API

UWB UCI Message API

Qorvo

Release R12.7.0-288-gb8203d55886

Contents

1 Glossary

ADC
Analog Digital Converter

AEAD
Authenticated Encryption with Associated Data

AES
Advanced Encryption Standard

AoA
Angle of Arrival

AIDL
Android Interface Definition Language

AOSP
Android Open Source Project

APDU
Application Protocol Data Unit

API
Application Programming Interface

BPRF
Base Pulse Repetition Frequency

CAP
Contention Access Period

CCC
Car Connectivity Consortium

CCM
Counter with Cipher Block Chaining Message Authentication Code

CFO
Clock Frequency Offset

CFP
Contention Free Period

CIR
Channel Impulse Response

CM
Control Message

CMAC
Cipher-Based Message Authentication Code

CRUM
Control Update Message

DL-TDoA
Downlink TDoA

DPF
Data Packet Format

DRBG
Deterministic Random Bit Generator

DS
Device Specific

DS-TWR
Double-Sided Two-Way Ranging

DTM
Downlink TDoA Message

DUT
Device Under Test

ECB
Electronic Code Book

EVB
Evaluation Board

FBS
FiRa Based Session

FoM
Figure of Merit

FP
First Path

GID
Group IDentifier

GPIO
General Purpose Input Output

HAL
Hardware Abstraction Layer

HIE
Header Information Element

HPRF
High Pulse Repetition Frequency

I2C
Inter-Integrated Circuit

IE
Information Element

IFI
Inter-Frame-Interval

IFI_GT
Inter-Frame-Interval Guard Time

IFI_BGT
Inter-Frame-Interval Block Guard Time

KDF
Key Derivation Function

HUS
Hybrid UWB Scheduling

L1
Layer One

LNA
Low Noise Amplifier

LLHW
Low Level Hardware

LUT
Lookup Table

LO
Local Oscillator

IV
Initialization Vector

MAC
Medium Access Control

MCU
MicroController Unit

MHR
MAC Header

MRM
Measurement Report Message

MRP
Measurement Report Phase

MTI
Moving Target Indicator

NA
Not Applicable

NL
NetLink

NONCE
Number used Once

OID
Opcode IDentifier

OOB
Out-Of-Band

OUI
Organizationally Unique Identifier

OWR
One-Way Ranging

PA
Power Amplifier

PDoA
Phase Difference of Arrival

PHR
Physical Layer Header

PHY
Physical Layer

PIE
Payload Information Element

PRF
Pulse Repetition Frequency

PSDU
PHY Service Data Unit

PSR
Preamble Symbol Repetitions

RAM
Random Access Memory

RCP
Ranging Control Phase

RDS
Ranging Data Set

RFFE
Radio Frequency Front End

RFM
Ranging Final Message

RFP
Ranging Final Phase

RFRAME
Ranging Frame

RFU
Reserved for Future Use

RIM
Ranging Initiation Message

RIP
Ranging Initiation Phase

RP
Ranging Phase

RRM
Ranging Response Message

RRP
Ranging Response Phase

RRRM
Ranging Result Report Message

RSL
Received Signal Level

RSSI
Received Signal Strength Indicator

S1
ACPI power state corresponding to CPU Idle

S3
ACPI power state corresponding to suspend to RAM

Sample
One complex value from CIR array

SE
Secure Element

SHR
Synchronization Header

SIP
System In a Package

SNR
Signal to Noise Ratio

SOC
System On a Chip

SPI
Serial Peripheral Interface

SS-TWR
Single-Sided Two-Way Ranging

STS
Scrambled Timestamp Sequence

SUS
Secure UWB Service

SYNC
Synchronization Preamble Sequence

TDoA
Time Difference of Arrival

TLV
Type Length Value

ToA
Time of Arrival

ToF
Time of Flight

TWR
Two Way Ranging

UCI
UWB Subsystem Command Interface

UL-TDoA
Uplink TDoA

UWB
Ultra-Wide Band

UWBS
Ultra Wide-Band Subsystem

2 Overview

UCI is used to control the *UWBS*. The protocol and its mandatory messages are defined in [FiR23a]. It relies on messages that are exchanged between a client and a server. The communication is symmetric, all messages can be sent or received by the two peers. Messages are put in *UCI* packets described in the specification.

There are two types of packets:

- Control packets are used to exchange commands between the client and the server (i.e. it conveys control plan information).
- Data packets are used to exchange user data between the client and the server (i.e. it conveys user plan information).

Note: This documentation is focused on Control packets content.

2.1 Control packets content

Content of control packets is named Control messages that can be of different types:

- **Command messages**. They are used to request the execution of a given command
- **Response messages**. They are used to respond to a request message (confirm the good reception of the command).
- **Notification messages**. They are used to notify the peer entity of something without needing a response.

The protocol is simple and is based on a request/response model. When a peer sends a request message to the other peer, it must wait for a response message before sending another request message. The response message must be sent by the peer that received the request message.

Notification messages are sent by a peer and do not mandate to wait for a response message.

Each control messages can have different content. In order to identify them, two elements are used:

- a *GID*: it identifies a group of messages and is coded on 4 bits
- an *OID*: it identifies a message within a group and is coded on 6 bits

Some *GID*s have defined in the specification and must be implemented. Others are vendor specific and can be used to add new messages to the protocol. Within a standard *GID*, no vendor specific *OID*s are available.

In the Qorvo *UWB* Subsystem 3 specific *GID*s have been defined:

- The Qorvo MCPS *GID*. It is used to implement the messages aimed to be processed by the MCPS itself and does not contain any protocol or function related messages.
- The Qorvo Extensions *GID*s. It contains all the messages considered as extensions of protocols. It can be standard protocol extensions (like FiRa) or Qorvo proprietary protocol extensions. There are 2 different *GID*s for extensions and no specific rule to add messages to either of the groups

2.2 FiRa specificities

The Qorvo *UWB* Subsystem support multiple different versions of the FiRa standard. Only on version is supported by a given binary as the version selection is done through a compile flag. The current supported version are:

- FiRa 1.x
- FiRa 2.x

When format of the messages changes between the two versions, it is indicated in the corresponding chapters of this documentation.

The field `Session ID` has been changed in all messages. For sake of clarity, the messages where not duplicated in all messages as its size did not changed.

The FiRa related chapters of this documentation contains only the format of the messages and, when needed, precision about specific behavior of the Qorvo *UWB* Subsystem when handling them. If more information is needed about a messages or its handling, please refer to (see [FiR23a]).

3 Supported messages List

This chapter covers the different control messages supported by the Qorvo *UWB* Subsystem regrouped in *GID and OID definitions for control messages*.

Messages are named using the following convention: `<message_id>[_CMD][_RSP][_NTF]`. The three letters at the end gives the type of messages for which the OID is valid (Command, Response or Notification). Note also that the *GID*/*OID* pair can be the same between the Command/Response/Notification messages when they are about the same subject.

Warning: If those messages need to be changed, it must be validated with the architecture team. Please make sure of an architecture team member is included in the MR as a reviewer.

Warning: The following messages are supported directly by the Qorvo *UWB* Subsystem. Additional messages can be supported by external *HAL*s which shall translate them into one of the following.

Warning: The messages `DATA_MESSAGE_SND` and `DATA_MESSAGE_RCV` does not appear in *GID and OID definitions for control messages* as they are not control messages.

