 2 bytes (16 bit unsigned number) Valid range 2000 to 37500	1 LSB = 1 ksps
rlUInt8_t	hpfCornerFreq1	Code for HPF1 corner frequency 0x00 175 kHz 0x01 235 kHz 0x02 350 kHz 0x03 700 kHz	-
rlUInt8_t	hpfCornerFreq2	Code for HPF2 corner frequency 0x00 350 kHz 0x01 700 kHz 0x02 1.4 MHz 0x03 2.8 MHz	-
rlUInt8_t	rxGain	This field defines RX gain for continuous streaming mode. Bit Definition [5:0] RX_GAIN This field defines RX gain for each profile. 1 LSB = 1 dB Valid values: all even values from 24 to 52 [7:6] RF_GAIN_TARGET Value RF gain target 00 30 dB	1 LSB = 1 dB

		01 34 dB 10 RESERVED 11 26 dB	
rlUInt8_t	vcoSelect	Bit Description b0 FORCE_VCO_SEL 0 internal VCO selection 1 external VCO selection b1 VCO_SEL 0 - VCO1 (76 - 78 GHz) 1 - VCO2 (77 - 81 GHz) b7:2 RESERVED	-
rlUInt16_t	Reserved0	Reserved	0

};

10.6.62 rlContModeEn_t

This structure is container of Continuous Mode Enable API.

struct rlContModeEn{

Type	Name	Description	Units
rlInt16_t	contModeEn	Enable continuous steaming mode 0x00 Disable continuous streaming mode 0x01 Enable continuous streaming mode	-
rlInt16_t	reserved1	Reserved	0

};

10.6.63 rlDevContStreamingModeCfg_t

This structure is container of Continuous streaming mode data size configuration.

struct rlDevContStreamingModeCfg{

Type	Name	Description	Units
rlInt16_t	contStreamModeEn	Enable continuous steaming mode 0x00 Disable continuous streaming mode 0x01 Enable continuous streaming mode	-
rlInt16_t	dataTransSize	Sample count. This refers to the number of samples per channel. The max allowed value allowed values depending on other configurations (No. of channels enabled and real/complex data). Ex: It is 8192 for real only mode one channel, and 4096 for complex mode one channel.	-

```
};
```

10.6.64 rlDfeStatReport_t

This structure is container of DFE statistic report for all profiles and receiver.

struct rlDfeStatReport{

Type	Name	Description	Units
rlDfeRxStatReport_t	dfeStatRepo[][]	Contain DFE statistic report for all profile and receiver	-

};

10.6.65 rlDynPwrSave_t

This structure is container to enable power save mode for transmitter and receiver.

struct rlDynPwrSave {

Type	Name	Description	Units
rlUInt16_t	blkCfg	b0 Enable power save by switching off TX during Inter-chirp idle time 0 Disable 1 Enable b1 Enable power save by switching off RX during Inter-chirp idle time 0 Disable 1 Enable b2 Enable power save by switching off LO distribution blocks during Inter-chirp idle time 0 Disable 1 Enable	-
rlUInt16_t	reserved	Reserved	0

};

10.6.66 rlRfDevCfg_t

This structure is container to set global asynchronous event direction

struct rlRfDevCfg {

Type	Name	Description	Units
rlUInt32_t	aeDirection	Bit Definition b1:0 Global Async event direction 00 - radarSS to MSS 01 - radarSS to HOST 10 - radarSS to DSS 11 - RESERVED The ASYNC_EVENT_DIR controls the direction for following ASYNC_EVENTS	-

		[1.] CPU_FAULT [2.] ESM_FAULT [3.] GPADC_READ All other ASYNC_EVENTs are sent to the subsystem which issues the API b3:2 Monitoring Async Event direction 00 - radarSS to MSS 01 - radarSS to HOST 10 - radarSS to DSS 11 - RESERVED b31:4 Reserved	
rlUInt8_t	aeControl	Bit Definition b0:FRAME_START_ASYNC_EVENT_DIS 0 Frame Start async event enable 1 Frame Start async event disable b1: FRAME_STOP_ASYNC_EVENT_DIS 0 Frame Stop async event enable 1 Frame Stop async event disable b7:2 Reserved	-
rlUInt8_t	Reserved0	Reserved	0
rlUInt8_t	Reserved1	Reserved	0
rlUInt8_t	bssDigCtrl	Bit Definition b0: Watchdog enable/disable 0 Keep watchdog disabled 1 Enable watch dog b7:1: Reserved	-
rlUInt8_t	aeCrcConfig	CRC Config for Asynchornous event message Value Description 0 16 bit CRC for radarSS async events 1 32 bit CRC for radarSS async events 2 64 bit CRC for radarSS async events	-
rlUInt8_t	Reserved2	Reserved	0
rlUInt16_t	Reserved3	Reserved	0

};

10.6.67 rlGpAdcSamples_t

This structure is container of number of samples configuration to be collected for GPADC.

struct rlGpAdcSamples{

Type	Name	Description	Units
rlUInt8_t	sampleCnt	Number of sample to collect	-
rlUInt8_t	reserved	Reserved	0

};

10.6.68 rlGpAdcCfg_t

This structure is container to set global asynchronous event direction struct rlGpAdcCfg{

Type	Name	Description	Units
rlUInt16_t	enable	[b0] 0:Disable, 1:Enable; ANATEST1 [b1] 0:Disable, 1:Enable; ANATEST2 [b2] 0:Disable, 1:Enable; ANATEST3 [b3] 0:Disable, 1:Enable; ANATEST4 [b4] 0:Disable, 1:Enable; IFORCE [b5] 0:Disable, 1:Enable; VSENSE [b6] 0:Disable, 1:Enable; IFORCEBUF	-
rlGpAdcSamples_t	numOfSamples[12]	Configure number of samples to be collected for each sensor	-
rlUInt16_t	Reserved0	Reserved	0

};

10.6.69 *rlRfLdoBypassCfg_t*

This structure is container to set Radar RF LDO bypass enable/disable configuration

struct rlRfLdoBypassCfg{

Type	Name	Description	Units
rlUInt16_t	ldoBypassEnable	0:Disable, 1:Enable	-
rlUInt16_t	supplyMonIrDrop	IR drop is the voltage drop from the PMIC output to the device pin. The user should program the voltage drop in percentage units which will be used for measuring the external supplies. Value Description 0 IR drop of 0% 1 IR drop of 3% 2 IR drop of 6% 3 IR drop of 9%	-

};

10.6.70 *rlRfPhaseShiftCfg_t*

This structure is container to set Radar RF Phase Shift enable/disable configuration

struct rlRfPhaseShiftCfg{

Type	Name	Description	Units
rlUInt16_t	chirpStartIdx	Chirp Start Index, Valid Range 0 -511	-
rlUInt16_t	chirpEndIdx	Chirp End Index, Valid Range from	-

		chirpStartIdx to 511	
rlUInt8_t	tx0PhaseShift	TX0 phase shift definition [b1:0] reserved (set it to 0b00) [b7:2] TX0 phase shift value Valid range: 0 to 63	1 LSB = 360/2^6 = 5.625 degrees
rlUInt8_t	Tx1PhaseShift	TX1 phase shift definition [b1:0] reserved (set it to 0b00) [b7:2] TX1 phase shift value Valid range: 0 to 63	1 LSB = 360/2^6 = 5.625 degrees
rlUInt8_t	Tx2PhaseShift	TX2 phase shift definition [b1:0] reserved (set it to 0b00) [b7:2] TX2 phase shift value Valid range: 0 to 63	1 LSB = 360/2^6 = 5.625 degrees
rlUInt8_t	reserved	Reserved	0

};

10.6.71 rlRfPALoopbackCfg_t

This structure is container to set Radar RF PA loopback configuration

struct rlRfPALoopbackCfg{

Type	Name	Description	Units
rlUInt16_t	paLoopbackFreq	Value is a 100MHz divider which sets the loopback frequency.	-
rlUInt8_t	paLoopbackEn	Enable/Disable PA loopback [b7:0] 1: PA loopback Enable, 0: PA loopback Disable	-
rlUInt8_t	reserved	Reserved	0

};

10.6.72 rlRfPSLoopbackCfg_t

This structure is container to set Radar RF Phase shift loopback configuration

struct rlRfPSLoopbackCfg{

Type	Name	Description	Units
rlUInt16_t	psLoopbackFreq	Loop back frequency in kHz.	1 LSB = 1kHz
rlUInt16_t	reserved0	Reserved	0
rlUInt8_t	psLoopbackEn	Enable/Disable PA loopback. [b0] 1: PS loopback Enable, 0: PS loopback Disable	-
rlUInt8_t	psLoopbackTxId	Tx used for loopback [b0] : Tx0 is used for loopback [b1] : Tx1 is used for loopback [b7:2]: reserved	-
rlUInt8_t	pgaGainIndex	PGA gain value 0 : PGA is OFF 1 : -22 dB 2 : -16 dB 3 : -14 dB	-

