SPECIFICATION

UWB-Stack L1 configuration

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 L1 Config

2.1 Overview

L1 Config allows to configure the **Layer 1** as defined by the OSI model, also know as the **Physical Layer**.

Following is a **non-exhaustive** list of the parameters L1 Config allows to configure:

- The crystal oscillator trim;
- The antenna configuration;
- The antenna delays required for an accurate ranging measure;
- The lookup table matching the mounted antennas and allowing to translate measured PDoA (Phase Difference to AoA (Angle Of Arrival);
- The TX power settings depending on configured Reference Frames;
- The configuration of some specific features.

Note: L1 config is sometimes wrongly referred as “calibration” module. That imprecise term is due to the fact that L1 config allows to store some parameters which may be specific per device, thus often called calibration parameters. However, the main part of the parameters stored in L1 config are actually **configuration parameters**.

2.2 Functional

2.2.1 List of parameters

That section lists all the L1 Config parameters.

Note: Some of the parameters are itemized, which is represented by <*> field, where:

- <*K*>: Is the Antenna Path index, from 0 to `CONFIG_L1_CONFIG_ANT_NUM` - 1.
- <*X*>: Is the Antenna Set index, from 0 to `CONFIG_L1_CONFIG_ANT_SET_NUM` - 1.
- <*P*>: Is the Antenna Pair index, from 0 to `CONFIG_L1_CONFIG_ANT_PAIR_NUM` - 1.
- <*C*>: Is the channel number, 5 or 9.
- <*R*>: Is the Reference Frame index, from 0 to 7.
- <*N*>: Is the measurement type, *range*, *azimuth*, *elevation*.
- <*I*>: Is the pdoa lut id, from 0 to `CONFIG_L1_CONFIG_PDOA_LUT_NUM`.

- *restricted_channels*:
  - Bit field allowing to restrict some channels. Used for regulatory concerns.
  - Type: `uint16_t`.
  - Default: `0`, no channel restricted.
- *alternate_pulse_shape*:
  - Boolean value that define if an alternative pulse shape is used:
    - `0`: standard pulse shape.
    - `1`: Japan-specific pulse shape.
  - Type: `uint8_t`.
  - Default: `0`.
- *ref_frame<R>.phy_cfg*:
  - PHY configuration which, combined to Payload size, defines a *Reference frames*.
  - Reference frame are then used to itemize some other parameters, used for TX power index selection.
  - Type: `uint8_t[3]`, composed by:
    - bit[0]: `prf`: PRF mode - 0 for BPRF or 1 for HPRF.
    - bits[1:3]: `sfd_type`: SFD types, as defined by *dwt_sfd_type_e*:
      - `0`: IEEE_4A
      - `1`: IEEE_4Z_4
      - `2`: IEEE_4Z_8
      - `3`: IEEE_4Z_16
      - `4`: IEEE_4Z_32
      - `6`: DW_8
      - `7`: DW_16
    - bit[4:7]: `psr`: Number of preamble symbol repetitions in the SYNC preamble:

      - `0`: PSR_16
      - `1`: PSR_24
      - `2`: PSR_32
      - `3`: PSR_48
      - `4`: PSR_64
      - `5`: PSR_96
      - `6`: PSR_128
      - `7`: PSR_256
      - `8`: PSR_512
      - `9`: PSR_1024
      - `10`: PSR_2048
      - `11`: PSR_4096
    - bits[8:11]: `payload_rate`: Payload data rate, as defined by values:
      - `0x0`: 850K
      - `0x1`: 6M8
      - `0x2`: NODATA
      - `0x4`: 6M8_128
      - `0x5`: 27M_256
      - `0xC`: 6M8_128_K7
      - `0xD`: 27M_256_K7
      - `0xE`: 54M_256
      - `0xF`: 108M_256
    - bit[12]: `phr_rate`: BPRF PHR rate:
      - `0` for Standard data rate.
      - `1` for High data rate (6M81).
    - bits[13:15]: `sts_seg_num`: Number of STS segments, from 0 to 4.
    - bits[16:23]: `sts_seg_len`: Length of STS segments:
      - `0`: STS_LEN_0 (No STS segment)
      - `1`: STS_LEN_16
      - `3`: STS_LEN_32
      - `7`: STS_LEN_64
      - `15`: STS_LEN_128
      - `31`: STS_LEN_256
      - `63`: STS_LEN_512
      - `127`: STS_LEN_1024
      - `255`: STS_LEN_2048
  - Default:

  * Only two first Reference Frames have default values.
    - Reference Frame #0, corresponding to BPRF SET#3 according to FiRa UWB PHY Technical specification. Its value is `{ 0x44, 0x21, 0x07 }`, corresponding to:
      - `prf`: BPRF
      - `sfd_type`: IEEE_4Z_8
      - `psr`: PSR_64
      - `payload_rate`: 6M8
      - `phr_rate`: Standard rate (0.85)
      - `sts_seg_num`: 1 segment
      - `sts_seg_len`: STS_LEN_64
    - Reference Frame #1, corresponding to HPRF SET#16 according to FiRa UWB PHY Technical specification. Its value is `{ 0x25, 0x25, 0x03 }`, corresponding to:
      - `prf`: HPRF
      - `sfd_type`: IEEE_4Z_8
      - `psr`: PSR_32
      - `payload_rate`: 27M2
      - `phr_rate`: 0 (N/A)
      - `sts_seg_num`: 1 segment
      - `sts_seg_len`: STS_LEN_32
    - **Note**: those two first reference frames are mandatory. They can be overwritten. However, some conditions apply:
      - Reference Frame #0 must always be BPRF, i.e bit[0] `prf` must be 0.
      - Reference Frame #1 must always be HPRF, i.e bit[0] `prf` must be 1.
  * Other reference frames are undefined by default.