Table 3.1: GID and OID definitions for control messages

GID	OID	Message name	Description
UCI Core 0x0	0x00	CORE_DEVICE_RESET_CMD	Reset device
		CORE_DEVICE_RESET_RSP	Reset device Ack
	0x01	CORE_DEVICE_STATUS_NTF	Device status notification
	0x02	CORE_GET_DEVICE_INFO_CMD	Get device information
		CORE_GET_DEVICE_INFO_RSP	Get device information Ack
	0x03	CORE_GET_CAPS_INFO_CMD	Get device capabilities
		CORE_GET_CAPS_INFO_RSP	Get device capabilities Ack
	0x04	CORE_SET_CONFIG_CMD	Set device configuration
		CORE_SET_CONFIG_RSP	Set device configuration Ack
	0x05	CORE_GET_CONFIG_CMD	Get device configuration
		CORE_GET_CONFIG_RSP	Get device configuration Ack
	0x06	Reserved for future usage	
	0x07	CORE_GENERIC_ERROR_NTF	Generic error notification
	0x08	*[Not supported in QM33 SDK]* CORE_QUERY_UWBS_TIMESTAMP_CMD	Get device current timestamp
		[Not supported in QM33 SDK] CORE_QUERY_UWBS_TIMESTAMP_RSP	Get device current timestamp Ack
	0x09 - 0xFF	Reserved for future usage of FiRa	
UCI Session Config 0x1	0x00	SESSION_INIT_CMD	Init a session
		SESSION_INIT_RSP	Init a session Ack
	0x01	SESSION_DEINIT_CMD	Deinit a session
		SESSION_DEINIT_RSP	Deinit a session Ack
	0x02	SESSION_STATUS_NTF	Device status notification
	0x03	SESSION_SET_APP_CONFIG_CMD	Configure a session
		SESSION_SET_APP_CONFIG_RSP	Configure a session Ack
	0x04	SESSION_GET_APP_CONFIG_CMD	Get parameters of a session
		SESSION_GET_APP_CONFIG_RSP	Get parameters of a session Ack
	0x05	*[Not supported in QM33 SDK]* SESSION_GET_COUNT_CMD	Get number of sessions created
		[Not supported in QM33 SDK] SESSION_GET_COUNT_RSP	Get number of sessions created Ack
	0x06	*[Not supported in QM33 SDK]* SESSION_GET_STATE_CMD	Get the state of a session
		[Not supported in QM33 SDK] SESSION_GET_STATE_RSP	Get the state of a session Ack

continues on next page

Table 3.1 – continued from previous page

GID	OID	Message name	Description
	0x07	SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_CMD	Update controlees list of a session
		SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_RSP	Update controlees list of a session Ack
		SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_NTF	Controlees list updated confirmation notification
	0x08	*[Not supported in QM33 SDK]* SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_CMD	Update controlees list of a session
		[Not supported in QM33 SDK] SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_RSP	Update controlees list of a session Ack
	0x09	*[Not supported in QM33 SDK]* SESSION_UPDATE_DT_TAG_RANGING_ROUNDS_CMD	Update controlees list of a session
		[Not supported in QM33 SDK] SESSION_UPDATE_DT_TAG_RANGING_ROUNDS_CMD	Update controlees list of a session Ack
	0x0A	*[Not supported in QM33 SDK]* SESSION_QUERY_DATA_SIZE_IN_RANGING_CMD	Update controlees list of a session
		[Not supported in QM33 SDK] SESSION_QUERY_DATA_SIZE_IN_RANGING_RSP	Update controlees list of a session Ack
	0x0B	*[Not supported in QM33 SDK]* SESSION_SET_HUS_CONTROLLER_CONFIG_CMD	Update HUS configuration of a controller
		[Not supported in QM33 SDK] SESSION_SET_HUS_CONTROLLER_CONFIG_RSP	Update HUS configuration of a controller Ack
	0x0C	*[Not supported in QM33 SDK]* SESSION_SET_HUS_CONTROLEE_CONFIG_CMD	Update HUS configuration of a controller
		[Not supported in QM33 SDK] SESSION_SET_HUS_CONTROLEE_CONFIG_RSP	Update controlees list of a session Ack
	0x0D - 0xFF	Reserved for future usage of FiRa	
UCI Session Control 0x2	0x00	SESSION_START_CMD	Start ranging session
		SESSION_START_RSP	Start ranging session Ack
		SESSION_INFO_NTF	Ranging session notification
	0x01	SESSION_STOP_CMD	Stop ranging session
		SESSION_STOP_RSP	Stop ranging session Ack
	0x02	Reserved for future usage of FiRa	
	0x03	*[Not supported in QM33 SDK]* SESSION_GET_RANGING_COUNT_CMD	Get number of attempted ranging
		[Not supported in QM33 SDK] SESSION_GET_RANGING_COUNT_RSP	Get number of attempted ranging Ack
	0x04	*[Not supported in QM33 SDK]* SESSION_DATA_CREDIT_NTF	Data credit availability notification

continues on next page

Table 3.1 – continued from previous page

GID	OID	Message name	Description
	0x05	*[Not supported in QM33 SDK]* SESSION_DATA_TRANSFER_STATUS_NTF	Data tranfer state notification
	0x06 - 0xFF	Reserved for future usage of FiRa	
0x3 - 0x8	Reserved for future usage of FiRa		
UCI Qorvo Extensions 1 0x09	0x00 - 0xFF	Available for future usage	
0xA	Available for future usage		
UCI Qorvo Extensions 2 0xB	0x00	QORVO_TEST_DEBUG_NTF	Debug notification
	0x01	QORVO_TEST_TX_CW_CMD	Control continuous wave test
		QORVO_TEST_TX_CW_RSP	Control continuous wave test Ack
	0x02	QORVO_TEST_PLLRF_CMD	Control PLL test
		QORVO_TEST_PLLRF_RSP	Control PLL test Ack
		QORVO_TEST_PLLRF_NTF	Control PLL test notification
	0x03	QORVO_FIRA_RANGE_DIAGNOSTICS_NTF	FiRa diagnostic notification
	0x04 - 0x06	Available for future usage	
	0x07 - 0x27	Reserved for future usage	
	0x28 - 0xFF	Available for future usage	
UCI Android 0x0C	0x00 - 0xFF	Available for future usage	
UCI Test 0xD	0x00	TEST_CONFIG_SET_CMD	Set test configuration
		TEST_CONFIG_SET_RSP	Set test configuration Ack
	0x01	TEST_CONFIG_GET_CMD	Get test configuration
		TEST_CONFIG_GET_RSP	Get test configuration Ack
	0x02	TEST_PERIODIC_TX_CMD	Run periodic TX test
		TEST_PERIODIC_TX_RSP	Run periodic TX test Ack
		TEST_PERIODIC_TX_NTF	Periodic TX test result notification
	0x03	TEST_PER_RX_CMD	Run PER RX test
		TEST_PER_RX_RSP	Run PER RX test Ack
		TEST_PER_RX_NTF	PER RX test result notification
	0x04	Reserved for future usage	
	0x05	TEST_RX_CMD	Run RX test
		TEST_RX_RSP	Run RX test Ack
		TEST_RX_NTF	RX test result notification
	0x06	Reserved for future usage	

continues on next page

Table 3.1 – continued from previous page

GID	OID	Message name	Description
	0x07	TEST_STOP_SESSION_CMD	Stop PCTT session
		TEST_STOP_SESSION_RSP	Stop PCTT session Ack
	0x08	QORVO_TEST_SS_TWR_CMD	Start PCTT test session for SS-TWR
		QORVO_TEST_SS_TWR_RSP	Start PCTT test session for SS-TWR Ack
		QORVO_TEST_SS_TWR_NTF	PCTT test session for SS-TWR notification
	0x09 - 0xFF	Available for future usage	
UCI Qorvo MAC 0x0E	0x00 - 0x29	Available for future usage	
	0x2A	QORVO_MAC_SET_CALIBRATIONS_CMD	Set calibration values
		QORVO_MAC_SET_CALIBRATIONS_RSP	Set calibration values Ack
	0x2B	QORVO_MAC_GET_CALIBRATIONS_CMD	Get calibration values
		QORVO_MAC_GET_CALIBRATIONS_RSP	Get calibration values Ack
	0x2C - 0xFF	Available for future usage	
UCI Qorvo Calib 0x0F	0x00	QORVO_CALIB_RESET_CMD	Reset the calibration and configuration
		QORVO_CALIB_RESET_RSP	Reset the calibration and configuration Ack
	0x01 - 0xFF	Available for future usage	

4 UCI Core GID

This *GID* and all its *OID*s are part of FiRa standard (see [FiR23a]). description of the messages is based on the specification and it is highly recommended to get additional information in the latter if needed.