		4 : -11 dB 5 : -9 dB 6 : -6 dB 7 : -5 dB 8 : -4 dB 9 : -3 dB 10: -2 dB 11: -1 dB 12: 0 dB 13: 1 dB 14: 2 dB 15: 3 dB 16: 4 dB 17: 5 dB 18: 6 dB 19: 7 dB 20: 8 dB 21: 9 dB 255-22:RESERVED	
rlUInt8_t	reserved1	Reserved	0

};

10.6.73 *rlRfIFLoopbackCfg_t*

This structure is container to set Radar RF IF loopback configuration

struct rlRfIFLoopbackCfg{

Type	Name	Description	Units
rlUInt16_t	ifLoopbackFreq	IF loopback frequency value 0 : 180 kHz 1 : 240 kHz 2 : 360 kHz 3 : 720 kHz 4 : 1 MHz 5 : 2 MHz 6 : 2.5 MHz 7 : 3 MHz 8 : 4.017857 MHz 9 : 5 MHz 10: 6 MHz 11: 8.03 MHz 12: 9 MHz 13: 10 MHz 65535-14:RESERVED	-
rlUInt8_t	ifLoopbackEn	Enable/Disable IF loopback [b7:0] 1: IF loopback Enable, 0: IF loopback Disable	-
rlUInt8_t	reserved	Reserved	0

};

10.6.74 rlRfProgFiltCoeff_t

This structure is container to set Radar RF Array of coefficients for the RF programmable filter.

struct rlRfProgFiltCoeff{

Type	Name	Description	Units
rlUInt16_t	coeffArray[104]	The array of coefficients for the programmable filter, across all profiles, to be stored in the coefficient RAM. Each tap is a 16-bit signed number. The exact set of taps to be used for a given profile can be specified through RL_RF_PROG_FILT_CONF_SET_SB	-

};

10.6.75 rlRfProgFiltConf_t

This structure is container to set Radar RF programmable filter configuration.

struct rlRfProgFiltConf{

Type	Name	Description	Units
rlUInt8_t	profileId	Profile Index for which this configuration applies.	-
rlUInt8_t	coeffStartIdx	The index of the first coefficient of the programmable filter taps corresponding to this profile in the coefficient RAM programmed using AWR_PROG_FILT_COEFF_SET_SB.	-
rlUInt8_t	progFiltLen	The length (number of taps) of the filter corresponding to this profile	-
rlUInt8_t	progFiltFreqShift	magnitude of the frequency shift	

};

10.6.76 rlRfMiscConf_t

This structure is container to set Radar RF miscellaneous configuration.

struct rlRfMiscConf{

Type	Name	Description	Units
rlUInt32_t	miscCtl	b0 PERCHIRP_PHASESHIFTER_EN 0 Per chirp phase shifter is disabled 1 Per chirp phase shifter is enabled This control is applicable only in AWR1243P. For xWR1443, xWR1642, this is a RESERVED bit and should be set to 0.	-
rlUInt32_t	reserved	Reserved	0

};

10.6.77 *rlRfCalMonTimeUntConf_t*

This structure is container to set Radar RF calibration monitoring time unit configuration.

struct rlRfCalMonTimeUntConf{

Type	Name	Description	Units
rlUInt16_t	calibMonTimeUnit	Defines the basic time unit, in terms of which calibration and/or monitoring periodicities are to be defined.	1 LSB = Duration of one frame
rlUInt8_t	numOfCascadeDev	Applicable only for 12xx devices, default value = 1	-
rlUInt8_t	devId	Applicable only for 12xx devices, default value = 1	-
rlUInt32_t	reserved	Reserved	0

};

10.6.78 *rlRfCalMonFreqLimitConf_t*

This structure is container to set Radar RF calibration monitoring Frequency Limit configuration.

struct rlRfCalMonFreqLimitConf{

Type	Name	Description	Units
rlUInt16_t	freqLimitLow	The sensor's lower frequency limit for calibrations and monitoring is encoded in 2 bytes (16 bit unsigned number) Valid range: 760 to 810	1 LSB = 100 MHz
rlUInt16_t	freqLimitHigh	The sensor's higher frequency limit for calibrations and monitoring is encoded in 2 bytes (16 bit unsigned number) Valid range: 760 to 810 NOTE: FREQ_LIMIT_HIGH should be strictly greater than FREQ_LIMIT_LOW	1 LSB = 100 MHz
rlUInt32_t	reserved0	Reserved	0
rlUInt32_t	Reserved1	Reserved	0

};

10.6.79 *rlRfInitCalConf_t*

This structure is container to set Radar RF Init Calibration configuration.

struct rlRfInitCalConf{

Type	Name	Description	Units
rlUInt32_t	calibEnMask	Allowed values = 0x000 or 0xFFF Normally, upon receiving RF INIT message, the BSS performs all relevant initial calibrations. This step can be disabled by the host	-

		by setting this field to 0x00. If disabled, the host needs to send the INJECT CALIB DATA message so that the BSS can operate using the calibration data thus injected. Each of these calibrations can be selectively disabled by issuing this message before RF INIT message. Bit Calibration 0 [Reserved] 1 [Reserved] 2 [Reserved] 3 [Reserved] 4 Enable LODIST calibration 5 Enable RX ADC DC offset calibration 6 Enable HPF cutoff calibration 7 Enable LPF cutoff calibration 8 Enable Peak detector calibration 9 Enable TX Power calibration 10 Enable RX gain calibration 11 Enable TX Phase calibration 12 Enable RX IQMM calibration 31:13 [Reserved]	
rlUInt8_t	reserved0	Reserved	0
rlUInt8_t	Reserved1	Reserved	0
rlUInt16_t	Reserved2	Reserved	0
rlUInt32_t	Reserved3	Reserved	0

};

10.6.80 rlRunTimeCalibConf_t

This structure is container to set Radar RF Run time calibration configuration.

struct rlRunTimeCalibConf{

Type	Name	Description	Units
rlUInt32_t	oneTimeCalibEnMask	Bit: Calibration 0:[Reserved] 1: [Reserved] 2: [Reserved] 3: [Reserved] 4: Enable LODIST calibration 5: [Reserved] 6: [Reserved] 7: [Reserved] 8: [Reserved] 9: Enable TX Power calibration 10:Enable RX gain calibration 11:[Reserved]	-

		12:[Reserved] 31:13:[Reserved]	
rlUInt32_t	periodicCalibEnMask	automatic periodic triggering of calibrations of various RF/analog aspects. It has same bit definition as above	-
rlUInt32_t	calibPeriodicity	As specified in RL_RF_CALIB_MON_TIME_UNIT_SB	1 LSB = 1 CALIB_MON_TIME_UNIT
rlUInt8_t	reportEn	Calibration report enable configuration 0 - Disable Calibration Reports 1 - Enable Calibration Reports	-
rlUInt8_t	reserved1	Reserved	0
rlUInt16_t	reserved2	Reserved	0
rlUInt32_t	reserved3	Reserved	0

};

10.6.81 rlInterRxGainPhConf_t

This structure is container to Inter-Rx gain and phase mismatch correction configuration.

struct rlInterRxGainPhConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this configuration applies.	-
rlUInt8_t	Reserved[4]	Reserved	0
rlUInt8_t	rxGain[4]	Bits Assignment b7:0 RX0 digital gain b15:8 RX1 digital gain b23:16 RX2 digital gain b31:24 RX3 digital gain	1 LSB = 0.1 dB
rlUInt16_t	rxPhase[4]	Bits Assignment b15:0 RX0 digital phase shift b31:16 RX1 digital phase shift b47:32 RX2 digital phase shift b63:48 RX3 digital phase shift	1 LSB = 360 degree / 2^16 ~ 0.0055 Valid Range: 0 to 65535 NOTE: This field is NOT applicable when

			ADC_OUT_FMT is 00
rlUInt8_t	Reserved0[8U]	Reserved	0

};

10.6.82 *rlRxGainTempLutReadReq_t*

This structure is container to read RX gain temperature LUT.

struct rlRxGainTempLutReadReq{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this configuration applies.	-
rlUInt8_t	Reserved[3U]	Reserved	0

};

10.6.83 *rlRxGainTempLutData_t*

This structure is container to inject RX gain temperature LUT.

struct rlRxGainTempLutData{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this configuration applies.	-
rlUInt8_t	rxGainTempLut[19U]		-

};

10.6.84 *rlTxGainTempLutReadReq_t*

This structure is container to read TX gain temperature LUT.

struct rlTxGainTempLutReadReq {

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this configuration applies.	-
rlUInt8_t	Reserved[3]	Reserved	0

};

10.6.85 rlTxGainTempLutData_t

This structure is container to inject TX gain temperature LUT.

struct rlTxGainTempLutData {

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this configuration applies.	-
rlUInt8_t	txGainTempLut[RL_TX_CNT][19U]		-
rlUInt8_t	reserved[2U]		

};

10.6.86 rlMonDigEnables_t

This structure is container to Digital monitoring configuration within radar sub-system.