- *ref_frame<R>.payload_size*:
  - Payload data size which, combined to PHY configuration, defines a *Reference frames*.
  - Type: `uint16_t`.
  - Default:
    * Two first Reference Frames are configured by default.
      - Reference Frame #0: `127`
      - Reference Frame #1: `3176`
    * Other reference frames are undefined by default.
- *rx_diag_config.cir_n_taps*:
  - Number of taps in one CIR, i.e. size of the CIR window, in CIR samples. This is used in UWB diagnostic.
  - Type: `uint16_t`.
  - Default: `16`.
- *rx_diag_config.cir_fp_tap_offset*:

  - CIR First Path tap offset, i.e. offset relative to first path tap at which the CIR window starts, in CIR samples. This is used in UWB diagnostic.
  - Type: `uint16_t`.
  - Default: `8`.
- *xtal_trim*:
  - Crystal trim value: in range 0x0 to 0x7F (127 steps, ~0.8ppm per step).
  - Type: `uint8_t`.
  - Default: Undefined by default. When no value is configured, value from OTP is used.
- *rf_noise_offset*:
  - Noise offset to apply when calculating the RX SNR. In dB unit.
  - Type: `int8_t`.
  - Default: -7 dB.
- *pdoa_lut<I>.data*:
  - PDOA Look up table.
  - Type: `int16_t[31][2]`.
  - Default: undefined by default.
- *debug.tx_power*:
  - Only used when *tx_power_control* is in *Debug mode*.
  - The configured TX power: The four bytes are in the same format than the corresponding register in the user manual.
  - Type: `uint32_t`.
  - Default: `0xFDFDFDFD`.
- *debug.pll_cfg*:
  - Only used when *tx_power_control* is in *Debug mode*.
  - PLL trim bias configuration.
  - Type: `uint32_t`.
  - Default: `0x00008104UL`.
- *debug.rx_segment*:
  - By default, the segment used for TOA is chosen by the TOA earliest first path algorithm.
  - In order to add more flexibility, especially needed for the RF engineers, the segment used can be overriden by current parameter, which in that case defines the segment to use to read the timestamp, STS or Ipatov quality as defined by *dwt_ip_sts_segment_e*:
    - `DWT_IP_M = 0x00`: Use Ipatov main receiver.
    - `DWT_STS0_M = 0x08`: Use STS0 main receiver.
    - `DWT_STS1_M = 0x10`: Use STS1 main receiver.
    - `DWT_STS2_M = 0x18`: Use STS2 main receiver.
    - `DWT_STS3_M = 0x20`: Use STS3 main receiver.
    - `DWT_IP_S = 0x28`: Use Ipatov slave receiver.

    * `DWT_STS0_S = 0x30`: Use STS0 slave receiver.
    * `DWT_STS1_S = 0x38`: Use STS1 slave receiver.
    * `DWT_STS2_S = 0x40`: Use STS2 slave receiver.
    * `DWT_STS3_S = 0x48`: Use STS3 slave receiver.
    * `RX_SEGMENT_DISABLED = 0xFF`: Disable the rx_segment forcing: segment used is the one defined by the TOA earliest first path algorithm.
  - If the received frame does not contain a STS segment while this parameter is set to a STS segment then the Ipatov segment of the same receiver is used. This is the case for instance if a SP0 frame is send.
  - Type: `uint8_t`
  - Default: `RX_SEGMENT_DISABLED`.
- *ant<K>.ch<C>.ref_frame<R>.tx_power_index*:
  - The TX power index (steps) where one unit means an attenuation of **0.25 dB** compared to maximum emitted power (calculated by the driver to reduce LO leakeage and maximize effective dynamics). Only used when `tx_power_control` is **not** in *Debug mode*. An index zero means no attenuation.
  - The power index (uint32_t) is divided into 4 different (uint8_t) power sections of the frame. Where:
    * `Byte 0`: power index on the STS section.
    * `Byte 1`: power index on the SHR section.
    * `Byte 2`: power index on the PHR section.
    * `Byte 3`: power index on the DATA section.
  - For each power sections of the frame, allowed values for the power index are between 0 and 0xFE.
  - See *TX power index* for more details about TX power control and implementation.
  - Type: `uint32_t`.
  - Default: `0`.
- *ant<K>.ch<C>.ref_frame<R>.max_gating_gain*:
  - The limit applied on the compensation/boosting gain calculated from Reference Frame properties, when *Adaptive TX Power* feature is enabled.
  - Usage: Prevent exceeding peak power (0 dBm in 50 MHz) for shortest SP0/SP1 frame types or to prevent from PSD failure issues due to ePA compression. Only used when `tx_power_control` corresponding frame bit is set to enable (bit 0 to 1).
  - The maximum Gating Gain value (uint32_t) is divided into 4 different (uint8_t) sections of the frame. Where:
    * `Byte 0`: power index on the STS section.
    * `Byte 1`: power index on the SHR section.
    * `Byte 2`: power index on the PHR section.
    * `Byte 3`: power index on the DATA section.
  - Type: `uint32_t`.
  - Default: `0xffffffff`.
- *ant<K>.ch<C>.ant_delay*:
  - The antenna delay of the specified antenna <K>, for channel <C>.
  - Type: `uint32_t`.

  - Default: `16405`.
- *ant<K>.ch<C>.pg_count*:
  - Pulse Generator count.
  - Type: `uint8_t`.
  - Default: `0`.
- *ant<K>.ch<C>.pg_delay*:
  - Pulse Generator delay.
  - Type: `uint8_t`.
  - Default: `0x34`.
- *ant<K>.ch<C>.rssi_offset_q3*:
  - Rssi Offset in Q5.3 format to be applied on computed Rssi (first path / peak path / segment).
  - Type: `int8_t`.
  - Default: `0`.
- *ant<K>.transceiver*:
  - Transceiver used on antenna Path K.
  - Expected value is among the following:
    * `TRANSCEIVER_TX = 0`
    * `TRANSCEIVER_RXA = 1`
    * `TRANSCEIVER_RXB = 2`
  - Type: `uint8_t`.
  - Default:
    * Only two first Antenna Paths have valid defined default values:
      · `ant0.transceiver = TRANSCEIVER_TX`
      · `ant1.transceiver = TRANSCEIVER_RXA`
      · `ant<K>.transceiver = 0`, with K from 2 to *CONFIG_L1_CONFIG_ANT_NUM* - 1.
- *ant<K>.port*:
  - Port used on Antenna Path K.
  - Expected value is among the following:
    * `ANT_PORT_1 = 1`
    * `ANT_PORT_2 = 2`
    * `ANT_PORT_3 = 3`
    * `ANT_PORT_4 = 4` (SiP only).
    * `ANT_PORT_NA = 0xF`: Invalid Antenna Port, unset.
  - Type: `uint8_t`.
  - Default:
    * Only two first Antenna Paths have valid defined default values:
      · `ant0.port = ANT_PORT_1`

        · `ant1.port = ANT_PORT_3`

    * Other Antenna Paths `port` are undefined by default:

        · `ant<K>.port = ANT_PORT_NA`, with K from 2 to *CONFIG_L1_CONFIG_ANT_NUM* - 1.