4.1 Resetting device

Device can be reset using the `CORE_DEVICE_RESET_CMD` message illustrated in Table 4.1. The returned `CORE_DEVICE_RESET_RSP` message is illustrated in Table 4.2.

Table 4.1: `CORE_DEVICE_RESET_CMD`

Size (octet)	Field
1	Reset configuration, must be 1

Table 4.2: `CORE_DEVICE_RESET_RSP`

Size (octet)	Field
1	Status

4.2 Getting Device Information

Device wide information are retrieved using the `CORE_GET_DEVICE_INFO_CMD` message. The latter has no data. Its corresponding response message named `CORE_GET_DEVICE_INFO_CMD` is illustrated in Table 4.3.

Note: Current implementation of `CORE_GET_DEVICE_INFO_CMD` returns UCI generic version based on compile flags. If `UCI_USE_FIRA20` is enabled in *cmake*, the version returned is 2.0.0.

When the FiRa protocol is not included in the device, FiRa MAC, PHY and UCI test versions are set to `0`. FiRa UCI generic version is the version on which UCI transport is based.

Table 4.3: `CORE_GET_DEVICE_INFO_RSP`

Size (octet)	Field
1	Status
2	FiRa UCI generic version
2	FiRa MAC version
2	FiRa PHY version
2	FiRa UCI test version
1	Vendor specific information length
n	Vendor specific information

The content of the vendor specific information is defined by the vendor. on Qorvo *UWB* Subsystem the format described in Table 4.4 is used.

Table 4.4: CORE_GET_DEVICE_INFO_RSP vendor specific information

Size (octet)	Field
1	Internal firmware version, major number part
1	Internal firmware version, minor number part
1	Internal firmware version, patch number part
1	Internal firmware version, RC number part
8	Unique firmware build identifier
1	Product firmware version, major number part
1	Product firmware version, minor number part
1	Product firmware version, patch number part
32	Unique chip identifier
4	Device identifier
1	Package identifier, 0 for SoC, 1 for SIP

4.3 Notifying Device Status Notification

When a device changes its state, it sends a `CORE_DEVICE_STATUS_NTF` message illustrated in Table 4.3. The different values are illustrated in Table 4.6.

Table 4.5: CORE_DEVICE_STATUS_NTF

Size (octet)	Field
1	Device state

Table 4.6: Device state values

Value	Description
0x00	Reserved for future usage
0x01	`DEVICE_STATE_READY`: device is initialized and ready
0x02	`DEVICE_STATE_ACTIVE`: device is started
0x03 - 0xFE	Reserved for future usage
0xFF	`DEVICE_STATE_ERROR`: unrecoverable error state

4.4 Getting Device Capabilities

Device capabilities can be retrieved using the `CORE_GET_CAPS_INFO_CMD` message (not illustrated as it does not contain any data). Its corresponding response message named `CORE_GET_CAPS_INFO_RSP` is illustrated in Table 4.7.

Note: Current implementation of `CORE_GET_CAPS_INFO_RSP` and its content is based on CR287, revision 4 and is not aligned with FiRa UCI 2.0 specification.

Table 4.7: CORE_GET_CAPS_INFO_RSP

Size (octet)	Field
1	Status
1	Number of capabilites (n)
$\sum_{i=1}^{n} len_{tlv_i}$	n capabilities as TLV form as described in Table 4.8

Table 4.8: CORE_GET_CAPS_INFO_RSP capability TLV

Size (octets)	Field
1	type
1	length
var	value

The content of the capability parameters is illustrated in *UCI Core GID*.

4.5 Managing Device Configuration

Device wide parameters can be set using the CORE_SET_CONFIG_CMD message illustrated in Table 4.9. Its corresponding response message named CORE_SET_CONFIG_RSP is illustrated in Table 4.11.

Table 4.9: CORE_SET_CONFIG_CMD

Size (octet)	Field
1	Number of parameters (n)
$\sum_{i=1}^{n} len_{param_i}$	List of n parameters in TLV form as described in Table 4.10

Table 4.10: CORE_SET_CONFIG_CMD parameter TLV

Size (octets)	Field
1	type
1	length (reset the parameter to default if 0)
var	value

Table 4.11: CORE_SET_CONFIG_RSP

Size (octet)	Field
1	Status
1	Number of statuses (n)
$\sum_{i=1}^{n} len_{status_i}$	List of n statuses as described in Table 4.12

Table 4.12: CORE_SET_CONFIG_RSP status

Size (octets)	Field
1	type
1	status

Parameters used in both messages are defined in Table 7.1.

Device wide parameters can be retrieved using the CORE_GET_CONFIG_CMD message illustrated in Table 4.13. Its corresponding response message named CORE_GET_CONFIG_RSP is illustrated in Table 4.14.

Table 4.13: `CORE_GET_CONFIG_CMD`

Size (octet)	Field
1	Number of parameters (n)
n	List of n parameter types

Table 4.14: `CORE_GET_CONFIG_RSP`

Size (octet)	Field
1	Status
1	Number of parameters (n)
$\sum_{i=1}^{n} len_{param_i}$	List of n parameters in TLV form as described in Table 4.15

Table 4.15: `CORE_GET_CONFIG_RSP` parameter TLV

Size (octets)	Field
1	type
1	length (0 if omitted)
var	value

Parameters used in both messages can be found in Table 7.1.

4.6 Notifying Generic Error

When a general error occurs in the *UWBS*, it can send the `CORE_GENERIC_ERROR_NTF` illustrated in Table 4.16.

Table 4.16: `CORE_GENERIC_ERROR_NTF`

Size (octet)	Field
1	Status

4.7 *[Not supported in QM33 SDK]* Retrieving current UWBS time

In some cases, applications need to get the absolute *UWBS* time reference. The latter can be retrieved using the `CORE_QUERY_UWBS_TIMESTAMP_CMD` message (not illustrated as it does not contain any information). Its corresponding response message named `CORE_QUERY_UWBS_TIMESTAMP_RSP` is illustrated in Table 4.17.

Table 4.17: `CORE_QUERY_UWBS_TIMESTAMP_RSP`

Size (octet)	Field
1	Status
8	Time in ns

5 UCI Session Config GID

This *GID*s and all its *OID*s are part of FiRa standard (see [FiR23a]). description of the messages is based on the specification and it is highly recommended to get additional information in the latter if needed.

If more information is needed about fields of those messages, please refer (see [FiR23a]). Initializing and de-initialize sessions

5.1 Initializing and de-initializing sessions

To initialize a session, an application shall use the SESSION_INIT_CMD illustrated in Table 5.1 and its corresponding response message SESSION_INIT_RSP illustrated in Table 5.2.

Table 5.1: SESSION_INIT_CMD

Size (octet)	Field
4	Session ID
1	Session type

Table 5.2: SESSION_INIT_CMD

Size (octet)	Field
1	Status
4	Session Handle

The value supported for the Session type field are defined in Table 5.3.

Table 5.3: Supported session types

Value	Description
0x00	Ranging session (no in-band data)
0x01 - 0x9F	Reserved for FiRa
0xA0 - 0xCF	Vendor specific
0xD0	Test Mode
0xD1 - 0xDF	Reserved for FiRa
0xE0 - 0xFF	Vendor specific

To de-initialize a session, an application shall use the SESSION_DEINIT_CMD illustrated in Table 5.4 and its corresponding response message SESSION_DEINIT_RSP illustrated in Table 5.5.

Table 5.4: SESSION_DEINIT_CMD

Size (octet)	Field
4	Session Handle

Table 5.5: SESSION_DEINIT_RSP

Size (octet)	Field
1	Status

[Not supported in QM33 SDK] It is possible to retrieve the current number of sessions using the `SESSION_GET_COUNT_CMD` message with its corresponding response message `SESSION_GET_COUNT_RSP` respectively illustrated in in Table 5.6 and Table 5.7.