struct rlMonDigEnables{

Type	Name	Description	Units
rlUInt32_t	enMask	Bit: Dig Monitoring 0 Reserved 1 CR4 and VIM lockstep test of diagnostic 2 Reserved 3 VIM test of diagnostic 4 Reserved 5 Reserved 6 CRC test of diagnostic 7 RAMPGEN memory ECC test of diagnostic 8 DFE Parity test of diagnostic 9 DFE memory ECC test of diagnostic	-

		10 RAMPGEN lockstep test of diagnostic 11 FRC lockstep test of diagnostic 12 Reserved 13 Reserved 14 Reserved 15 WDT test of diagnostic 16 ESM test of diagnostic 17 DFE STC 18 FRC test of diagnostic 19 ATCM, BTCM ECC test of diagnostic 20 ATCM, BTCM parity test of diagnostic 21 DCC test of diagnostic 22 SOCC test of diagnostic 23 GPADC test of diagnostic 24 FFT test of diagnostic 25 RTI test of diagnostic 26 PCR test of diagnostic 31:27 RESERVED	
rlUInt8_t	testMode	0 – Production Mode 1 – Characterization Mode	

rlUInt8_t	reserved0[3U]	Reserved	0
rlUInt32_t	reserved1	Reserved	0

};

10.6.87 rlDigMonPeriodicConf_t

This structure is container to Digital monitoring latent fault reporting configuration within radar sub-system.

struct rlDigMonPeriodicConf{

Type	Name	Description	Units
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without any threshold check 1 Report is sent only on a failure 2 Report is sent every monitoring period with threshold check	-
rlUInt8_t	reserved [3]	Reserved	0
rlUInt32_t	periodicEnableMask	Bit Monitoring 0 PERIODIC_CONFG_REGISTER_READ_EN 1 ESM_MONITORING_EN 2 DFE_STC_EN 3 FRAME_TIMING_MONITORING_EN 31:4 RESERVED	
rlUInt32_t	reserved1	Reserved	0

};

10.6.88 rlMonAnaEnables_t

This structure is container to Analog monitoring configuration.

struct rlMonAnaEnables{

Type	Name	Description	Units
rlUInt32_t	enMask	Bit Analog monitoring control 0 TEMPERATURE MONITOR EN	-

		1 RX_GAIN_PHASE_MONITOR_EN 2 RX_NOISE_MONITOR_EN 3 RX_IFSTAGE_MONITOR_EN 4 TX0_POWER_MONITOR_EN 5 TX1_POWER_MONITOR_EN 6 TX2_POWER_MONITOR_EN 7 TX0_BALLBREAK_MONITOR_EN 8 TX1_BALLBREAK_MONITOR_EN 9 TX2_BALLBREAK_MONITOR_EN 10 TX_GAIN_PHASE_MONITOR_EN 11 TX0_BPM_MONITOR_EN 12 TX1_BPM_MONITOR_EN 13 TX2_BPM_MONITOR_EN 14 SYNTH_FREQ_MONITOR_EN 15 EXTERNAL_ANALOG_SIGNALS_MONITOR_EN 16 INTERNAL_TX0_SIGNALS_MONITOR_EN 17 INTERNAL_TX1_SIGNALS_MONITOR_EN 18 INTERNAL_TX2_SIGNALS_MONITOR_EN 19 INTERNAL_RX_SIGNALS_MONITOR_EN 20 INTERNAL_PMCLKLO_SIGNALS_MONITOR_EN 21 INTERNAL_GPADC_SIGNALS_MONITOR_EN 22 PLL_CONTROL_VOLTAGE_MONITOR_EN 23 DCC_CLOCK_FREQ_MONITOR_EN 24 RX_IF_SATURATION_MONITOR_EN 25 RX_SIG_IMG_BAND_MONITORING_EN 31:25 RESERVED	
rlUInt32_t	reserved1	Reserved	0

};

10.6.89 *rlTempMonConf_t*

This structure is container to Temperature sensor monitoring configuration

struct rlTempMonConf{

Type	Name	Description	Units
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only	-

		upon a failure (after checking for thresholds) 2 Report is sent every monitoring period with threshold check	
rlUInt8_t	reserved0	Reserved	0
rlUInt16_t	anaTempThreshMin	The temperatures read from near the sensors near the RF analog modules are compared against a minimum threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). Valid range: TBD Recommended value = TBD	1 LSB = 1 degree Celsius, signed number
rlUInt16_t	anaTempThreshMax	The temperatures read from near the sensors near the RF analog modules are compared against a minimum threshold.	1 LSB = 1 degree Celsius

		The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). Valid range: TBD Recommended value = TBD	
rlUInt16_t	digTempThreshMin	The temperatures read from near the sensor near the digital module are compared against a minimum threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). Valid range: TBD Recommended value = TBD	1 LSB = 1 degree Celsius
rlUInt16_t	digTempThreshMax	The temperatures read from near the sensor near the digital module are	1 LSB = 1 degree Celsius

		compared against a minimum threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). Valid range: TBD Recommended value = TBD	
rlUInt16_t	tempDiffThresh	The maximum difference across temperatures read from all the enabled sensors is compared against this threshold.The comparison result is part of the monitoring report message(Error bit is set if the measured difference exceeds this field)	1 LSB = 1o Celsius, signed number Valid range : TBD Recommended value = TBD
rlUInt32_t	reserved1	Reserved	0
rlUInt32_t	reserved1	Reserved	0

};

10.6.90 *rlRxGainPhaseMonConf_t*

This structure is container to RX gain and phase monitoring configuration.

struct rlRxGainPhaseMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this monitoring configuration applies	-
rlUInt8_t	rfFreqBitMask	This field indicates the RF frequencies inside the profile's RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled w.r.t. the profile's RF band.	-
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds) 2 Report is sent every monitoring period with threshold check	
rlUInt8_t	reserved0	Reserved	0
rlInt16_t	rxGainAbsThresh	The magnitude of difference between the programmed and measured RX gain for each enabled channel at each enabled RF frequency, is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting	1 LSB = 0.1 dB

		the offset given in the RX_GAIN_MISMATCH_OFFSET_VALUE field	
rlUInt16_t	rxGainMismatchErrThresh	The magnitude of difference between measured RX gains across the enabled channels at each enabled RF frequency is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if the measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting the offset given in the RX_GAIN_MISMATCH_OFFSET_VALUE field.	1 LSB = 0.1 dB -
rlUInt16_t	rxGainFlatnessErrThresh	The magnitude of measured RX gain flatness error, for each enabled channel, is compared against this threshold. The flatness error for a channel is defined as the peak to peak variation across RF frequencies. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting the offset given in the RX_GAIN_MISMATCH_OFFSET_VALUE field.	1 LSB = 0.1 dB
rlUInt16_t	rxGainPhaseMismatchErr Thresh	The magnitude of measured RX phase mismatch across the enabled channels at each enabled RF frequency is compared against this threshold. The comparison result is part of the monitoring report message	1 LSB = 360/2^16

		(Error bit is set if any measurement is above this threshold). Before the comparison, the measured phases for each RF and RX are adjusted by subtracting the offset given in the RX_PHASE_MISMATCH_OFFSET_VALUE field.	
rlUInt16_t	rxGainMismatchOffsetVal [4] [3]	The offsets to be subtracted from the measured RX gain for each RX and RF before the relevant threshold comparisons are given here. Byte numbers corresponding to different RX and RF, in this field are here: RF1 RF2 RF3 RX0 [1:0] [9:8] [17:16] RX1 [3:2] [11:10] [19:18] RX2 [5:4] [3:12] [21:20] RX3 [7:6] [15:14] [23:22]	1 LSB = 0.1 dB, signed number
rlUInt16_t	rxGainPhaseMismatch OffsetVal [4] [3]	The offsets to be subtracted from the measured RX phase for each RX and RF before the relevant threshold comparisons are given here. Byte numbers corresponding to different RX and RF, in this field are here: RF1 RF2 RF3 RX0 [1:0] [9:8] [17:16] RX1 [3:2] [11:10]	1 LSB = 360 / 2^16 , unsigned number

		[19:18] RX2 [5:4] [13:12] [21:20] RX3 [7:6] [15:14] [23:22]	
rlUInt16_t	reserved0	Reserved	0
rlUInt16_t	reserved0	Reserved	0

};

10.6.91 rlRxNoiseMonConf_t

This structure is container to RX noise figure monitoring configuration.

struct rlRxNoiseMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this monitoring configuration applies.	-
rlUInt8_t	rfFreqBitMask	This field indicates the exact RF frequencies inside the profile's RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled w.r.t. the profile's RF band	-
rlUInt16_t	reserved0	Reserved	0
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds) 2 Report is sent every monitoring	

		period with threshold check	
rlInt8_t	reserved1	Reserved	0
rlUInt16_t	noiseThresh	The measured RX input referred noise figure at the enabled RF frequencies, for all channels, is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold)	1 LSB = 0.1 dB
rlUInt32_t	reserved2	Reserved	0

};

10.6.92 *rlRxIfStageMonConf_t*

This structure is container to RX IF stage monitoring configuration.