- *ant<K>.ext_sw_cfg*:
  - External SPDT Switch value used on Antenna Path K.
  - Type: `uint8_t`.
  - Default: `0`.
- *ant<K>.lna*:
  - LNA (Low Noise Amplifier) value used on Antenna Path K.
    * `0: Bypass.`
    * `1: Active.`
  - Type: `uint8_t`
  - Default: `Bypass`.
- *ant<K>.pa*:
  - PA (Power Amplifier) value used on Antenna Path K.
    * `0: Bypass.`
    * `1: Active.`
  - Type: `uint8_t`
  - Default: `Bypass`.
- *ant_pair<P>.axis*:
  - PDoA axis computed using Antenna Pair P:
    * `AOA_TYPE_X_AXIS = 0`
    * `AOA_TYPE_Y_AXIS = 1`
    * `AOA_TYPE_Z_AXIS = 2`
  - Type: `uint8_t`
  - Default: `AOA_TYPE_X_AXIS`.
- *ant_pair<P>.ant_paths*:
  - Index of the two Antenna Paths used for Antenna Pair P.
  - Type: `int8_t[2]`
  - Default: `{ -1, -1 }`.
- *ant_pair<P>.ch<C>.pdoa.offset*:
  - PDoA offset to use for the given Antenna Pair <P> and channel <C>.
  - Type: `int16_t`.
  - Default: `0`.
- *ant_pair<X>.ch<C>.pdoa.lut_id*:
  - PDoA Look up table index to use for the given Antenna Pair <P> and channel<C>. When special value `-1` is used, the L1 will use the default PDoA LUT table corresponding to the current channel. Those default LUTs are:

```
const pdoa_lut_t default_lut_ch5 = {
        { 0xe6de, 0xf36f },
        { 0xe88b, 0xf36f },
        { 0xea38, 0xf5b0 },
        { 0xebe5, 0xf747 },
        { 0xed92, 0xf869 },
        { 0xef3f, 0xf959 },
        { 0xf0ec, 0xfa2e },
        { 0xf299, 0xfaf1 },
        { 0xf445, 0xfba7 },
        { 0xf5f2, 0xfc53 },
        { 0xf79f, 0xfcf9 },
        { 0xf94c, 0xfd9a },
        { 0xfaf9, 0xfe36 },
        { 0xfca6, 0xfed0 },
        { 0xfe53, 0xff69 },
        { 0x0000, 0x0000 },
        { 0x01ad, 0x0097 },
        { 0x035a, 0x0130 },
        { 0x0507, 0x01ca },
        { 0x06b4, 0x0266 },
        { 0x0861, 0x0307 },
        { 0x0a0e, 0x03ad },
        { 0x0bbb, 0x0459 },
        { 0x0d67, 0x050f },
        { 0x0f14, 0x05d2 },
        { 0x10c1, 0x06a7 },
        { 0x126e, 0x0797 },
        { 0x141b, 0x08b9 },
        { 0x15c8, 0x0a50 },
        { 0x1775, 0x0c91 },
        { 0x1922, 0x0c91 }
};

const pdoa_lut_t default_lut_ch9 = {
        { 0xe6de, 0xf701 },
        { 0xe88b, 0xf7ff },
        { 0xea38, 0xf8d2 },
        { 0xebe5, 0xf98d },
        { 0xed92, 0xfa38 },
        { 0xef3f, 0xfad7 },
        { 0xf0ec, 0xfb6d },
        { 0xf299, 0xfbfc },
        { 0xf445, 0xfc86 },
        { 0xf5f2, 0xfd0c },
        { 0xf79f, 0xfd8f },
        { 0xf94c, 0xfe0f },
        { 0xfaf9, 0xfe8d },
        { 0xfca6, 0xff09 },
        { 0xfe53, 0xff85 },
        { 0x0000, 0x0000 },
        { 0x01ad, 0x007b },
        { 0x035a, 0x00f7 },
        { 0x0507, 0x0173 },
```

(continues on next page)

(continued from previous page)

```
        { 0x06b4, 0x01f1 },
        { 0x0861, 0x0271 },
        { 0x0a0e, 0x02f4 },
        { 0x0bbb, 0x037a },
        { 0x0d67, 0x0404 },
        { 0x0f14, 0x0493 },
        { 0x10c1, 0x0529 },
        { 0x126e, 0x05c8 },
        { 0x141b, 0x0673 },
        { 0x15c8, 0x072e },
        { 0x1775, 0x0801 },
        { 0x1922, 0x08ff }
};
```

  - Type: `int8_t`.
  - Default: `-1`.
- *ant_set<X>.tx_ant_path*:
  - Index of TX Antenna Path used in Antenna Set X.
  - Type: `int8_t`.
  - Default:
    * Only first Antenna Set has a valid default value:
      · `ant_set0.tx_ant_path = 0`
    * Other Antenna Sets `tx_ant_path` are undefined by default:
      · `ant_set<X>.tx_ant_path = -1`, with X from 1 to *CONFIG_L1_CONFIG_ANT_SET_NUM* - 1.
- *ant_set<X>.nb_rx_ants*:
  - Number of RX Antennas (Antenna Path or Antenna Pairs) contained in Antenna Set X.
  - Type: `uint8_t`.
  - Default: `0`.
- *ant_set<X>.rx_ants*:
  - List of RX Antennas indexes (Antenna Path or Antenna Pairs) used in Antenna Set X.
  - Type: `int8_t[3]`.
  - Default:
    * Only first Antenna Set has a valid default value:
      · `ant_set0.rx_ants = { 1, -1, -1 }`
    * Other Antenna Sets `rx_ants` are undefined by default:
      · `ant_set<X>.rx_ants = { -1, -1, -1 }`, with X from 1 to *CONFIG_L1_CONFIG_ANT_SET_NUM* - 1.
- *ant_set<X>.rx_ants_are_pairs*:
  - Indicate if the Antenna Set X contains Antenna Paths or Antenna Pairs.
  - Antenna Pairs are used for PDoA/AoA.
  - If *ant_set<X>.rx_ants_are_pairs* is `true`:

    * PDoA/AoA will be measured.
    * *ant_set<X>.rx_ants* contains index(es) of Antenna Pair(s).
  – If *ant_set<X>.rx_ants_are_pairs* is `false`:
    * PDoA/AoA will NOT be measured.
    * *ant_set<X>.rx_ants* contains index(es) of Antenna Path(s).
  – Type: `bool`.
  – Default: `false`.
- *ant_set<X>.tx_power_control*:
  – Bitfield that defines the TX power control mode:
    * bit[0]: `frame`: Frame *Adaptive TX Power* control: 0 to disable, 1 to enable.
    * bits[1:6]: Reserved for future usage.
    * bit[7]: `debug`: Debug mode: *tx_power*, *pa* and *pll_cfg* are taken from the debug specific parameters.
  – When bit `debug` is not set, Normal mode is used. Values of PA, TX power and PLL config are computed by driver depending on the TX power index.
    * Setting bit `frame` allows to enable Adaptive TX power for UWB frame.
    * When Adaptive TX Power is enabled, the TX power is compensated depending on the duration of the frame, compared to a given Reference frame.
  – When bit `debug` is set, Debug mode is used. Values of PA and PLL config must be provided by user using keys `debug.pa_enabled` and `debug.pll_cfg`.
  – Type: `uint8_t`.
  – Default: `0`.

3 Antenna management

3.1 Overview

UWB transceivers offer different antenna, *RFFE* and PDoA capabilities. Antenna management aims at configuring:

- The **complete antenna paths** (transceiver, RF port, LNA, PA, and switches) to use for TX and RX frames for different use-cases (Ranging, PDoA, Radar, etc.).
- The **PDoA configuration**, including the pair of antennas to use, the axis to measure, the PDoA to AoA Look-Up Tables and offset, etc.
- Some **antenna related calibration parameters** such as the antenna delays, the Pulse Generator count and delay.

That section aims at:

- Presenting the RF capabilities of the different UWB chip of Qorvo's portfolio.
- Describing the antenna management model of the solution.
- Presenting the PDoA and the different modes defined.
- Describing the parameters to calibrate and which relates to the antennas.
- Explaining some antenna related features, such as the Dual Rx Auto feature.

3.2 Functional

3.2.1 QM33 RF capabilities

QM33 is a transceiver-only chip containing only one transmitter (TX) and one Receiver (RX).

Both DW3000 and QM33 exist in two different versions:

- A **non-PDoA version**, which contains only one RF port.
- A **PDoA version**, which has two RF ports RF1 and RF2.

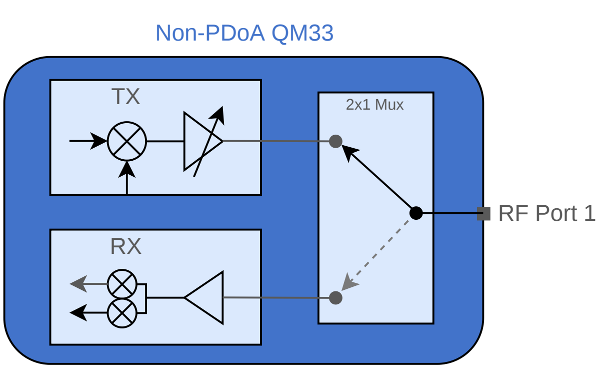

Fig. 3.1: QM33 SoC RFFE

On PDoA version, an internal switch allows to dynamically switch during the frame reception between the two RF ports in order to measure PDoA. Two modes are available:

- **PDoA mode 1**: the switch is done between Ipatov and STS0.
- **PDoA mode 3**: the switch is done in the middle of the STS0 reception. That mode requires STS length being a multiple of 128.

For both of those modes, a STS segment is required to be able to measure PDoA.

Important: PDoA mode 3 should be prefered to mode 1 because using STS instead of Ipatov is more accurate and secured. However, it requires a STS length value which is out of FiRa specification.

3.2.2 Antenna Management model

3.2.2.1 Concepts involved

Three concepts are defined in order to configure the antennas:

- *Antenna Path*
- *Antenna Pair*
- *Antenna Set*

3.2.2.1.1 Antenna Path

The **Antenna Path** defines the complete antenna path, allowing to configure all the internal and external switches involved. That complete path is defined by the following information:

- The transceiver (Receiver or Transmitter ID)
- The RF Port
- The state of the LNA, when applicable
- The state of the PA, when applicable
- The state of the external switch(es), when applicable.

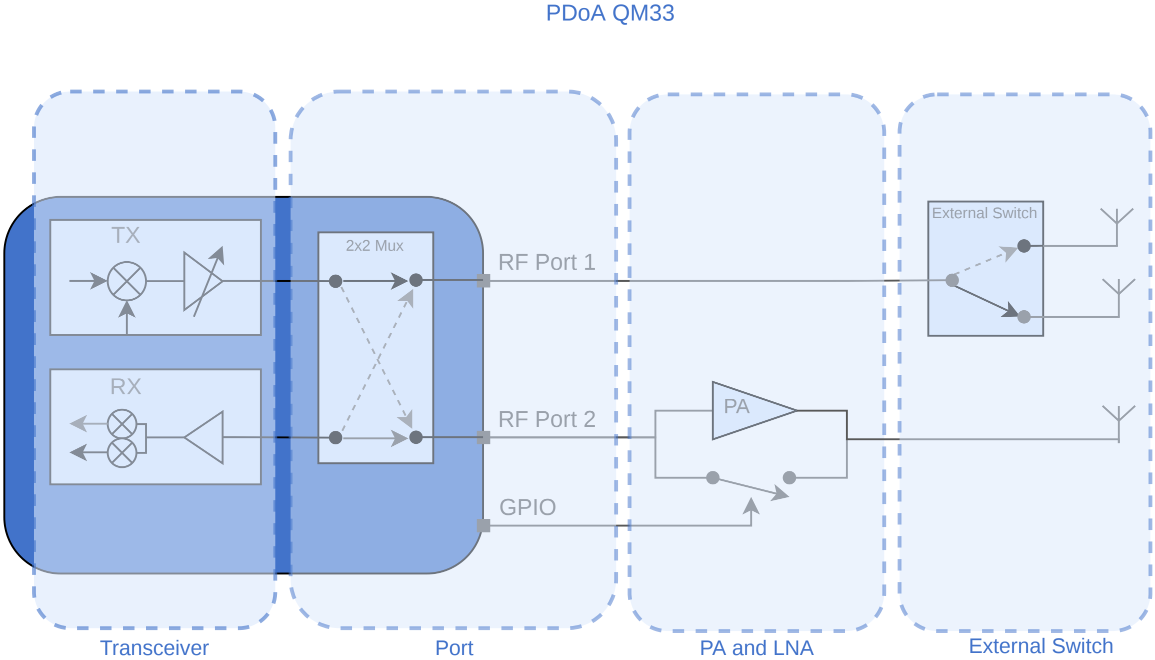

Fig. 3.2: Antenna Path components

The configuration keys allowing to configure all the elements composing the antenna path are the following:

- *ant_path<>.transceiver*
- *ant_path<>.port*
- *ant_path<>.lna*
- *ant_path<>.pa*
- *ant_path<>.ext_sw_cfg*

Note: QM33 and DW3000 do not contain internal LNA nor PA. The activation of *ant_path<>.lna* and *ant_path<>.pa* parameters only makes sense when the board design embeds external LNA or PA components.

3.2.2.1.2 Antenna Pair

An **Antenna Pair** defines a pair of Antenna Path used to measure one axis of *PDoA*/*AoA*.

The configuration keys allowing to configure an antenna pair are the following:

- *ant_pair<>.axis* defines the PDoA axis for which that pair must be used.
- *ant_pair<>.ant_paths* defines the two Antenna Path indexes making up the pair.

3.2.2.1.3 Antenna Set

An **Antenna Set** is a group which contains TX Antenna Path and/or RX Antenna Path(s) or Pair(s) fitting to a specific use case.

When an Antenna Set is used for PDoA measurement, it must always contain RX Antenna Pair(s). On the other hand, when the Antenna Set is not used for PDoA, it can contain simple RX Antenna Path(s).

The configuration keys allowing to configure an antenna pair are the following:

- *ant_set<>.tx_ant_path* defines the Antenna Path index to be used for TX frames.
- *ant_set<>.rx_ants_are_pairs* defines if RX antenna(s) indexes contained in the Antenna Set (`rx_ants`) reference Antenna Pairs, or simple Antenna Paths.
- *ant_set<>.nb_rx_ants* defines the number of Antenna Path(s) or Antenna Pair(s) contained in the set.
- *ant_set<>.rx_ants* is a table of up to 3 Antenna Path(s) or Antenna Pair(s).

3.2.2.2 Examples

3.2.2.2.1 Example 1: No PDoA

Let's create Antenna Set index 0 to be used for **non-PDoA frames**, and containing TX ANT1 and RX ANT2.

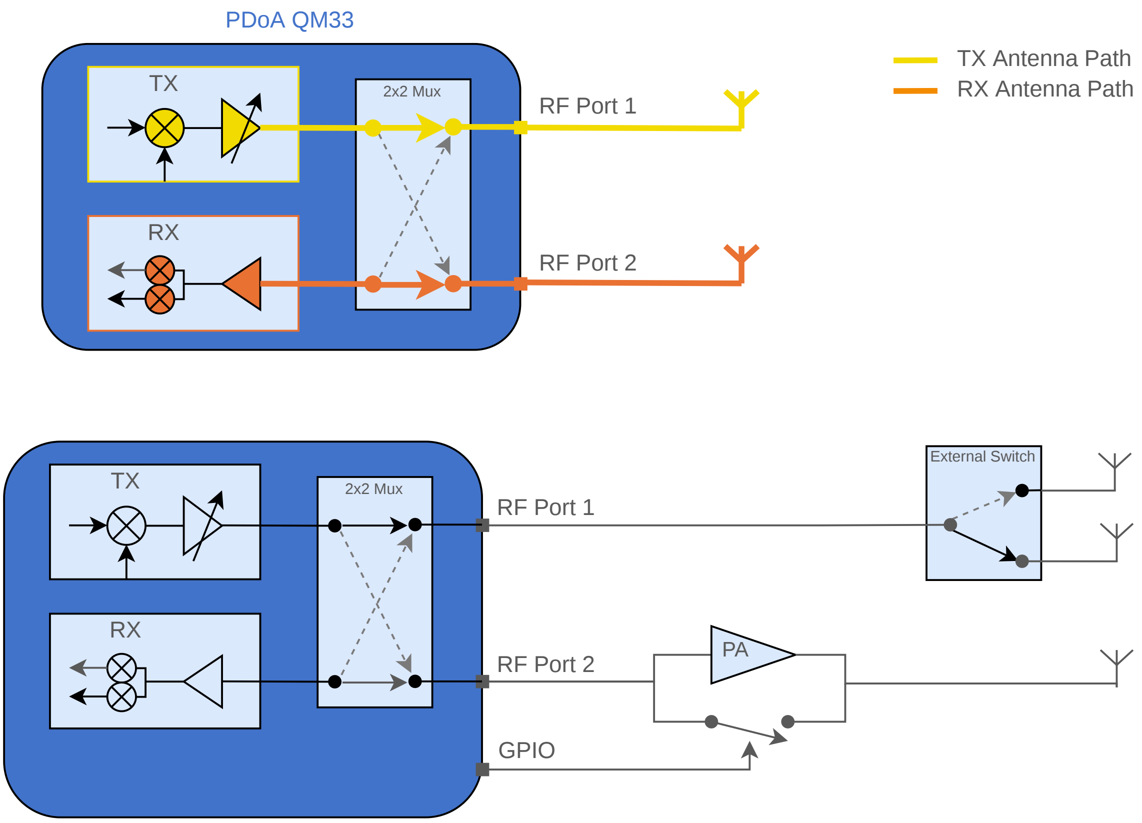

Fig. 3.3: Example of Antenna Set used for non-PDoA frames

- **Step 1:** Create the Antenna Paths:
  - Create the TX Antenna path on index 0:

```
ant0.transceiver = 0 # TRANSCEIVER_TX
ant0.port = 1 # RF port 1
ant0.ext_sw_cfg = 0 # No external switch or disabled
ant0.lna = false # No external LNA or disabled
ant0.pa = false # No external PA or disabled
```

  - Create the RX Antenna path on index 1:

```
ant1.transceiver = 1 # TRANSCEIVER_RXA
ant1.port = 2 # RF port 2
ant1.ext_sw_cfg = 0 # No external switch or disabled
ant1.lna = false # No external LNA or disabled
ant0.pa = false # No external PA or disabled
```

- **Step 2:** Create the Antenna Set index 0 containing the two Antenna Paths defined at step 1.

```
ant_set0.tx_ant_path = 0 # TX ant_path index = 0
ant_set0.rx_ants_are_pairs = false # Antenna Set does NOT contain pairs
ant_set0.nb_rx_ants = 1 # Antenna Set contains only 1 Antenna path
ant_set0.rx_ants = 0x000001 # Antenna Set uses RX ant_path index 1
```

3.2.2.2.2 Example 2: PDoA

Let's create Antenna Set index 1 to be used for **PDoA frames**, and containing TX ANT1 and a PDoA pair composed by RX ANT1 and ANT2.