Table 5.6: `SESSION_GET_COUNT_CMD`

Size (octet)	Field
N/A	None

Table 5.7: `SESSION_GET_COUNT_RSP`

Size (octet)	Field
1	Status
1	Number of sessions

5.2 Controlling Sessions

Sessions can be started and stopped using the `SESSION_START_CMD/SESSION_START_RSP` and `SESSION_STOP_CMD/SESSION_STOP_RSP` messages. Those messages are illustrated in Table 5.8 Table 5.9 Table 5.10 and Table 5.11 respectively.

Table 5.8: `SESSION_START_CMD`

Size (octet)	Field
4	Session Handle

Table 5.9: `SESSION_START_RSP`

Size (octet)	Field
1	Status

Table 5.10: `SESSION_STOP_CMD`

Size (octet)	Field
4	Session Handle

Table 5.11: `SESSION_STOP_RSP`

Size (octet)	Field
1	Status

[Not supported in QM33 SDK] It is possible to retrieve the current state of a session using the `SESSION_GET_STATE_CMD` message with its corresponding response message `SESSION_GET_STATE_RSP` respectively illustrated in in Table 5.12 and Table 5.12.

Table 5.12: `SESSION_GET_STATE_CMD`

Size (octet)	Field
4	Session Handle

Table 5.13: SESSION_GET_STATE_RSP

Size (octet)	Field
1	Status
1	Session state

The SESSION_STATUS_NTF message is used to notify the application of a session status change and is implemented as illustrated in Table 5.14.

Table 5.14: SESSION_STATUS_NTF

Size (octet)	Field
4	Session Handle
1	Session status
1	Reason code

5.3 Managing Session Configuration

Configuration of an existing session is done using the SESSION_SET_APP_CONFIG_CMD message illustrated in Table 5.15. The returned SESSION_SET_APP_CONFIG_RSP message is illustrated in Table 5.17.

Table 5.15: SESSION_SET_APP_CONFIG_CMD

Size (octet)	Field
4	Session Handle
1	Number of parameters (n)
$\sum_{i=1}^{n} len_{param_i}$	List of n parameters in TLV form as described in Table 5.16

Table 5.16: SESSION_SET_APP_CONFIG_CMD parameter TLV

Size (octets)	Field
1	type
1	length (reset to default if 0)
var	value

Table 5.17: SESSION_SET_APP_CONFIG_RSP

Size (octet)	Field
1	Status
1	Number of failed parameters (n)
$\sum_{i=1}^{n} len_{param_i}$	List of n parameters as described in Table 5.18

Table 5.18: SESSION_SET_APP_CONFIG_RSP status

Size (octets)	Field
1	type
1	status

The parameters used in both messages are defined in Table 7.2.

It is possible to retrieve the session configuration using the SESSION_GET_APP_CONFIG_CMD message illustrated in Table 5.19. Its corresponding response message SESSION_GET_APP_CONFIG_RSP is illustrated in Table 5.21.

Table 5.19: SESSION_GET_APP_CONFIG_CMD

Size (octet)	Field
4	Session Handle
1	Number of parameters (n)
$\sum_{i=1}^{n} len_{param_i}$	List of n parameters as described in Table 5.20

Table 5.20: SESSION_GET_APP_CONFIG_CMD type

Size (octets)	Field
1	type

Table 5.21: SESSION_GET_APP_CONFIG_RSP

Size (octet)	Field
1	Status
1	Number of parameters (n)
$\sum_{i=1}^{n} len_{param_i}$	List of n parameters as described in Table 5.20

Table 5.22: SESSION_SET_APP_CONFIG_RSP parameter TLV

Size (octets)	Field
1	type
1	length (0 if omitted)
var	value

The parameters used in both messages are defined in Table 7.2.

5.4 Managing FiRa multicast TWR ranging session configuration

When a session is a multicast TWR ranging session, it is possible to configure those sessions using the messages SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_CMD and its corresponding SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_RSP.

..note:: The content of the SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_RSP changed on FiRA 2.0 and is not backward compatible.

The messages are illustrated respectively in Table 5.23, Table 5.25, Table 5.27. and Table 5.28.

Table 5.23: SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_CMD prior to FiRa 2.0

Size (octet) Field	
4	Session ID
1	Action
1	Number of controlees (n)
$\sum_{i=1}^{n} len_{controlee_i}$	List of n controlees as described in Table 5.24

Table 5.24: SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_CMD controlee prior to FiRa 2.0

Size (octets)	Field
2	short address
4	sub-session id

Table 5.25: SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_CMD from FiRa 2.0

Size (octet)	Field
4	Session Handle
1	Action
1	Number of controlees (n)
$\sum_{i=1}^{n} len_{controlee_i}$	List of n controlees as described in Table 5.26

Table 5.26: SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_CMD controlee from FiRa 2.0

Size (octets)	Field
2	short address
4	sub-session id
0|16|32	sub-session key

Table 5.27: SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_RSP prior to FiRa 2.0

Size (octet)	Field
1	Status

Table 5.28: SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_RSP after FiRa 2.0

Size (octet)	Field
1	Status
1	Number of statuses (n)
$\sum_{i=1}^{n} len_{status_i}$	List of n status as described in Table 5.29

Table 5.29: `SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_RSP` status from FiRa 2.0

Size (octets)	Field
2	short address
1	status

When the update of the multicast list is done, the Qorvo *UWB* Subsystem must produce a `SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_NTF` message to notify the application of the change.

..note:: The content of the `SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_NTF` changed on FiRA 2.0 and is not backward compatible.

This message is illustrated in Table 5.30 and Table 5.32

Table 5.30: `SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_NTF` prior to FiRa 2.0

Size (octet) Field	
4	Session ID
1	Remaining multicast list size
1	Number of controlee statuses (n)
$\sum_{i=1}^{n} len_{controlee_i}$	List of n controlees as described in Table 5.31

Table 5.31: `SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_NTF` status from FiRa 2.0

Size (octets)	Field
2	controlee short address
4	sub-session ID
1	status

Table 5.32: `SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_NTF` from to FiRa 2.0

Size (octet)	Field
4	Session Handle
1	Number of controlee statuses (n)
$\sum_{i=1}^{n} len_{controlee_i}$	List of n controlees as described in Table 5.33

Table 5.33: `SESSION_UPDATE_CONTROLLER_MULTICAST_LIST_NTF` status from FiRa 2.0

Size (octets)	Field
2	controlee short address
1	status

5.5 *[Not supported in QM33 SDK]* Managing FiRa Hybrid scheduling session configuration

There are two messages to control FiRa Hybrid sessions, `SESSION_SET_HUS_CONTROLEE_CONFIG_CMD` and its corresponding `SESSION_SET_HUS_CONTROLEE_CONFIG_RSP` for controlees, `SESSION_SET_HUS_CONTROLLER_CONFIG_CMD` and its corresponding `SESSION_SET_HUS_CONTROLLER_CONFIG_RSP` for controler. They are illustrated in Table 5.34, Table 5.36, Table 5.37 and Table 5.39

Table 5.34: `SESSION_SET_HUS_CONTROLLER_CONFIG_CMD`

Size (octet)	Field
4	Session handle
1	Number of phase (n)
8	Update time
$\sum_{i=1}^{n} len_{phase_i}$	List of n phase as described in Table 5.35

Table 5.35: `SESSION_SET_HUS_CONTROLLER_CONFIG_CMD` phase

Size (octet)	Field
4	Session ID
2	Start slot index
2	End slot index
1	Control
2|8	MAC address

Table 5.36: `SESSION_SET_HUS_CONTROLLER_CONFIG_CMD`

Size (octet)	Field
1	Status

Table 5.37: `SESSION_SET_HUS_CONTROLEE_CONFIG_CMD`

Size (octet)	Field
4	Session handle
1	Remaining multicast list size
1	Number of phases (n)
$\sum_{i=1}^{n} len_{phase_i}$	List of n phases as described in Table 5.38

Table 5.38: `SESSION_SET_HUS_CONTROLEE_CONFIG_CMD` phase

Size (octet)	Field
4	Session handle

Table 5.39: `SESSION_SET_HUS_CONTROLEE_CONFIG_RSP`

Size (octet)	Field
1	Status

5.6 *[Not supported in QM33 SDK]* Managing FiRa OWR DLTDOA ranging session configuration

There are two types of OWR DLTDOA ranging sessions: anchors and tags sessions. The configuration of those sessions is done using two different pair of messages: `SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_CMD`/`SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_RSP` and `SESSION_UPDATE_DT_TAG_RANGING_ROUNDS_CMD`/`SESSION_UPDATE_DT_TAG_RANGING_ROUNDS_RSP`.