struct rlRxIfStageMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this monitoring configuration applies.	-
rlUInt8_t	reserved0[3U]	Reserved	0
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds) 2 Report is sent every monitoring period with threshold check	
rlUInt8_t	reserved1	Reserved	0
rlUInt16_t	hpfCutoffErrThresh	The absolute values of RX IF HPF cutoff percentage frequency	1 LSB = 1%, unsigned number

		errors are compared against the corresponding thresholds given in this field. The comparison results are part of the monitoring report message (Error bit is set if the absolute value of the errors exceeds respective thresholds).	Valid range: TBD to 255
rlInt16_t	lpfCutoffErrThresh	The absolute values of RX IF LPF cutoff percentage frequency errors are compared against the corresponding thresholds given in this field. The comparison results are part of the monitoring report message (Error bit is set if the absolute value of the errors exceeds respective thresholds).	1 LSB = 1%, unsigned number Valid range: TBD to 255
rlUInt16_t	ifaGainErrThresh	The absolute deviation of RX IFA Gain from the expected gain for each enabled RX channel is compared against the thresholds given in this field. The comparison result is part of the monitoring report message (Error bit is set if the absolute value of the errors exceeds respective thresholds).	1 LSB = 0.1dB, unsigned number Valid range: TBD to 255
rlUInt32_t	reserved2	Reserved	0

```
};
```

10.6.93 rlAllTxPowMonConf_t

This structure is container to all TX power monitoring configurations.

struct rlAllTxPowMonConf{

Type	Name	Description	Units
rlTxPowMonConf_t	*tx0PowrMonCfg	Power Monitoring Configuration for Tx0	-
rlTxPowMonConf_t	*tx1PowrMonCfg	Power Monitoring Configuration for Tx1	-
rlTxPowMonConf_t	*tx2PowrMonCfg	Power Monitoring Configuration for Tx2	-

};

10.6.94 rlTxPowMonConf_t

This structure is container to TX power monitoring configuration.

struct rlTxPowMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this monitoring configuration applies.	-
rlUInt8_t	rfFreqBitMask	his field indicates the exact RF frequencies inside the profile’s RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled w.r.t. the profile’s RF band	-
rlUInt16_t	reserved0	Reserved	0
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds) 2 Report is sent everymonitoring period with	

		threshold check	
rlUInt8_t	reserved1	Reserved	0
rlUInt16_t	txPowAbsErrThresh	The magnitude of difference between the programmed and measured TX power for each enabled channel at each enabled RF frequency, is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold).	1 LSB = 0.1 dBm
rlUInt16_t	txPowFlatnessErrThresh	The magnitude of measured TX power flatness error, for each enabled channel, is compared against this threshold. The flatness error for a channel is defined as the peak to peak variation across RF frequencies. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold).	1 LSB = 0.1 dB
rlUInt16_t	reserved2	Reserved	0
rlUInt32_t	reserved2	Reserved	0

};

10.6.95 rlAllTxBallBreakMonCfg_t

This structure is container to all TX ballbreak monitoring configuration.

struct rlAllTxBallBreakMonCfg{

Type	Name	Description	Units
rlTxBallbreakMonConf_t	*tx0BallBrkMonCfg	Tx ballbreak monitoring config for Tx0	-
rlTxBallbreakMonConf_t	*tx1BallBrkMonCfg	Tx ballbreak monitoring config for Tx1	-
rlTxBallbreakMonConf_t	*tx2BallBrkMonCfg	Tx ballbreak monitoring config for Tx2	-

};

10.6.96 rlTxBallbreakMonConf_t

This structure is container to TX ball break monitoring configuration.

struct rlTxBallbreakMonConf{

Type	Name	Description	Units
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds) 2 Report is sent every monitoring	-
rlUInt8_t	reserved0	Reserved	0
rlUInt16_t	txReflCoeffMagThresh	The TX reflection coefficient's magnitude for each enabled channel is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is lower than this threshold, with the units of both quantities being the same).	1 LSB = 0.1 dB, signed number
rlUInt32_t	reserved1	Reserved	0

rlUInt32_t	reserved2	Reserved	0

};

10.6.97 rlTxGainPhaseMismatchMonConf_t

This structure is container to TX gain and phase mismatch monitoring configuration.

struct rlTxGainPhaseMismatchMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the Profile Index for which this monitoring configuration applies. The TX settings corresponding to this profile index are used during the monitoring. The RX gain used in this measurement may differ from the given profile's RX gain.	-
rlUInt8_t	rfFreqBitMask	This field indicates the exact RF frequencies inside the profile's RF band at which to measure the required parameters. When each bit in this field is set, the measurement at the corresponding RF frequency is enabled w.r.t. the profile's RF band.	
rlUInt8_t	txEn	This field indicates the TX channels that should be compared for gain and phase balance. Setting the corresponding bit to 1 enables that channel for imbalance measurement	
rlUInt8_t	rxEn	This field indicates the RX channels that should be enabled for TX to RX loopback measurement. Setting the corresponding bit to 1 enables that channel for imbalance measurement. Bit RX Channel 0 RX0 1 RX1 2 RX2 3 RX3	
rlUInt8_t	Reserved0	Reserved	0
rlUInt8_t	reportMode	Value Definition	

		0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds) 2 Report is sent every monitoring period with threshold check	
rlUInt8_t	reserved1	Reserved	0
rlUInt16_t	txGainMismatch Thresh	he magnitude of difference between measured TX powers across the enabled channels at each enabled RF frequency is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if the measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting the offset given in the TX_GAIN_MISMATCH_OFFSET_VALUE field.	1 LSB = 0.1dB, signed number Valid range: TBD to 60
rlUInt16_t	txPhaseMismatch Thresh	The magnitude of measured TX phase mismatch across the enabled channels at each enabled RF frequency is compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if any measurement is above this threshold). Before the comparison, the measured gains for each RF and RX are adjusted by subtracting the offset given in the TX_PHASE_MISMATCH_OFFSET_VALUE field.	1 LSB = 360°/ 2^16, signed number Valid range: corresponding to 2 deg (TBD) to 20 deg.
rlUInt16_t	txGainMismatchOffset Val[3] [3]	The offsets to be subtracted from the measured TX gain for each TX and RF before the relevant threshold comparisons are given here. Byte numbers corresponding to different RX and RF, in this	1 LSB = 0.1 dB

		field are here: RF1 RF2 RF3 TX0 [1:0] [7:6] [11:12] TX1 [3:2] [9:8] [15:14] TX2 [5:4] [11:10] [17:16]	
rlUInt16_t	txPhaseMismatch OffsetVal[3] [3]	he offsets to be subtracted from the measured TX phase for each TX and RF before the relevant threshold comparisons are given here. Byte numbers corresponding to different RX and RF, in this field are here: RF1 RF2 RF3 TX0 [1:0] [7:6] [11:12] TX1 [3:2] [9:8] [15:14] TX2 [5:4] [11:10] [17:16]	1 LSB = 360°/2^16.
rlUInt16_t	Reserved1	Reserved	0
rlUInt32_t	reserved2	Reserved	0

};

10.6.98 rlAllTxBpmMonConf_t

This structure is container to all TX BPM monitoring configuration.

struct rlAllTxBpmMonConf{

Type	Name	Description	Units
rlTxBpmMonConf_t	*tx0BallBrkMonCfg	Tx ballbreak monitoring config for Tx0	-
rlTxBpmMonConf_t	*tx1BallBrkMonCfg	Tx ballbreak monitoring config for Tx1	-
rlTxBpmMonConf_t	*tx2BallBrkMonCfg	Tx ballbreak monitoring config	-

		for Tx2	

};

10.6.99 rlTxBpmMonConf_t

This structure is container to TX BPM monitoring configuration.

struct rlTxBpmMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the Profile Index for which this monitoring configuration applies.	-
rlUInt8_t	reserved0	Reserved	0
rlUInt16_t	reserved1	Reserved	0
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure after checking for thresholds) 2 Report is sent every monitoring period with threshold check	
rlUInt8_t	rxEn	This field indicates the RX channels that should be enabled for TX to RX loopback measurement. Setting the corresponding bit to 1 enables that channel for imbalance measurement. Bit RX Channel 0 RX0 1 RX1 2 RX2 3 RX3	
rlUInt16_t	txBpmPhaseErrThresh	The deviation of the TX output phase difference between the two BPM settings from the ideal 180o is	1 LSB = 360°/2^16.

		compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is lower than this threshold, with the units of both quantities being the same).	
rlUInt16_t	txBpmAmplErrThresh	The deviation of the TX output amplitude difference between the two BPM settings is compared against the threshold given here. The comparison result is part of the monitoring report message (Error bit is set if the measurement is lower than this threshold, with the units of both quantities being the same).	1 LSB = 0.1 dB
rlUInt16_t	reserved2	Reserved	0
rlUInt16_t	reserved3	Reserved	0

};

10.6.100rlSynthFreqMonConf_t

This structure is container to Synthesizer frequency monitoring configuration.

struct rlSynthFreqMonConf_t{

Type	Name	Description	Units
rlUInt8_t	profileIndx	his field indicates the Profile Index for which this monitoring configuration applies.	-
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure after checking for thresholds)	

