QM33100 FAMILY USER MANUAL

HOW TO USE, CONFIGURE AND CONTROL THE QM33100 UWB TRANSCEIVER

This document is subject to change without notice

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 About the QM33100

The QM33100 is a family of fully integrated low power, single chip CMOS radio transceivers IC implementing HRP UWB PHY as specified by the IEEE802.15.4 standard [1], including the BPRF mode specified by the IEEE802.15.4z amendment [2]. There are currently two versions a non-PDoA and a PDoA device with 0xDECA0304 and 0xDECA314 device identifiers respectively.

- Supports UWB channels 5 and 9 (6489.6 MHz and 7987.2 MHz)
- Supports 2-way ranging, TDoA, and optionally AoA/PDoA location schemes
- Low external component count
- Supports enhanced Time-of-Flight security modes
- Integrated AES CCM* and AES GCM 128/192/256 functionality
- Worldwide UWB Radio Regulatory compliance
- Low power consumption (suitable for coin cell battery-powered applications)
- Data rates of 850 kb/s, and 6.8 Mb/s.
- Packet length from zero to 1023 octets
- Integrated MAC support features
- Up to 32 MHz SPI interface to host MCU
- Provides precision location and data transfer simultaneously
- Asset location to an accuracy of 10 cm
- High multipath fading immunity
- Supports high tag densities in RTLS
- Supports Japan and Korea regulatory compliance
- WLCSP52 (3.1 mm x 3.5 mm) package

Table 1: QM33100 variants

IC Variant	Type of package	PDoA support	Operating temperature
QM33110	WLCSP52	No	-40°C to +85°C
QM33120	WLCSP52	Yes	

1.2 About this document

This user manual describes the operation and programming of the QM33100 and discusses some of the design choices to be considered when implementing systems using it. Information already contained in the QM33100 Datasheet is not reproduced here and it is intended that the reader should use this user manual in conjunction with the QM33100 Datasheet. The document is divided into a number of sections each of which deals with a particular aspect of the QM33100 as follows:

Table 2: Brief description of document sections

Section	Section Name	Information covered
2	Overview of the QM33100	Gives an overview of the QM33100, describes how to interface to the device, and details its various operating modes
3	Message transmission	Describes the functionality and use of the transmitter
4	Message reception	Describes the functionality and use of the receiver
5	Media Access Control (MAC) hardware features	Describes the MAC level functionality provided by the IC.
6	Secure ranging / timestamping	Describes the secure timestamping capability of the device
7	Other features of the IC	Describes other features supported by the device.
8	The QM33100 register set	Desribes the register set in detail, lists all user accessible bit fields in each register and their respective functions.
9	Fast Commands	Describes the supported “fast commands”. Single octet SPI transaction command to place the device into TX or RX.
10	Calibration	Describes the parameters of the QM33100 that require calibration; the methodology that should be used in calibrating them and how often they require calibration.
11	Location schemes	Discusses some of the issues to be considered and trade-offs to be made when building systems based on the QM33100
12	APPENDIX 1: Two-way ranging	Gives an introduction to the use of the QM33100 in two-way ranging proximity systems
13	APPENDIX 2: Abbreviations and acronyms	Provides a list and explanation of abbreviations and acronyms used in the rest of the document
14	APPENDIX 3: References	Lists the documents referred to in this user manual
15	APPENDIX 4: Reference TX_POWER setting tables	Provides reference settings for which PHR byte is reduced relatively to DATA_PWR, SHR_PWR and STS_PWR bytes.
16	Document History	Gives the revision history of this document

Qorvo also provides QM33100 device driver software libraries. This software includes a set of API functions to initialize, configure and control the QM33100. It provides API functions for transmission and reception, and for driving the functionalities of the IC. The QM33100 driver code is targeted for the ARM Cortex-M. The API code comes with a number of simple examples that exercise the API and show how to use various features of the QM33100, including an example of two-way ranging [3].

2 Overview of the QM33100

2.1 Introduction

The QM33100 consists of an analog front-end (both RF and baseband) containing a receiver and transmitter and a digital back-end. The latter interfaces to a host processor control the analog front-end, accepts data from the host processor for transmission, and provides received data to the host processor over an industry standard SPI interface.