Those messages are illustrated respectively in Table 5.40, Table 5.44, Table 5.46 and Table 5.48.

Table 5.40: `SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_CMD`

Size (octet)	Field
4	Session Handle
1	Number of active ranging rounds (n)
$\sum_{i=1}^{n} len_{round_i}$	List of n rounds as described in Table 5.41

Table 5.41: `SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_CMD` round

Size (octets)	Field
1	Round index
1	Ranging role
0|1	Number of responders (n)
$\sum_{i=1}^{n} len_{responder_i}$	List of n responders as described in Table 5.42

Table 5.42: `SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_CMD` responder

Size (octets)	Field
2|8	Responder address
$\sum_{i=1}^{n} len_{slot_i}$	List of n responder slots as described in Table 5.43

Table 5.43: `SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_CMD` responder slot

Size (octets)	Field
1	Slot index

Table 5.44: `SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_RSP`

Size (octet)	Field
1	Status
1	Number of non-activated ranging rounds (n)
$\sum_{i=1}^{n} len_{round_i}$	List of n non-activated rounds as described in Table 5.45

Table 5.45: SESSION_UPDATE_DT_ANCHOR_RANGING_ROUNDS_RSP non activated ranging round

Size (octets)	Field
1	Round index

Table 5.46: SESSION_UPDATE_DT_TAG_RANGING_ROUNDS_CMD

Size (octet)	Field
4	Session Handle
1	Number of active ranging rounds (n)
$\sum_{i=1}^{n} len_{round_i}$	List of n active rounds as described in Table 5.47

Table 5.47: SESSION_UPDATE_DT_TAG_RANGING_ROUNDS_CMD active ranging round

Size (octets)	Field
1	Round index

Table 5.48: SESSION_UPDATE_DT_TAG_RANGING_ROUNDS_RSP

Size (octet)	Field
1	Status
1	Number of non-activated ranging rounds (n)
$\sum_{i=1}^{n} len_{round_i}$	List of n non-activated rounds as described in Table 5.49

Table 5.49: SESSION_UPDATE_DT_TAG_RANGING_ROUNDS_RSP non activated ranging round

Size (octets)	Field
1	Round index

5.7 Receiving FiRa ranging session information

For all FiRa modes, the Qorvo *UWB* Subsystem uses the same message named `SESSION_INFO_NTF` (illustrated in Table 5.50) to send ranging result to the host.

Table 5.50: `SESSION_INFO_NTF`

Size (octet)	Field
4	Session Sequence counter
4	Session Handle
1	*RFU*
4	Current ranging interval
1	Ranging measurement type
1	*RFU*
1	MAC addressing mode indicator
8	*RFU*
1	Number of ranging measurements (n)
$\sum_{i=1}^{n} len_{meas_i}$	List of n ranging measurements as described in Table 5.51, Table 5.52 and Table 5.53

Depending on the session type, the content of a measurement may change. The 3 types of measurements are illustrated in Table 5.51, Table 5.52 and Table 5.53.

Table 5.51: `SESSION_INFO_NTF` TWR measurement

Size (octets)	Field
2|8	Responder address
1	Status
1	NLoS
2	Distance
2	*AoA* Azimuth
1	*AoA* Azimuth FoM
2	*AoA* Elevation
1	*AoA* Elevation FoM
2	Destination AoA Azimuth
1	Destination AoA Azimuth FoM
2	Destination AoA Elevation
1	Destination AoA Elevation FoM
1	Slot index
1	*RSSI*
11|5	*RFU*

Table 5.52: SESSION_INFO_NTF OWR AoA measurement

Size (octets)	Field
2|8	Responder address
1	Status
1	NLoS
1	Frame Sequence number
2	Block index
2	*AoA* Azimuth
1	*AoA* Azimuth FoM
2	*AoA* Elevation
1	*AoA* Elevation FoM

Table 5.53: SESSION_INFO_NTF OWR DLTDOA measurement

Size (octets)	Field
2|8	Responder address
1	Status
1	Message type
2	Message Control
2	Block index
1	Round Index
1	NLoS
2	*AoA* Azimuth
1	*AoA* Azimuth FoM
2	*AoA* Elevation
1	*AoA* Elevation FoMe
1	*RSSI*
5|8	Tx Timestamp
5|8	Rx Timestamp
2	Anchor CFO
2	CFO
4	Initiator Reply Time
4	Responder Reply Time
2	Initiator Responder *ToF*
0|10|12	Anchor location
$\sum_{i=1}^{n} len_{round_i}$	List of n active round indexes as described in Table 5.54

Table 5.54: SESSION_INFO_NTF OWR DLTDOA measurement active round

Size (octets)	Field
1	Round index

5.8 *[Not supported in QM33 SDK]* Sending data over a FiRa session

On some session types, it is possible to send and receive data. The two messages DATA_MESSAGE_SND and DATA_MESSAGE_RCV are used to send and receive data respectively and are illustrated in Table 5.55 and Table 5.56

Note: DATA_MESSAGE_SND and DATA_MESSAGE_RCV are not control messages but data messages. They do not have a GID/OID but is present in this chapter for sake of clarity.

Table 5.55: DATA_MESSAGE_SND

Size (octet)	Field
4	Session Handle
8	Destination address
2	Data Sequence Number
2	Application data length (n)
n	Application data

Table 5.56: DATA_MESSAGE_RCV

Size (octet)	Field
4	Session Handle
8	Source address
2	Data Sequence Number
2	Application data length (n)
n	Application data

The data transmission itself between an host and a device is managed using a set of control messages. A client can request the size of the data it is possible to send using the SESSION_QUERY_DATA_SIZE_IN_RANGING_CMD and its corresponding message SESSION_QUERY_DATA_SIZE_IN_RANGING_RSP are illustrated in Table 5.57 and Table 5.58.

Table 5.57: SESSION_QUERY_DATA_SIZE_IN_RANGING_CMD

Size (octet)	Field
4	Session Handle

Table 5.58: SESSION_QUERY_DATA_SIZE_IN_RANGING_RSP

Size (octet)	Field
1	Session handle
1	Status
2	Maximum data size

The ongoing transmission control itself is done using the SESSION_DATA_TRANSFER_STATUS_NTF and SESSION_DATA_CREDIT_NTF messages illustrated in Table 5.60 and Table 5.59 respectively.

Table 5.59: SESSION_DATA_CREDIT_NTF

Size (octet)	Field
4	Session Handle
1	Credit available

Table 5.60: SESSION_DATA_TRANSFER_STATUS_NTF

Size (octet)	Field
4	Session Handle
2	Data Sequence Number
1	TX Count

6 UCI Qorvo extension 2 GID

6.1 Getting Debug Information

This notification has been designed to be able to get information about the device during specific test mode. It is always active when these tests are running. This notification is sent for every frame sent.

Table 6.1: QORVO_TEST_DEBUG_NTF

Size (octet)	Field
2	Azimuth PDoA (degrees) Q7
2	Elevation PDoA (degrees) Q7
1	Number of RSSIs
2*n	RSSI (dBm) Q8
2	Noise Value (dB)
2	Azimuth AoA (degrees) Q7
2	Elevation AoA (degrees) Q7
4	CFO Q26, to get PPM the value must be divided by 2^26 and then multiplied by 10^6

6.2 Getting FiRa Diagnostics

This section describes the different *UCI* messages generated by the FiRa diagnostics feature. For more information about this feature, see fira/fira_diagnostics.