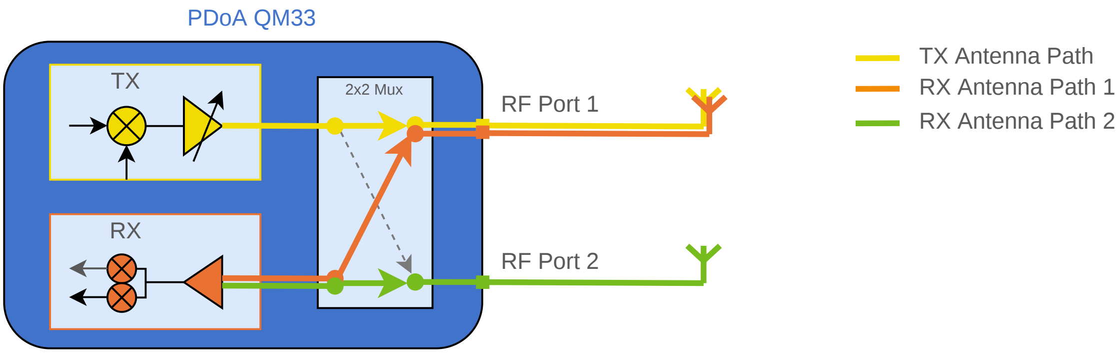

Fig. 3.4: Example of Antenna Set used for PDoA frames

- **Step 1:** Create the Antenna Paths:
  - Create the TX Antenna path on index 0:

```
ant0.transceiver = 0 # TRANSCEIVER_TX
ant0.port = 1 # RF port 1
ant0.ext_sw_cfg = 0 # No external switch or disabled
ant0.lna = false # No external LNA or disabled
ant0.pa = false # No external PA or disabled
```

  - Create the first RX Antenna path on index 1:

```
ant1.transceiver = 1 # TRANSCEIVER_RXA
ant1.port = 1 # RF port 1
ant0.ext_sw_cfg = 0 # No external switch or disabled
ant0.lna = false # No external LNA or disabled
ant0.pa = false # No external PA or disabled
```

  - Create the second RX Antenna path on index 2:

```
ant2.transceiver = 1 # TRANSCEIVER_RXA
ant2.port = 2 # RF port 2
ant0.ext_sw_cfg = 0 # No external switch or disabled
ant0.lna = false # No external LNA or disabled
ant0.pa = false # No external PA or disabled
```

- **Step 2:** Create the Antenna Pair index 0 containing the two RX Antenna Paths defined at step 1.

```
ant_pair0.axis = 0 # Used for azimuth measurement
ant_pair0.ant_paths = 0x0201
```

- **Step 3:** Create the Antenna Set index 1 containing the TX Antenna Path defined at step 1 and the RX Antenna Pair defined at step 2.

```
ant_set1.tx_ant_path = 0 # TX ant_path index = 0
ant_set1.rx_ants_are_pairs = true # Antenna Set contains pairs
```

(continues on next page)

(continued from previous page)

```
ant_set1.nb_rx_ants = 1 # Antenna Set contains 1 Antenna pair
ant_set1.rx_ants = 0x000000 # Antenna Set uses RX ant_pair index 0
```

3.2.3 PDoA/AoA Look-Up Tables

When using the PDoA version of the QM33/DW3000, the UWB transceiver allows to measure one *PDoA* values per RX frame. The translation from the PDoA to the *AoA* is allowed thanks to Look-Up Tables.

The maximum number of LUTs which can be configured is defined by the compilation flag `CONFIG_L1_CONFIG_PDOA_LUT_NUM`. The LUTs are configured using the configuration keys *pdoa_lut<>.data*.

A LUT is associated to an Antenna Pair, thanks to the configuration key *ant_pair<>.ch<>.pdoa.lut_id*.

In addition to the LUT, a **PDoA offset** can also be configured thanks to the configuration key *ant_pair<>.ch<>.pdoa.offset*.

3.2.4 Antenna Delay

The complete antenna path is associated to a specific antenna delay. That delay has an **impact on the ranging distance** measurement, and should be calibrated in order to take it into consideration when computing the distance.

The antenna delay **depends on all the components involved in the antenna path**: the transceiver used, the RF port, the presence of internal or external LNA or PA, the distance between the RF port and the physical antenna, etc. As a consequence, the antenna delay must be calibrated for each antenna path used in the final product.

The antenna delay also **depends on the channel** used to receive or transmit de frame. As a consequence, it must be calibrated for each channel used in the final product.

The procedure to calibrate one antenna delay, for a given antenna path and channel, is the following:

- The *DUT* is configured so that the antenna path being calibrated is used.
- The session is configured so that the channel being calibrated is used.
- A *TWR* ranging is performed between the DUT and a Reference Test Board in free space, with the two board units separated by a **known distance**.
- The antenna delay of the DUT is adjusted until the reported distance by the ranging is correct.

The antenna delay must be configured using the configuration key *ant<>.ch<>.ant_delay*.

4 TX Power

4.1 Overview

The transmitter output power used to send UWB frames can be configured.

Increasing the TX power will increase the power of the signal transmitted, thus increasing the distance reached by this signal. On the other hand, FCC regulations impose limitations on the highest TX power.

Calibrating the TX power consists in finding the highest TX power which complies to the regulation.

4.2 Functional

4.2.1 TX power index

The UWB stack offers a **linear and reliable API** to configure the TX power, which insures **30 dB effective Tx power dynamic without any significant leakage** (<1 dB).