		2 Report is sent every monitoring period with threshold check	
rlUInt16_t	freqErrThresh	During the chirp, the error of the measured instantaneous chirp frequency w.r.t. the desired value is continuously compared against this threshold. The comparison result is part of the monitoring report message (Error bit is set if the measurement is above this threshold, ever during the previous monitoring period).	1 LSB = 10 kHz
rlUInt8_t	monStartTime	This field determines when the monitoring starts in each chirp relative to the start of the ramp	1 LSB = 0.2us, signed number
rlUInt8_t	reserved0[3U]	Reserved	0
rlUInt32_t	reserved	Reserved	0

};

10.6.101rlExtAnaSignalsMonConf_t

This structure is container to External analog signals monitoring configuration.

struct rlExtAnaSignalsMonConf{

Type	Name	Description	Units
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure after checking for thresholds) 2 Report is sent every monitoring period with threshold check	
rlUInt8_t	reserved0	Reserved	0
rlUInt8_t	signalInpEnables	This field indicates the sets of externally fed DC signals which are to be monitored using GPADC. When each bit in this field is set, the corresponding	1 LSB = 0.2us, signed number

		signal is monitored. The monitored signals are compared against programmed limits. The comparison result is part of the monitoring report message.	
rlUInt8_t	signalBuffEnables	This field indicates the sets of externally fed DC signals which are to be buffered before being fed to the GPADC. When each bit in this field is set, the corresponding signal is buffered before the GPADC. The monitored signals are compared against programmed limits. The comparison result is part of the monitoring report message.	
rlUInt16_t	signalSettlingTime[6U]	After connecting an external signal to the GPADC, the amount of time to wait for it to settle before taking GPADC samples is programmed in this field. For each signal, after that settling time, GPADC measurements take place for 6.4us (averaging 4 samples of the GPADC output).	
rlUInt16_t	signalThresh[2U][6U]	The external DC signals measured on GPADC are compared against these minimum and maximum thresholds. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range).	
rlUInt16_t	reserved1	Reserved	0
rlUInt32_t	reserved2	Reserved	0
rlUInt32_t	reserved3	Reserved	0

```
};
```

10.6.102rlTxIntAnaSignalsMonConf_t

This structure is container for internal TX signals monitoring configuration.

```
struct rlTxIntAnaSignalsMonConf{
```

Type	Name	Description	Units

rlUInt8_t	profileIndx	The RF analog settings corresponding to this profile are used for monitoring the enabled signals, using test chirps (static frequency, at the center of the profile's RF frequency band).	
rlUInt8_t	reserved0	Reserved	0
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds) 2 Report is sent every monitoring period with threshold check	
rlUInt8_t	reserved1	Reserved	0

};

10.6.103rlRxIntAnaSignalsMonConf_t

This structure is container to internal RX signals monitoring configuration.

struct rlTxIntAnaSignalsMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	The RF analog settings corresponding to this profile are used for monitoring the enabled signals, using test chirps (static frequency, at the center of the profile's RF frequency band).	
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds)	
rlUInt16_t	reserved0	Reserved	0
rlUInt32_t	reserved1	Reserved	0

};

10.6.104rlPmClkLoIntAnaSignalsMonConf_t

This structure is container to internal signals for PM, CLK and LO monitoring configuration.

struct rlPmClkLoIntAnaSignalsMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	The RF analog settings corresponding to this profile are used for monitoring the enabled signals, using test chirps (static frequency, at the center of the profile's RF frequency band).	
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds)	
rlUInt16_t	reserved0	Reserved	0
rlUInt32_t	reserved1	Reserved	0

};

10.6.105rlGpadcIntAnaSignalsMonConf_t

This structure is container to internal signals for GPADC monitoring configuration.

struct rlGpadcIntAnaSignalsMonConf{

Type	Name	Description	Units
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds)	
rlUInt8_t	reserved0	Reserved	0
rlUInt32_t	reserved1	Reserved	0

};

10.6.106rlPllContrVoltMonConf_t

This structure is container to internal signals for PLL control voltage monitoring configuration.

struct rlPllContrVoltMonConf{

Type	Name	Description	Units
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds) 2 Report is sent every monitoring period with threshold check	
rlUInt8_t	reserved0	Reserved	0
rlUInt16_t	signalEnables	This field indicates the sets of signals which are to be monitored. When each bit in this field is set, the corresponding signal set is monitored using test chirps. Rest of the RF analog may not be ON during these test chirps. The APLL VCO control voltage can be monitored. The Synthesizer VCO control voltage for both VCO1 and VCO2 can be monitored, while operating at their respective minimum and maximum frequencies, and their respective VCO slope (Hz/V) can be monitored if both frequencies are enabled for that VCO. The monitored signals are compared against internally chosen valid limits.	
rlUInt32_t	reserved1	Reserved	0

};

10.6.107rlDualClkCompMonConf_t

This structure is container to internal signals for DCC based clock monitoring configuration.

struct rlDualClkCompMonConf{

Type	Name	Description	Units
rlUInt8_t	reportMode	Value Definition	

		0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for thresholds)	
rlUInt8_t	reserved0	Reserved	0
rlUInt16_t	dccPairEnables	This field indicates which pairs of clocks to monitor. When a bit in the field is set to 1, the firmware monitors the corresponding clock pair by deploying the hardware's Dual Clock Comparator in the corresponding DCC mode 0 XTAL BSS_600M 1 BSS_600M BSS_200M 2 BSS_600M BSS_100M 3 BSS_600M GPADC_10M 4 BSS_600M RCOSC_10M 5 BSS_600M RAMPGEN_100M	
rlUInt32_t	reserved1	Reserved	0

};

10.6.108rlRxSatMonConf_t

This structure is container to RX saturation monitoring configuration.

struct rlRxSatMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this monitoring configuration applies	
rlUInt8_t	satMonSel	01 => Enable only the ADC saturation monitor 10 => Enable only the IFA1 saturation monitor 11 => Enable both the ADC and IFA1 saturation monitors	
rlUInt16_t	Reserved0		0
rlUInt16_t	primarySliceDuration	It specifies the duration of each (primary) time slice.	1 LSB = 0.16us
rlUInt16_t	numSlices	Number of (primary + secondary) time slices to monitor.	valid range: 1 to 127
rlUInt8_t	rxChannelMask	Masks RX channels used for	

		monitoring. In every slice, saturation counts for all unmasked channels are added together, and the total is capped to 127.	
rlUInt8_t	reserved1	Reserved	0
rlUInt16_t	Reserved2	Reserved	0
rlUInt32_t	Reserved3	Reserved	0
rlUInt32_t	Reserved4	Reserved	0

};

10.6.109rlSigImgMonConf_t

This structure is container to signal and image band energy monitoring configuration.

struct rlSigImgMonConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile index for which this monitoring configuration applies	
rlUInt8_t	numSlices	Number of (primary + secondary) slices to monitor	Valid range: 1 to 127
rlUInt16_t	timeSliceNumSamples	This field specifies the number of samples constituting each time slice. The minimum allowed value for this parameter is 4.	
rlUInt32_t	reserved0	Reserved	0
rlUInt32_t	reserved1	Reserved	0

};

10.6.110rlRxMixInPwrMonConf_t

This structure is container to signal and image band energy monitoring configuration.

struct rlRxMixInPwrMonConf {

Type	Name	Description	Units
rlUInt8_t	reportMode	Value Definition 0 Report is sent every monitoring period without threshold check 1 Report is send only upon a failure (after checking for	

		thresholds) 2 Report is sent every monitoring period with threshold check	
rlUInt8_t	profileIndx	This field indicates the profile Index for which this monitoring configuration applies	rlUInt8_t
rlUInt8_t	txEnable	This field indicates if and which TX channels should be enabled while measuring RX mixer input power. Setting a bit to 1 enables the corresponding TX channel. Enabling a TX channel may help find reflection power while disabling may help find interference power. Bit 0 – TX0 Bit 1 – TX1 Bit 2 – TX2	
rlUInt8_t	reserved0	Reserved	0
rlUInt8_t	thresholds	The measured RX mixer input voltage swings during this monitoring is compared against the minimum and maximum thresholds configured in this field. The comparison result is part of the monitoring report message (Error bit is set if any measurement is outside this (minimum, maximum) range). Byte Threshold \n 0 Minimum Threshold 1 Maximum Threshold Only the RX channels enabled in the static configuration APIs are monitored.	1 LSB = 1800mV/256, Valid range: TBD, maximum threshold >= minimum threshold Recommended value = TBD
rlUInt16_t	reserved1	Reserved	0
rlUInt32_t	reserved2	Reserved	0

};

10.6.111rlMcuClkCfg_t

This structure is container to mmwave radar device MCU Clock output configuration.

struct rlMcuClkCfg{

Type	Name	Description	Units
rlUInt8_t	mcuClkCtrl	This field controls the enable - disable of the MCU clock. Value Description 0x0 Disable MCU clock 0x1 Enable MCU clock	
rlUInt8_t	mcuClkSrc	This field specifies the source of the MCU clock. Applicable only in case of MCU clock enable. Else ignored. Value Description 0x0 XTAL(as connected to the device) 0x2 600MHz PLL divided clock	
rlUInt8_t	srcClkDiv	This field specifies the division factor to be applied to source clock. Applicable only in case of MCU clock enable. Else ignored. Value Description 0x0 Divide by 1 0x1 Divide by 2 0xFF Divide by 256	
rlUInt32_t	reserved	Reserved	0