The *OID* of the FiRa diagnostic message is described in Table 6.2. The feature is configurable on a per-session basis.

Table 6.2: OID of the message containing the FiRa diagnostics

OID	Control messages
QORVO_FIRA_RANGE_DIAGNOSTICS (b0000011)	QORVO_FIRA_RANGE_DIAGNOSTICS_NTF

The format of the *TLV*s contained in the QORVO_FIRA_RANGE_DIAGNOSTICS_NTF message is described in Table 6.3.

Table 6.3: Content of a TLV in QORVO_FIRA_RANGE_DIAGNOSTICS_NTF UCI message

Field name	Length (octets)	Description
Tag	1	The ID of the field reported in the *TLV*.
Length	2	The total length of the *TLV*.
Data	variable	The content of the *TLV*.

The QORVO_FIRA_RANGE_DIAGNOSTICS_NTF *UCI* message sent just after standard SESSION_INFO_NTF *UCI* message is illustrated in Table 6.4.

Its different fields are illustrated in Table 6.5, Table 6.6, Table 6.7, Table 6.9, Table 6.10, Table 6.11, Table 6.12, Table 6.13.

Note: Despite being coded as a *TLV*s the Frame Status (for RX & TX frames) cannot be deactivated like the others fields.

Table 6.4: Content of the QORVO_FIRA_RANGE_DIAGNOSTICS_NTF UCI message

Field name	Length (octets)	Description
Session ID	4	The id of the session to which the range diagnostics notification is related to
Sequence number	4	The current sequence number of the round (same as the *sequence number* field of SESSION_INFO_NTF).
Number of frame reports	1	The number of frames reports stored in the field *frame reports*. Shall be equal to the number of frames composing the round.
Frame reports	variable	The frame reports for each frame of the round. See Table 6.5 for details.

Table 6.5: Content of Frame reports in QORVO_FIRA_RANGE_DIAGNOSTICS_NTF UCI message

Field name	Length (octets)	Description
UWB Msg ID	1	The ID of the FiRa message (CM, RRRM, etc...) carried by the frame. 0xFF if undefined.
Action	1	Action performed on the frame: • 0x00: RX • 0x01: TX • 0x02-0xFF: *RFU*
Antenna Set	1	The antenna set used to exchange the frame.
Number of frame report fields	1	The number of fields contained in the frame report
Frame report fields	variable	The different fields of the report as *TLV*s. See different possible *TLV*s in Table 6.6.

Table 6.6: Content of `Frame reports > Frame report fields` in `QORVO_FIRA_RANGE_DIAGNOSTICS_NTF` UCI message

Field name	Tag	Length (octets)	Description
Deprecated	0x0	*NA*	Deprecated, should not be used anymore.
AoAs	0x1	variable	AoA measurements measured on the frame. Only for RX frames. See Table 6.9.
Deprecated	0x2	*NA*	Deprecated, should not be used anymore.
Frame status	0x3	2	Status attached to the frame. See Table 6.13.
CFO	0x4	4	Clock frequency offset measured between the transmitter and the receiver on the frame coded as an signed Q26 real.
Emitter Short address	0x5	2	The MAC short address of the frame emitter. May be not available.
Segment Metrics	0x6	variable	Segment Metrics measured on the frame. Only for RX frames. See Table 6.7.
CIRs	0x7	variable	CIRs measured on the frame. Only for RX frames. See Table 6.11.

Table 6.7: Content of `Frame reports > Frame report fields > Segment Metrics TLV` in `QORVO_FIRA_RANGE_DIAGNOSTICS_NTF` UCI message

Field name	Length (octets)	Description
Segment Metrics measurements	variable	List of Segment Metrics measurements computed on the frame. See Table 6.8.

Table 6.8: Content of `Frame reports > Frame report fields > Segment Metrics TLV > One Segment Metrics measurement` in `QORVO_FIRA_RANGE_DIAGNOSTICS_NTF` UCI message

Field name	Length (octets)	Description
Receiver/segment Id	1	The receiver and the frame segment the *CIR* has been computed formatted as: • b2-b0: Id of the segment with: – b000: IPATOV – b001: STS0 – b010: STS1 – b011: STS2 – b100: STS3 – b101-b111: RFU • b3: Master/slave indicator: – 0: the receiver is slave. – 1: the receiver is master. • b7-b4: Id of the receiver from 0x0 to 0xF
RF Noise Floor	2	The RF noise floor value in dBm measured during reception of the whole segment. Value is coded as a signed integer.
Global segment RSL	2	The absolute value in dBm of the *RSL* measured on the whole segment. Value is coded as an unsigned Q8 real.
First path index	2	The absolute index of the sample considered as the first path of the segment. Value is coded as an unsigned integer.
First path RSL	2	The absolute value in dBm of the *RSL* of the sample considered as the first path of the segment. Value is coded as an unsigned Q8 real.
First path ns	2	The position in nanoseconds of the First Path from the start of the *CIR* for the segment. Value is coded as an unsigned Q6 real.
Peak path index	2	The absolute index of the sample considered as the peak path of the segment. Value is coded as an unsigned integer.
Peak path RSL	2	The absolute value in dBm of the *RSL* of the sample considered as the peak path of the segment. Value is coded as an unsigned Q8 real.
Peak path ns	2	The position in nanoseconds of the Peak path from the start of the *CIR* for the segment. Value is coded as an unsigned Q6 real.

Table 6.9: Content of `Frame reports > Frame report fields > AoAs TLV` in `QORVO_FIRA_RANGE_DIAGNOSTICS_NTF` UCI message

Field name	Length (octets)	Description
AoA measurements	variable	List of *AoA* measurements computed on the frame. See Table 6.10.

Table 6.10: Content of `Frame reports > Frame report fields > AoAs TLV > One AoA measurement` in `QORVO_FIRA_RANGE_DIAGNOSTICS_NTF` UCI message

Field name	Length (octets)	Description
TDoA	2	The *TDoA* in RCTU use to compute the *PDoA* coded as a signed integer.
PDoA	2	The *PDoA* in radians use to compute the *AoA* coded as a signed Q11 real.
AoA	2	The *AoA* in radians coded as a signed Q11 real.
FOM	1	The *FoM* attached to *AoA* coded as a unsigned integer.
Type	1	The type of the *AoA* coded as a unsigned integer.

Table 6.11: Content of `Frame reports > Frame report fields > CIR TLV` in `QORVO_FIRA_RANGE_DIAGNOSTICS_NTF` UCI message

Field name	Length (octets)	Description
Computed CIRs	variable	List of *CIR*s measurements computed on the frame. See Table 6.12

Table 6.12: Content of `Frame reports > Frame report fields > CIR TLV > One CIR` in `QORVO_FIRA_RANGE_DIAGNOSTICS_NTF` UCI message

Field name	Length (octets)	Description
Receiver/segment	1	The receiver and the frame segment
Id		the CIR has been computed formated as: • b2-b0: Id of the segment with: – b000: IPATOV – b001: STS0 – b010: STS1 – b011: STS2 – b100: STS3 – b101-b111: RFU • b3: Master/slave indicator: – 0: the receiver is slave. – 1: the receiver is master. • b7-b4: Id of the receiver from 0x0 to 0xF
First path sample offset	1	The offset within the sample window where the first path sample can be found coded as an unsigned integer.
Samples number	1	The offset within the sample window where the first path sample can be found coded as an unsigned integer.
Sample size	1	The offset within the sample window where the first path sample can be found coded as an unsigned integer.
Sample window	variable	The samples of the window. Each sample is coded using the sample size.

Table 6.13: Content of `Frame reports > Frame report fields > Frame status TLV` in `QORVO_FIRA_RANGE_DIAGNOSTICS_NTF` UCI message

Field name	Length (octets)	Description
Frame status	2	Bitfield containing different status about the frame: • b0: Frame processed correctly • b1: WiFi activation state during the frame • b2: Max grant duration exceeded • b15-b3: *RFU*

7 Messages Parameters

This section contains the UCI message parameters used in different messages types.