That API, called **TX power index**, is defined by:

- **One unit** corresponds to an attenuation of **0.25 dB**.
- **Index 0** corresponds to **maximum TX power**.

The TX power index is a 32-bits value divided into four bytes, each of which specifies the TX power index for one part of the UWB frame: *STS*, *SHR*, *PHR* and DATA (PHY payload).

The TX power value may depend on the antenna path (including the port, the state of the PA, etc), and the channel used to transmit the signal. The configuration key *ant<>.ch<>.ref_frame<>.tx_power_index* allows to **configure the TX power index** for a specific antenna path, channel and *Reference frames*.

Note: It is the responsibility of the UWB stack to convert the TX Power index to the low-level configuration of the transceiver registers.

4.2.2 Reference frames

The UWB stack offers the possibility to the customers to calibrate the TX power considering “real life” UWB frames, in particular frame configurations that are used for FCC lab testing. It allows to insure that the TX power will never exceed the limitation for those specific frames.

Our solution brings the flexibility to define and calibrate the TX power for up to 8 reference UWB packets, called **reference frames**.

A reference frame is defined by a **PHY configuration and a payload size**.

The **definition of the reference frames** are configured via the configuration keys *ref_frame<>.phy_cfg* and *ref_frame<>.payload_size*.

A reference frame is:

- **Defined** when its PHY configuration and payload size are configured.
- **Calibrated** when it is defined, and its TX power indexes are configured.

The two first reference frames `ref_frame0` and `ref_frame1` have default values. Their definitions can be updated. However, the following rules must always be fulfilled:

- Ref frame 0 must always be defined for **BPRF mode**.
- Ref frame 1 must always be defined for **HPRF mode**.

Note: Even if QM33 does not support HPRF, `ref_frame1` should still remain defined for HPRF mode, but will end at being unused.

The solution requires calibrating at least the first reference frame `ref_frame0`.

When transmitting a UWB frame, the solution determines the reference frame having the closest characteristics to that TX frame, and uses its calibrated TX power as a reference.

When transmitting a UWB frame, the L1 determines if the characteristics of that frame match exact ones of a configured reference frame. If such reference frame exists, its calibrated TX power index is used as is.

If the transmitted frame does not correspond to an exact reference frame, the TX power index used is the one from the closest calibrated reference frame. That closest reference frame is determined thanks to the following algorithm:

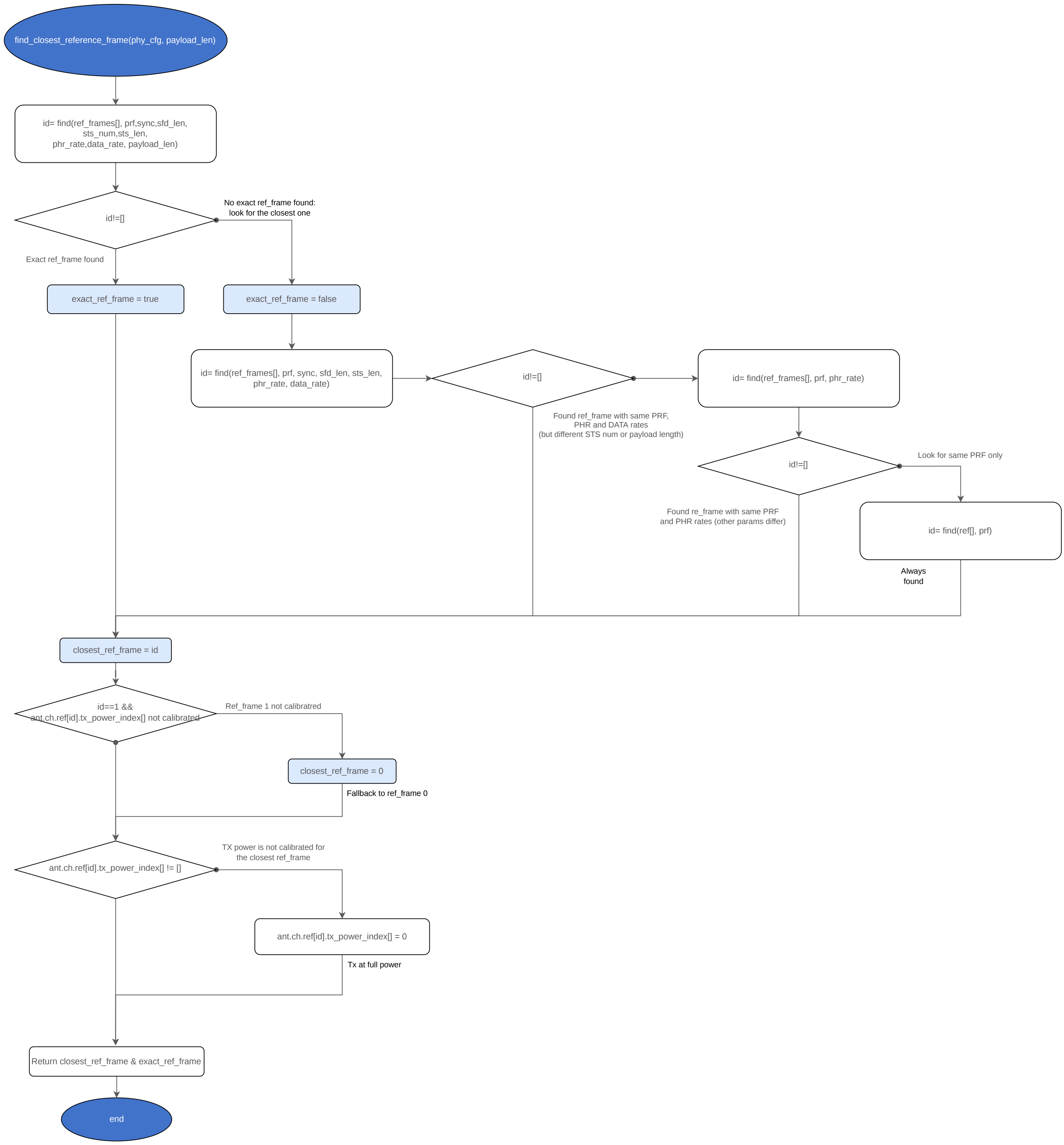

Fig. 4.1: Algorithm finding the closest reference frame

The itemization per reference frame also enables the *Adaptive TX Power* feature.