};

10.6.112rlPmicClkCfg_t

This structure is container to mmwave radar device PMIC Clock output configuration.

struct rlPmicClkCfg{

Type	Name	Description	Units
rlUInt8_t	pmicClkCtrl	This field controls the	

		enable - disable of the PMIC clock. Value Description 0x0 Disable PMIC clock 0x1 Enable PMIC clock	
rlUInt8_t	pmicClkSrc	This field specifies the source of the PMIC clock. Applicable only in case of PMIC clock enable. Else ignored. Value Description 0x0 XTAL (as connected to the device) 0x2 600MHz PLL divided clock	
rlUInt8_t	srcClkDiv	his field specifies the division factor to be applie to source clock. Applicable only in case of PMIC clock enable. Else ignored. Value Description 0x0 Divide by 1 0x1 Divide by 2 … … 0xFF Divide by 256	
rlUInt8_t	modeSel	This field specifies the mode of operation for the PMIC clock generation. Applicable only in case of PMIC clock enable. Else ignored.	
rlUInt32_t	freqSlope	Applicable only in case of PMIC clock enable. Else ignored. Bit[25:0] – Frequency slope value to be applied in [8.18] format	1 LSB = 1/2^18
rlUInt8_t	minNdivVal	Applicable only in case of PMIC clock enable. Else ignored	

rlUInt8_t	maxNdivVal	Applicable only in case of PMIC clock enable. Else ignored.	
rlUInt8_t	clkDitherEn	Applicable only in case of PMIC clock enable. Else ignored.	
rlUInt8_t	reserved	Reserved	0

};

10.6.113rllatentFault_t

This structure is container to mmwave radar device latent fault test for the master sub-system.

struct rllatentFault{

Type	Name	Description	Units
rlUInt32_t	testEn1	Bits Definition 0 MibSPI self-test 1 DMA self-test 2 Watchdog self-test 3 RTI self-test 4 ESM self-test 5 EDMA self-test 6 CRC self-test 7 VIM self-test 8 MPU self-test 9 Mailbox self-test 10 LVDS pattern generation test 11 CSI2 pattern generation test 12 Generating NERROR 13 MibSPI single bit error test 14 MibSPI double bit error test 15 DMA Parity error	

		16 TCMA RAM single bit errors 17 TCMB RAM single bit errors 18 TCMA RAM double bit errors 19 TCMB RAM double bit errors 20 TCMA RAM parity errors. 21 TCMB RAM parity errors. 22 VIM lockstep test 23 CCM R4 lockstep test 24 DMA MPU Region tests 25 MSS Mailbox Single bit errors 26 MSS Mailbox double bit errors 27 BSS Mailbox Single bit errors 28 BSS Mailbox double bit errors 29 EDMA MPU test 30 EDMA parity test 31 CSI2 parity test	
rlUInt32_t	testEn2	Bits Definition 0 PBIST(VIM RAM/TCM RAM/MibSPI SRAM/Mailbox/EDMA/DMA/CSI2) 1 LBIST(VIM/CR4) 31:2 RESERVED	
rlUInt8_t	repMode	Value Definition 0 Report is sent every monitoring period	

		1 Report is send only upon a failure	
rlUInt8_t	testMode	Value Definition 0 Production mode. Latent faults are tested and any failures are reported 1 Characterization mode. Faults are injected and failures are reported which allows testing of the failure reporting	
rlUInt16_t	reserved	Reserved	0

};

10.6.114rlperiodicTest_t

This structure is container to mmwave radar periodicity test configuration for the master sub-system.

struct rlperiodicTest{

Type	Name	Description	Units
rlUInt16_t	periodicity	Periodicity at which tests need to be run	1 LSB = 1 ms
rlUInt32_t	testEn	Bit value definition: 1 - Enable, 0 - Disable Bit Monitoring type 0 Reriodic conf register read enable 1 ESM_MONITORING_EN [31:2] RESERVED	
rlUInt8_t	repMode	Value Definition 0 Report is sent every monitoring period 1 Report is sent only on a failure	
rlUInt8_t	Reserved0	Reserved	0
rlUInt16_t	Reserved1	Reserved	0

};

10.6.115rltestPattern_t

This structure is container to mmwave radar test pattern configuration.

struct rltestPattern{

Type	Name	Description	Units

rlUInt8_t	testPatGenCtrl	This field controls the enable-disable of the generation of the test pattern. Value Description 0x0 Disable test pattern generation 0x1 Enable test pattern generation	
rlUInt8_t	testPatGenTime	Number of system clocks (200 MHz) between successive samples for the test pattern gen. Applicable only in case of Test pattern enable. Else ignored.	
rlUInt16_t	testPatrnPktSize	Number of ADC samples to capture for each RX Valid range: 64 to MAX_NUM_SAMPLES, Where MAX_NUM_SAMPLES is such that all the enabled RX channel's data fits into 16 kB memory, with each sample consuming 2 bytes for real ADC output case and 4 bytes for complex 1x and complex 2x ADC output cases. For example: 4 RX, Complex ADC output -> MAX_NUM_SAMPLES = 1024 4 RX, Real ADC output -> MAX_NUM_SAMPLES = 2048 2 RX, Complex ADC output -> MAX_NUM_SAMPLES = 2048 2 RX, Real ADC output -> MAX_NUM_SAMPLES = 4096	

rlUInt16_t	numTestPtrnPkts	Number of test pattern packets to send, for infinite packets set it to 0	
rlUInt32_t	testPatRx0Icfg	This field specifies the values for Rx0, I channel. Applicable only in case of test pattern enable. Else ignored. Bits Description [15:0] Start offset value to be used for the first sample for the test pattern data [31:16] Value to be added for each successive sample for the test pattern data	
rlUInt32_t	testPatRx0Qcfg	This field specifies the values for Rx0, Q channel. Applicable only in case of test pattern enable. Else ignored. Bits Description [15:0] Start offset value to be used for the first sample for the test pattern data [31:16] Value to be added for each successive sample for the test pattern data	
rlUInt32_t	testPatRx1Icfg	This field specifies the values for Rx1, I channel. Applicable only in case of test pattern enable. Else ignored. Bits Description [15:0] Start offset value to be used for the first sample for the test pattern data	

		[31:16] Value to be added for each successive sample for the test pattern data	
rlUInt32_t	testPatRx1Qcfg	This field specifies the values for Rx1, Q channel. Applicable only in case of test pattern enable. Else ignored. Bits Description [15:0] Start offset value to be used for the first sample for the test pattern data [31:16] Value to be added for each successive sample for the test pattern data	
rlUInt32_t	testPatRx2Icfg	This field specifies the values for Rx2, I channel. Applicable only in case of test pattern enable. Else ignored. Bits Description [15:0] Start offset value to be used for the first sample for the test pattern data [31:16] Value to be added for each successive sample for the test pattern data	
rlUInt32_t	testPatRx2Qcfg	This field specifies the values for Rx2, Q channel. Applicable only in case of test pattern enable. Else ignored. Bits Description [15:0] Start offset value to be used for	

		the first sample for the test pattern data [31:16] Value to be added for each successive sample for the test pattern data	
rlUInt32_t	testPatRx3Icfg	This field specifies the values for Rx3, I channel. Applicable only in case of test pattern enable. Else ignored. Bits Description [15:0] Start offset value to be used for the first sample for the test pattern data [31:16] Value to be added for each successive sample for the test pattern data	
rlUInt32_t	testPatRx3Qcfg	This field specifies the values for Rx3, Q channel. Applicable only in case of test pattern enable. Else ignored. Bits Description [15:0] Start offset value to be used for the first sample for the test pattern data [31:16] Value to be added for each successive sample for the test pattern data	
rlUInt32_t	resereved	Revereved	0

```
};
```

10.6.116rlDynChirpCfg_t

This structure is container Dynamic chirp configuration for 16 chirp configurations.

```
struct rlDynChirpCfg{
```

Type	Name	Description	Units
rlUInt8_t	reserved0	Reserved	0
rlUInt8_t	chirpSegSel	Indicates the segment of the chirp RAM that the 16 chirps definitions in this sub block map to any of one segment out of 32 segments of SW chirp RAM. Valid range 0 to 31	-
rlUInt16_t	Reserved1	Reserved	0
rlChirpRow_t	chirpRow[16]	Chirp row configurations for 16 chirps.	