7.1 Device Configuration Parameters

The parameters defined into Table 7.1 are used with *CORE_SET_CONFIG_CMD* and *CORE_GET_CONFIG_CMD*.

Table 7.1: Device configuration parameters

Name	Type	Length	Default	Description
FiRa defined:				
DEVICE_STATE	0x00	1	*NA*	State reported by CORE_DEVICE_STATUS, read only. see FiRa UCI specification
LOW_POWER_MODE	0x01	1	1	Enable (1) or disable (0) low power mode, see FiRa UCI specification
FiRa *RFU*	0x02 - 0x9F			Reserved for future usage
Reserved:				
Vendor *RFU*	0xA0 - 0xDF			Reserved for future usage
Extension *RFU*	0xE0 - 0xE2			Reserved for future usage
Vendor *RFU*	0xE3 - 0xFF			Reserved for future usage

Device configuration can be overridden by a protocol during its operation.

7.2 Application Configuration Parameters

The parameters defined into Table 7.2 are used with *SESSION_SET_APP_CONFIG* and *SESSION_GET_APP_CONFIG*.

The “Protocol” Column indicate for which session type the parameter is applicable:

- F: FiRa sessions
- P: PCTT sessions

Table 7.2: Application configuration parameters

Name	Type	Length	Protocol	Description
FiRa defined:				
DEVICE_TYPE	0x00	1	F	• 0x00: Controlee • 0x01: Controller
RANGING_ROUND_USAGE	0x01	1	F	• 0x00: OWR UL-TDoA *[Not supported in QM33 SDK]* • 0x01: SS-TWR Deferred • 0x02: DS-TWR Deferred • 0x03: SS-TWR Non-deferred • 0x04: DS-TWR Non-deferred • 0x05: OWR DL-TDoA *[Not supported in QM33 SDK]* • 0x06: OWR AoA *[Not supported in QM33 SDK]* • 0x07: eSS-TWR Non-deferred for contention-based *[Not supported in QM33 SDK]* • 0x08: aDS_TWR for contention-based *[Not supported in QM33 SDK]*
STS_CONFIG	0x02	1	F/P	• 0x00: Static STS • 0x01: Dynamic STS *[Not supported in QM33 SDK]* • 0x02: Dynamic STS for Responder specific sub-session key *[Not supported in QM33 SDK]* • 0x03: Provisioned STS • 0x04: Provisioned STS for Responder specific sub-session key
MULTI_NODE_MODE	0x03	1	F	• 0x00: One2One • 0x01: One2Many
CHANNEL_NUMBER	0x04	1	F/P	**RF channel to be used. Supported values are:** • 5 for 6489.6 MHz • 9 for 7987.2 MHz (Default)

continues on next page

Table 7.2 – continued from previous page

Name	Type	Length	Protocol	Description
NUMBER_OF_CONTROLEES	0x05	1	F	Number of controlees in the session, value between 1 and 8. (Default is 1)
DEVICE_MAC_ADDRESS	0x06	2/8	F/P	Either a short address (2 bytes) or a long (8 bytes)
DST_MAC_ADDRESS	0x07	2*N	F/P	N should match the NUMBER_OF_CONTROLEES
SLOT_DURATION	0x08	2	F/P	Duration of a ranging slot in the unit of RTSU (Default 2400 = 2 ms)
RANGING_DURATION	0x09	4	F	Duration of a ranging block in the unit of 1200 RSTU = 1 ms. (Default 200)
STS_INDEX	0x0A	4	P	
MAC_FCS_TYPE	0x0B	1	F/P	• 0x00: CRC 16 (Default) • 0x01: CRC 32 (Not yet supported)
RANGING_ROUND_CONTROL	0x0C	1	F	
AOA_RESULT_REQ	0x0D	1	F	• 0x00: All AoA results disabled • 0x01: All AoA result enabled (Default) • 0x02: Only AoA Azimuth *[Not supported in QM33 SDK]* • 0x03: Only AoA Elevation *[Not supported in QM33 SDK]*
[Not supported in QM33 SDK] SESSION_INFO_NTF_CONFIG	0x0E	1	F	• 0x00: ranging data notification disabled • 0x01: ranging data notification enabled (Default) • 0x02: ranging data notification while inside proximity range • 0x03: ranging data notification while inside AoA bounds • 0x04: ranging data notification while inside proximity range and AoA bounds • 0x05: ranging data notification while entering or leaving proximity range • 0x06: ranging data notification while entering or leaving AoA bounds • 0x07: ranging data notification while entering or leaving proximity range and AoA bounds
[Not supported in QM33 SDK] NEAR_PROXIMITY_CONFIG	0x0F	2	F	In cm. (Default 0)
[Not supported in QM33 SDK] FAR_PROXIMITY_CONFIG	0x10	2	F	In cm. (Default 20000)

continues on next page

Table 7.2 – continued from previous page

Name	Type	Length	Protocol	Description
DEVICE_ROLE	0x11	1	F/P	• 0x00: Responder • 0x01: Initiator • 0x02: UT-Synchronization Anchor *[Not supported in QM33 SDK]* • 0x03: UT-Anchor *[Not supported in QM33 SDK]* • 0x04: UT-Tag *[Not supported in QM33 SDK]* • 0x05: Advertiser *[Not supported in QM33 SDK]* • 0x06: Observer *[Not supported in QM33 SDK]* • 0x07: DT-Anchor *[Not supported in QM33 SDK]* • 0x08: DT-Tag *[Not supported in QM33 SDK]*
RFRAME_CONFIG	0x12	1	F/P	• 0x00: SP0 (Applicable only for PCTT) • 0x01: SP1 • 0x02: RFU • 0x03: SP3 (Default)
RSSI_REPORTING	0x13	1	F	• 0x00: Disabled (Default) • 0x01: Enabled
PREAMBLE_CODE_INDEX	0x14	1	F/P	9 - 12 for BPRF 25 - 32 for HPRF *[Not supported in QM33 SDK]* (Default 10)
SFD_ID	0x15	1	F/P	• 0 • 1 *[Not supported in QM33 SDK]* • 2 • 3 *[Not supported in QM33 SDK]* • 4 *[Not supported in QM33 SDK]*
PSDU_DATA_RATE	0x16	1	F/P	• 0x00: 6.81 Mbps (Default) • 0x01: 7.80 Mbps *[Not supported in QM33 SDK]* • 0x02: 27.2 Mbps *[Not supported in QM33 SDK]* • 0x03: 31.2 Mbps *[Not supported in QM33 SDK]* • 0x04: 850 kbps (Not supported)

continues on next page

Table 7.2 – continued from previous page

Name	Type	Length	Protocol	Description
PREAMBLE_DURATION	0x17	1	F/P	• 0x00: 32 Symbols *[Not supported in QM33 SDK]* • 0x01: 64 Symbols (Default)
LINK_LAYER_MODE	0x18	1	F	• 0x00: Bypass Mode (Default) • 0x01: Connectionless Mode *[Not supported in QM33 SDK]*
[Not supported in QM33 SDK] DATA_REPETITION_COUNT	0x19	1	F	**Only applicable to OWR session:** • 0x00: No repetition (Default) • 0xFF: Repeat infinitely
RANGING_TIME_STRUCT	0x1A	1	F	• 0x01: Block Based Scheduling (Default)
SLOTS_PER_RR	0x1B	1	F	(Default 25)
RFU	0x1C	1		
[Not supported in QM33 SDK] AOA_BOUND_CONFIG	0x1D	8	F	
RFU	0x1E	1		
PRF_MODE	0x1F	1	F	• 0x00: 62.4 MHz, BPRF mode • 0x01: 124.8 MHz, HPRF mode *[Not supported in QM33 SDK]* • 0x02: 249.6 MHz. HPRF mode *[Not supported in QM33 SDK]*
[Not supported in QM33 SDK] CAP_SIZE_RANGE	0x20	2	F	
TX_JITTER_WINDOW_SIZE	0x21	1	F	Not supported
SCHEDULE_MODE	0x22	1	F	• 0x00: Contention-based ranging *[Not supported in QM33 SDK]* • 0x01: Time schedule ranging