4.2.3 Adaptive TX Power

The highest TX Power is described by the mean power level over a reference time period. If the packet is shorter than the reference, the system is allowed to transmit with a proportionally higher power.

The **Adaptive TX Power** feature, sometimes also refered as the Smart TX Power, allows to optimize the TX power configured depending on the actual size of the transmitted frame.

Adaptive TX Power can be activated at runtime using the configuration parameter *ant_set<>.tx_power_control*.

In that context, *Reference frames* serve as “checkpoints” for Adaptive TX Power: **a unitary gain** (0 dB) is always applied when the **exact reference frame** is transmitted by the UWB system. It ensures the FCC test results can be reproduced whether Adaptive Tx power is enabled or not.

If the transmitted frame does not correspond to an exact reference frame:

- When Adaptive TX Power is **disabled**, the *TX power index* of the closest calibrated reference frame applies.
- When Adaptive TX power is **enabled**, the *TX power index* of the closest calibrated reference frame is adjusted so that average emissions for actual TX frame equals average emissions for the closest calibrated reference frame.

In order to prevent exceeding the peak power (0 dBm in 50 MHz) for shortest SP0/SP1 frame types, or to prevent PSD failure issues due to ePA compression, a limit applied on the compensation/boosting gain calculated in Adaptive TX Power can be configured. The configuration key *ant<>.ch<>.ref_frame<>.max_gating_gain* allows to configure that gain limit. Same as for TX power index, each byte corresponds to one section of the UWB frame: *STS*, *SHR*, *PHR* and DATA (PHY payload).

The Adaptive TX Power gain is determined by the following algorithm:

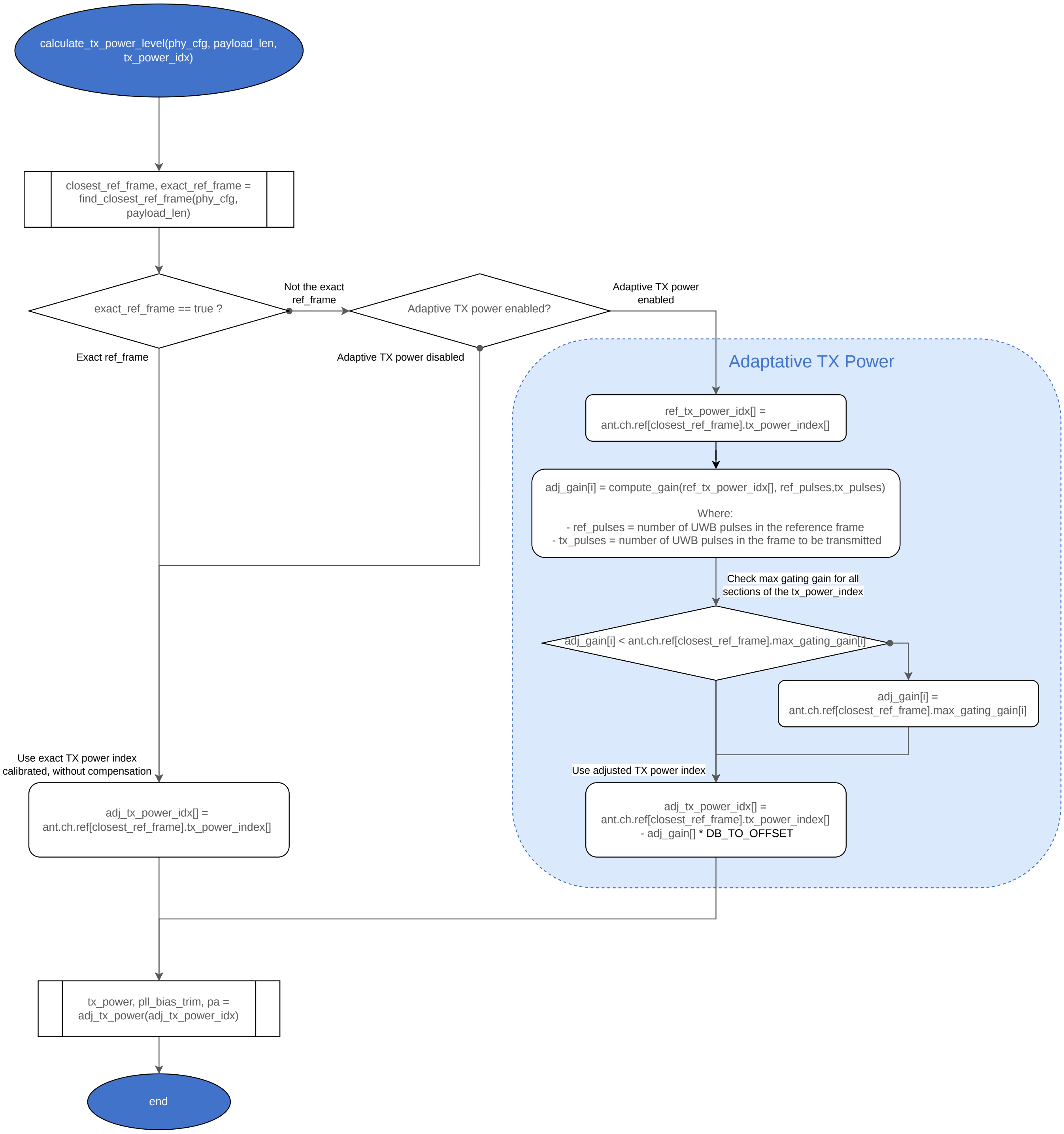

Fig. 4.2: Adaptive TX Power algorithm

4.2.4 Debug mode

In normal operation mode, the TX power and the PLL bias trim are automatically controlled depending on the TX power index configured, and the Adaptive TX Power state. However, in some debug or characterization and qualification context, the direct control of those settings is required.

The **debug mode** is enabled thanks to bit 7 of the configuration key *ant_set<>.tx_power_control*. When debug mode is enabled, the TX power and the PLL bias trim are respectively controlled via the parameters *debug.tx_power* and *debug.pll_cfg*.

References

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

FiRa. Uci test specification. Technical Report, FiRa, 2022.

FiRa. Uci test specification. Technical Report, FiRa, 2022.

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

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

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.

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

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

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

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

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.

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

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.

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

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.

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

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.

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

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.

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

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.

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

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

Index