};

10.6.117rlChirpRow_t

This structure is container Dynamic chirp configuration for 16 chirp configurations.

struct rlChirpRow{

Type	Name	Description	Units
rlUInt32_t	chirpNR1	Nth Chirp config Row 1 Bits Definition 3:0 PROFILE_INDX Valid range 0 to 3 7:4 RESERVED 13:8 FREQ_SLOPE_VAR 1 LSB = 3.6e9*900 /2^26 ~48.279kHz Valid range: 0 to 63 15:14 RESERVED 18:16 TX_ENABLE Bit Definition b0 TX0 Enable b1 TX1 Enable b2 TX2 Enable 23:19 RESERVED 29:24 BPM_CONSTANT_BITS Bit Definition b0 CONST_BPM_VAL_TX0_OFF Value of Binary Phase Shift value for TX0, when during idle time b1 CONST_BPM_VAL_TX0_ON Value of Binary Phase Shift value for TX0, during chirp b2 CONST_BPM_VAL_TX1_OFF For TX1 b3 CONST_BPM_VAL_TX1_ON	-

		For TX1 b4 CONST_BPM_VAL_TX2_OFF For TX2 b5 CONST_BPM_VAL_TX2_ON For TX2 31:30 RESERVED	
rlUInt8_t	chirpNR2	Nth Chirp config Row 2 Bits Definition b22:0 FREQ_START_VAR 1 LSB = 3.6e9/2^26 ~53.644kHz Valid range: 0 to 8388607 b31:23 RESERVED	-
rlUInt16_t	chirpNR3	Nth Chirp config Row 3 Bits Definition b11:0 IDLE_TIME_VAR 1 LSB = 10 ns Valid range: 0 to 4095 b15:12 RESERVED b27:16 ADC_START_TIME_VAR 1 LSB = 10 ns Valid range: 0 to 4095 b31:28 RESERVED	

};

10.6.118rlDynChirpEnCfg_t

This structure is container to Dynamic chirp enable configuration.

struct rlDynChirpEnCfg {

Type	Name	Description	Units
rlUInt32_t	Reserved0	Reserved	0

};

10.6.119rlDynPerChirpPhShftCfg_t

This structure is container to Dynamic chirp enable configuration.

struct rlDynPerChirpPhShftCfg{

Type	Name	Description	Units
rlUInt8_t	Reserved0	Reserved	0

rlUInt8_t	chirpSegSel	Indicates the segment of the chirp RAM that the 16 chirps definitions in this sub block map to any of one segment out of 32 segments of SW chirp RAM. Valid range 0 to 31.	-
rlChirpPhShiftPerTx_t	phShiftPerTx[16U]	x phase shifter configurations for 16 chirps.	-
rlUInt16_t	Reserved1	Reserved	0

};

10.6.120 rlCalDataStore_t

This structure is container to calibration data which application will receive from radarSS and will feed in to the Device in next power up to avoid calibration.

struct rlCalDataStore{

Type	Name	Description	Units
rlUInt16_t	numOfChunk	Number of calibration data chunks Available in device.	
rlUInt16_t	chunkId	Current Calibration Data Chunk Id. Valid range 0-2.	-
rlUInt8_t	calData[224]	Calibration chunk Data	-

};

10.6.121 rlChirpPhShiftPerTx_t

This structure is container to Dynamic chirp enable configuration.

struct rlChirpPhShiftPerTx{

Type	Name	Description	Units
rlUInt8_t	chirpNTx0PhaseShifter	Nth Chirp TX0 phase shift value Bits definition b1:0 RESERVED (set it to 0b00) b7:2 TX0 phase shift value	1 LSB = 360/2^6 = 5.625 (degree)
rlUInt8_t	chirpNTx1PhaseShifter	Nth Chirp TX1 phase shift value Bits definition b1:0 RESERVED (set it to 0b00) b7:2 TX1 phase shift value	1 LSB = 360/2^6 = 5.625 (degree)
rlUInt8_t	chirpNTx2PhaseShifter	Nth Chirp TX2 phase shift value Bits definition b1:0 RESERVED (set it to 0b00) b7:2 TX2 phase shift value	1 LSB = 360/2^6 = 5.625 (degree)

};

10.6.122rlInterRxGainPhConf_t

This structure is container to Inter-Rx gain and phase offset configuration.

struct rlInterRxGainPhConf{

Type	Name	Description	Units
rlUInt8_t	profileIndx	This field indicates the profile Index for which this configuration applies.	-
rlUInt8_t	Reserved0	Reserved	0
rlUInt16_t	Reserved1	Reserved	0
rlUInt8_t	digRxGain[4]	One byte per RX (8-bit signed number) Byte Assignment 0 RX0 digital gain 1 RX1 digital gain 2 RX2 digital gain 3 RX3 digital gain	1 LSB = 0.1 dB Valid Range: -120 to 119
rlUInt16_t	digRxPhShift[4]	Two bytes per RX Bits Assignment b15:0 RX0 digital phase shift b31:16 RX1 digital phase shift b47:32 RX2 digital phase shift b63:48 RX3 digital phase shift NOTE: This field is NOT applicable when ADC_OUT_FMT is 00 (real output)	1 LSB = 360 degree / 2^16 ~ 0.0055 (degree) Valid Range: 0 to 65535
rlUInt32_t	Reserved2	Reserved	0
rlUInt32_t	Reserved3	Reserved	0

};

10.6.123rlRfBootStatusCfg_t

This structure is container to RadarSS Bootup status info.

struct rlRfBootStatusCfg{

Type	Name	Description	Units
rlUInt32_t	bssSysStatus	RadarSS bootup status Bit definition[1:pass,0:fail] 0 image CRC validation 1 CPU and VIM self-test status 2 Reserved 3 VIM test 4 STC self-test 5 CR4 STC 6 CRC test 7 Pampgen ECC 8 DFE Parity	-

		9 DFE ECC 10 Rampgen Lockstep 11 FRC lockstep 12 DFE PBIST 13 Rampgen lockstep 14 PBIST test 15 WDT test 16 ESM test 17 DFE STC 18 FRC test 19 TCM ECC 20 TCM parity 21 DCC test 22 SOCC test 23 GPADC test 24 FFT test 25 RTI test 26 PCR test 31:27 reserved	
rlUInt32_t	bssBootUpTime	RF BIST SS power up time	1 LSB = 5ns
rlUInt32_t	Reserved0	Reserved	0
rlUInt16_t	Reserved1	Reserved	0

};

10.6.124rlInterChirpBlkCtrlCfg_t

This structure is container to Inter Chirp block control configuration parameters.

struct rlRfBootStatusCfg{

Type	Name	Description	Units
rlUInt16_t	rx02RfTurnOffTime	Time to wait after ramp end before turning off RX0 and RX2 RF stages.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	Rx13RfTurnOffTime	Time to wait after ramp end before turning off RX1 and RX3 RF stages.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	rx02BbTurnOffTime	Time to wait after ramp end before turning off RX0 and RX2 baseband stages.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	Rx13BbTurnOffTime	Time to wait after ramp end before turning off RX1 and RX3 baseband stages.	1 LSB = 10 ns Valid

			range: -1024 to 1023
rlUInt16_t	rx02RfPreEnTime	Time before TX Start Time when RX0 and RX2 RF stages are to be put in fast-charge state.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	Rx13RfPreEnTime	Time before TX Start Time when RX1 and RX3 RF stages are to be put in fast-charge state.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	Rx02BbPreEnTime	Time before TX Start Time when RX1 and RX3 baseband stages are to be put in fast-charge state.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	rx13BbPreEnTime	Time before TX Start Time when RX1 and RX3 baseband stages are to be put in fast-charge state.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	rx02RfTurnOnTime	Time before TX Start Time when RX0 and RX2 RF stages are to be enabled.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	Rx13RfTurnOnTime	Time before TX Start Time when RX1 and RX3 RF stages are to be enabled.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	rx02BbTurnOnTime	Time before TX Start Time when RX0 and RX2 baseband stages are to be enabled.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	Rx13BbTurnOnTime	Time before TX Start Time when RX1 and RX3 baseband stages are to be enabled.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	rxLoChainTurnOffTime	Time to wait after ramp end before turning off RX LO chain.	1 LSB = 10 ns Valid range: -1024 to

			1023
rlUInt16_t	txLoChainTurnOffTime	Time to wait after ramp end before turning off TX LO chain.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	rxLoChainTurnOnTime	Time to wait after ramp end before turning on RX LO chain.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt16_t	txLoChainTurnOnTime	Time to wait after ramp end before turning on TX LO chain.	1 LSB = 10 ns Valid range: -1024 to 1023
rlUInt32_t	Reserved0	Reserved	0
rlUInt32_t	Reserved1	Reserved	0

};

10.6.125rlRfTxFreqPwrLimitMonConf_t

This structure is container to Tx freq and power limit configuration.