continues on next page

Table 7.2 – continued from previous page

Name	Type	Length	Protocol	Description
KEY_ROTATION	0x23	1	F	• 0x00: Disabled (Default) • 0x01: Enabled
KEY_ROTATION_RATE	0x24	1	F	
SESSION_PRIORITY	0x25	1	F	Value between 1 and 100. (Default 50)
MAC_ADDRESS_MODE	0x26	1	F	• 0x00: MAC address is 2 bytes (Default) • 0x01: MAC address is 8 bytes but 2 bytes in MAC header (Not supported) • 0x02: MAC address is 8 bytes (Not supported)
VENDOR_ID	0x27	2	F	
STATIC_STS_IV	0x28	6	F	
NUMBER_OF_STS_SEGMENTS	0x29	1	F	• 0x00: No STS Segments • 0x01: 1 STS Segment • 0x02: 2 STS Segments *[Not supported in QM33 SDK]* • 0x03: 3 STS Segments *[Not supported in QM33 SDK]* • 0x04: 4 STS Segments *[Not supported in QM33 SDK]*
MAX_RR_RETRY	0x2A	2	F	
[Not supported in QM33 SDK] UWB_INITIATION_TIME	0x2B	8	F	
HOPPING_MODE	0x2C	1	F	
BLOCK_STRIDE_LENGTH	0x2D	1	F	
RESULT_REPORT_CONFIG	0x2E	1	F	
IN_BAND_TERMINATION_ATTEMPT_COUNT	0x2F	1	F	
SUB_SESSION_ID	0x30	4	F	
BPRF_PHR_DATA_RATE	0x31	1	F/P	• 0x00: 850 kbps (default) • 0x01: 6.81 Mbps
MAX_NUMBER_OF_MEASUREMENTS	0x32	2	F	
[Not supported in QM33 SDK] UL_TDOA_TX_INTERVAL	0x33	4	F	Not part of FiRa 2.0
[Not supported in QM33 SDK] UL_TDOA_RANDOM_WINDOW	0x34	4	F	Not port of FiRa 2.0

continues on next page

Table 7.2 – continued from previous page

Name	Type	Length	Protocol	Description
STS_LENGTH	0x35	1	F/P	• 0x00: 32 symbols • 0x01: 64 symbols (default) • 0x02: 128 symbols
Assigned	0x36	1	NA	
[Not supported in QM33 SDK] UL_TDOA_NTF_REPORT_CONFIG	0x37	3	F	Not part of FiRa 2.0
[Not supported in QM33 SDK] UL_TDOA_DEVICE_ID	0x38	1/3/5/9	F	Not part of FiRa 2.0
[Not supported in QM33 SDK] UL_TDOA_TX_TIMESTAMP	0x39	1	F	Not part of FiRa 2.0
[Not supported in QM33 SDK] MIN_FRAMES_PER_RR	0x3A	1	F	
[Not supported in QM33 SDK] MTU_SIZE	0x3B	2	F	
[Not supported in QM33 SDK] INTER_FRAME_INTERVAL	0x3C	1	F	
[Not supported in QM33 SDK] DL_TDOA_RANGING_METHOD	0x3D	1	F	
[Not supported in QM33 SDK] DL_TDOA_TX_TIMESTAMP_CONF	0x3E	1	F	
[Not supported in QM33 SDK] DL_TDOA_HOP_COUNT	0x3F	1	F	
[Not supported in QM33 SDK] DL_TDOA_ANCHOR_CFO	0x40	1	F	
[Not supported in QM33 SDK] DL_TDOA_ANCHOR_LOCATION	0x41	1/11/13	F	
[Not supported in QM33 SDK] DL_TDOA_TX_ACTIVE_RANGING_ROUNDS	0x42	1	F	
[Not supported in QM33 SDK] DL_TDOA_BLOCK_SKIPPING	0x43	1	F	
[Not supported in QM33 SDK] DL_TDOA_TIME_REFERENCE_ANCHOR	0x44	1	F	
SESSION_KEY	0x45	16/32	F	
SUB_SESSION_KEY	0x46	16/32	F	
[Not supported in QM33 SDK] SESSION_DATA_TRANSFER_STATUS_NTF_CONI	0x47	1	F	
SESSION_TIME_BASE	0x48	9	F	Not supported

continues on next page

Table 7.2 – continued from previous page

Name	Type	Length	Protocol	Description
[Not supported in QM33 SDK] DL_TDOA_RESPONDER_TOF	0x49	1	F	
SECURE_RANGING_NEFA_LEVEL	0x4A	1	F	Not supported
SECURE_RANGING_CSW_LENGTH	0x4B	1	F	Not supported
APPLICATION_DATA_ENDPOINT	0x4C	1	F	• 0x00: Host (Default) • 0x01: Secure Component *[Not supported in QM33 SDK]*
[Not supported in QM33 SDK] OWR_AOA_MEASUREMENT_NTF_PERIOD	0x4D	1	F	
RFU	0x4E - 0xDF			Reserved for future usage
Reserved for extension of IDs:				
RFU	0xE0 - 0xE2			
Vendor defined:				
Vendor *RFU*	0xE3 - 0xE5			Reserved for future usage
RX_ANTENNA_SELECTION	0xE6	1	F/P	
TX_ANTENNA_SELECTION	0xE7	1	F/P	
ENABLE_DIAGNOSTICS	0xE8	1	F/P	**Activate the diagnostic notification.** • 0x00: Disable (Default) • 0x01: Enable • 0x02..0xFF: *RFU* **Only for FiRa and PCTT sessions, it enables:** • *QORVO_TEST_DEBUG_NTF* for PCTT • *QORVO_FIRA_RANGE_DIAGNOSTICS_NTF* for FiRa

continues on next page

Table 7.2 – continued from previous page

Name	Type	Length	Protocol	Description
DIAGNOSTICS_FRAME_REPORTS_FIELDS	0xE9	1	F	**Bitfield to select reported diagnostics:** • b0: Deprecated • b1: Activate AoAs field • b2: Deprecated • b3: Activate CFO field • b4: Activate Emitter Short addr field • b5: Activate Segment Metrics field • b6: Activate CIRs field • b7: RFU If the ENABLE_DIAGNOSTICS parameters is not true this parameter does not activate the diagnostics itself. Only supported by FiRa sessions.
Vendor RFU	0xEA - 0xFF			Reserved for future usage

References

[FiR22] FiRa. Uci test specification. Technical Report, FiRa, 2022.

[FiR23a] FiRa. Fira consortium uwb command interface generic technical specification. Technical Report, FiRa, 2023.

[FiR23b] FiRa. Fira consortium uwb mac technical requirements. Technical Report, FiRa, 2023.

[IEE20a] IEEE. Standard for low-rate wireless networks. Technical Report, IEEE, 2020.

[IEE20b] IEEE. Standard for low-rate wireless networks - amendment 1: enhanced ultra wideband (uwb) physical layers (phys) and associated ranging techniques. Technical Report, IEEE, 2020.

[NIS01a] NIST. Fips 197: advanced encryption standard (aes). Technical Report, NIST, 2001.

[NIS01b] NIST. Nist sp 800-38a: recommendation for block cipher modes of operation: methods and techniques. Technical Report, NIST, 2001.

[NIS04] NIST. Nist sp 800-38c: recommendation for block cipher modes of operation: the ccm mode for authentication and confidentiality. Technical Report, NIST, 2004.

[NIS05] NIST. Nist sp 800-38b: recommendation for block cipher modes of operation: the cmac mode for authentication. Technical Report, NIST, 2005.

[NIS09] NIST. Nist sp 800-108: recommendation for key derivation using pseudorandom functions. Technical Report, NIST, 2009.

Index