struct rlRfTxFreqPwrLimitMonConf{

Type	Name	Description	Units
rlUInt16_t	freqLimitLowTx0	The sensor's lower frequency limit for calibrations and monitoring for TX0 encoded in 2 bytes (16 bit unsigned number). Valid range: 7600 to 8100	1 LSB = 10 MHz
rlUInt16_t	freqLimitLowTx1	The sensor's lower frequency limit for calibrations and monitoring for TX1 encoded in 2 bytes (16 bit unsigned number). Valid range: 7600 to 8100	1 LSB = 10 MHz
rlUInt16_t	freqLimitLowTx2	The sensor's lower frequency limit for calibrations and monitoring for TX2 encoded in 2 bytes (16 bit unsigned number). Valid range: 7600 to 8100	1 LSB = 10 MHz
rlUInt16_t	freqLimitHighTx0	The sensor's higher frequency limit for calibrations and monitoring for TX0 encoded in 2 bytes (16 bit unsigned number). Valid range: 7600 to 8100 NOTE: FREQ_LIMIT_HIGH_TXn should be strictly greater than FREQ_LIMIT_LOW_TXn	1 LSB = 10 MHz

rlUInt16_t	freqLimitHighTx1	The sensor's higher frequency limit for calibrations and monitoring for TX1 encoded in 2 bytes (16 bit unsigned number). Valid range: 7600 to 8100 NOTE: FREQ_LIMIT_HIGH_TXn should be strictly greater than FREQ_LIMIT_LOW_TXn	1 LSB = 10 MHz
rlUInt16_t	freqLimitHighTx2	The sensor's higher frequency limit for calibrations and monitoring for TX2 encoded in 2 bytes (16 bit unsigned number). Valid range: 7600 to 8100 NOTE: FREQ_LIMIT_HIGH_TXn should be strictly greater than FREQ_LIMIT_LOW_TXn	1 LSB = 10 MHz
rlUInt8_t	tx0PwrBackOff	TX0 output power back off Valid Value: 0, 3, 6, 9	1LSB = 1dB
rlUInt8_t	tx1PwrBackOff	TX1 output power back off Valid Value: 0, 3, 6, 9	1LSB = 1dB
rlUInt8_t	tx2PwrBackOff	TX2 output power back off Valid Value: 0, 3, 6, 9	1LSB = 1dB
rlUInt8_t	Reserved0	Reserved	0
rlUInt16_t	Reserved1	Reserved	0
rlUInt16_t	Reserved2	Reserved	0
rlUInt16_t	Reserved3	Reserved	0
rlUInt16_t	Reserved4	Reserved	0

};

10.6.126rlAnaFaultInj_t

This structure is container to RF fault injection configuration.

struct 10.6.127 rlAnaFaultInj {

Type	Name	Description	Units
rlUInt8_t	reserved0	Reserved	0
rlUInt8_t	rxGainDrop	Primary Fault: RX Gain. This field indicates which RX RF sections should have fault injected. If the fault is enabled, the RX RF gain drops significantly. The fault can be used to cause significant gain change, inter-RX gain imbalance and an uncontrolled amount of inter-RX phase imbalance. Bit RX Channel 0 RX0 1 RX1 2 RX2 3 RX3	-

		Others RESERVED For each bit, 1 = inject fault, 0 = remove injected fault	
rlUInt8_t	rxPhInv	Primary Fault: RX Phase. This field indicates which RX channels should have fault injected. If the fault is enabled, the RX phase gets inverted. The fault can be used to cause a controlled amount (180o) of inter-RX phase imbalance. Bit RX Channel 0 RX0 1 RX1 2 RX2 3 RX3 Others RESERVED For each bit, 1 = inject fault, 0 = remove injected fault	-
rlUInt8_t	rxHighNoise	Primary Fault: RX Noise. This field indicates which RX channels should have fault injected. If the fault is enabled, the RX IFA square wave loopback paths are engaged to inject high noise at RX IFA input. The fault can be used to cause significant RX noise floor elevation. Bit RX Channel 0 RX0 1 RX1 2 RX2 3 RX3 Others RESERVED For each bit, 1 = inject fault, 0 = remove injected fault	-
rlUInt8_t	rxIfStagesFault	Primary Fault: Cutoff frequencies of RX IFA HPF & LPF, IFA Gain. This field indicates which RX channels should have fault injected. If the fault is enabled, the RX IFA HPF cutoff frequency becomes very high (about 15MHz). The fault can be used to cause the measured inband IFA gain, HPF and LPF attenuations to vary from ideal expectations. Bit RX Channel 0 RX0 1 RX1 2 RX2 3 RX3 Others RESERVED For each bit, 1 = inject fault, 0 = remove injected fault Note: during the execution of RX_IFSTAGE_MONITOR, the RX_HIGH_NOISE faults are temporarily removed.	-

rlUInt8_t	rxLoAmpFault	Primary Fault: RX Mixer LO input swing reduction. This field indicates which RX channels should have fault injected. If the fault is enabled, the RX mixer LO input swing is significantly reduced. The fault is primarily expected to be detected by RX_INTERNAL_ANALOG_SIGNALS_MONITOR (under PWRDET_RX category). Bit Channel 0 RX0 and RX1 1 RX2 and RX3 Others RESERVED For each bit, 1 = inject fault, 0 = remove injected fault	-
rlUInt8_t	txLoAmpFault	Primary Fault: TX PA input signal generator turning off. This field indicates which TX channels should have fault injected. If the fault is enabled, the amplifier generating TX power amplifiers LO input signal is turned off. The fault is primarily expected to be detected by TX<n>_INTERNAL_ANALOG_SIGNALS_MONITOR (under DCBIAS category). Bit Channel 0 TX0 and TX1 1 TX2 (applicable only if available in the xWR device) Others RESERVED For each bit, 1 = inject fault, 0 = remove injected fault	-
rlUInt8_t	txGainDrop	Primary Fault: TX Gain (power). This field indicates which TX RF sections should have fault injected. If the fault is enabled, the TX RF gain drops significantly. The fault can be used to cause significant TX output power change, inter-TX gain imbalance and an uncontrolled amount of inter-TX phase imbalance. Bit TX Channel 0 TX0 1 TX1 2 TX2 Others RESERVED For each bit, 1 = inject fault, 0 = remove injected fault	-
rlUInt8_t	txPhInv	Primary Fault: TX Phase. This field indicates if TX channels should have fault injected, along with some further programmability. If the fault is enabled, the TX BPM polarity (phase) is forced to a constant value as programmed. The fault can be used	-

		to cause a controlled amount (180 degree) of inter-TX phase imbalance as well as BPM functionality failure. Bit TX Channel 0 TX fault (Common for all TX channels) 1 RESERVED 2 RESERVED 3 TX0 BPM VALUE 4 TX1 BPM VALUE 5 TX2 BPM VALUE Others RESERVED For each TX<n> BPM VALUE: Applicable only if TX FAULT is enabled. \n Value = 0: force TX<n> BPM polarity to 0 \n Value = 1: force TX<n> BPM polarity to 1.	
rlUInt8_t	synthFault	Primary Fault: Synthesizer Frequency. This field indicates which Synthesizer faults should be injected. SYNTH_VCO_OPENLOOP: If the fault is enabled, the synthesizer is forced in open loop mode with the VCO control voltage forced to a constant. In order to avoid out of band emissions in this faulty state, this fault is injected just before the PLL_CONTROL_VOLTAGE_MONITOR is executed and released just after its completion. SYNTH_FREQ_MON_OFFSET: If the fault is enabled, the synthesizer frequency monitor's ideal frequency ramp waveform is forced to be offset from the actual ramp waveform by a constant, causing monitoring to detect failures. Bit Enable Fault 0 SYNTH_VCO_OPENLOOP 1 SYNTH_FREQ_MON_OFFSET Others RESERVED For each bit, 1 = inject fault, 0 = remove injected fault	-
rlUInt8_t	supplyLdoFault	This field indicates whether some LDO output voltage faults should be injected or not Bit Enable Fault 0 SUPPLY_LDO_RX_LODIST_FAULT Others RESERVED SUPPLY_LDO_RX_LODIST_FAULT: if enabled, the RX LO distribution sub	-

		system's LDO output voltage is slightly changed compared to normal levels to cause INTERNAL_PMCLKLO_SIGNALS_MONITOR to detect failure (under SUPPLY category). For each bit, 1 = inject fault, 0 = remove injected fault	
rlUInt8_t	miscFault	This field indicates whether a few miscellaneous faults should be injected or not. Bit Enable Fault 0 GPADC_CLK_FREQ_FAULT Others RESERVED GPADC_CLK_FREQ_FAULT: if enabled, the GPADC clock frequency is slightly increased compared to normal usage to cause BSS DCC_CLOCK_FREQ_MONITOR to detect failure. For each bit, 1 = inject fault, 0 = remove injected fault	-
rlUInt8_t	miscThreshFault	This field indicates whether faults should be forced in the threshold comparisons in the software layer of some monitors. If a fault is enabled, the logic in the min-max threshold comparisons used for failure detection is inverted, causing a fault to be reported. During these faults, no hardware fault condition is injected in the device. Bit Enable Fault 0 EXTERNAL_ANALOG_SIGNALS_MONITOR 1 GPADC_INTERNAL_SIGNALS_MONITOR Others RESERVED For each bit, 1 = inject fault, 0 = remove injected fault	-
rlUInt8_t	Reserved1	Reserved	0
rlUInt16_t	Reserved2	Reserved	0
rlUInt16_t	Reserved3	Reserved	0
rlUInt16_t	Reserved4	Reserved	0

};

10.7 Sample Application

For sample application please refer DFP (device firmware package) user guide document.