2.2 Comparison and compatibility with DW1000

The main differences to DW1000 are that the QM33100 has:

- Reduced peak and mean power consumption figures
- Addition of channel 9 (with 8 GHz center frequency)
- Reduced bill of materials for the full UWB solution by integrating the balun and filters on die
- Reduced external component count
- Increased ease of use for the end-user via the simplified software interface
- Allows for a single chip AoA/PDoA measurement
- Support for enhanced Time-of-Flight security modes using STS
- Integrated hardware AES (GCM and CCM*) 128/192/256 for data confidentiality and authenticity
- No support of the 110 kb/s data rate
- No support of channels 1, 2, 3, 4, 7
- No support for Smart Tx, this must be implemented on the host instead

2.2.1 Backwards compatibility with DW1000

QM33100 is backward compatible with DW1000 for a sub-set of use case configurations. As QM33100 has a reduced channel set, the only common channel is channel 5 (6.5 GHz). Both 16 and 64 MHz PRFs are supported as well as the 850 kb/s and 6.81 Mb/s data rates.
Although the QM33100 is operationally backward compatible with DW1000 on channel 5, modified software is needed since the QM33100 control register interface is different from that of the DW1000.

2.3 Interfacing to the QM33100

2.3.1 The SPI interface

The QM33100 host communications interface is a slave-only Serial Peripheral Interface (SPI) compliant with the industry protocol. The host system must include a master SPI bus controller in order to communicate with the QM33100

The host system controls the QM33100 via the SPI, reading and writing configuration and status registers, and data buffers, and issuing commands. This section describes the format of the SPI transactions. For details of the SPI bus signals, their voltage levels, operational mode configuration and timing parameters please refer to the QM33100 Datasheet [5]. The SPI-accessible buffers and registers of the QM33100 are detailed in section 8 - *The QM33100 register set* , and the commands are detailed in section *9 – Fast Commands*.

2.3.1.1 SPI operating modes

The operating mode of the SPI is determined when the QM33100 is initialised as a result of a device reset or is woken up from a sleep state. At this time GPIO lines 5 and 6 are sampled, (see Figure 11), and their values used to select the SPI polarity and phase mode respectively. Please refer to the IC Datasheet [5] section "SPI Operating Modes" for details of the operation resulting from this.

It is possible to set the SPI mode within the QM33100's one-time programmable (OTP) configuration block to avoid needing any external components and leave the GPIO free for alternative use. This is a one-time activity and cannot be reversed thus care must be taken to ensure that the desired SPI mode is correctly set if this method is used. Please refer to section *7.3 – Using the on-chip OTP memory* for more details of OTP configuration.

For full details of the SPI operating modes and their configuration, please refer to the QM33100 Datasheet.

2.3.1.2 Transaction formats of the SPI interface

Each SPI transaction starts with a one or two octet transaction header followed by a variable number of octets making up the transaction data. The size of an SPI transfer is not limited, however when writing to any of the QM33100 registers care must be taken not to write extra data beyond the published length of the selected register (see section 8 *– The QM33100 register set*) as doing so may cause the IC to malfunction.

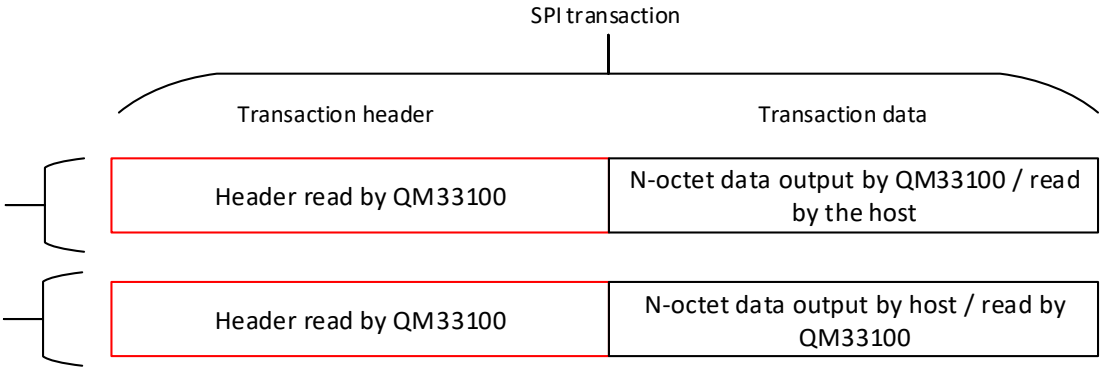

Figure 1: SPI transaction header and transaction data

Physically the SPI interface is full duplex in that every transaction shifts bits both into and out of the QM33100. Logically however each transaction is either reading data from the QM33100 or writing data to it. All SPI communication shown here is from the SPI-Master (host microprocessor) point of view. Thus an SPI read means reading data from QM33100 and SPI write means writing data into QM33100. As shown in Figure 1, for a read transaction all octets beyond the transaction header are ignored by the QM33100, and for a write transaction all octets output by the QM33100 should be ignored by the host system. The SPI transaction header selects the SPI transaction format, shown in Figure 2.

Note: The octets are physically presented on the SPI interface data lines with the high order bit sent first in time (MSB first). The first bit in the transaction sequence determines the direction of the communication, which is 0 for the SPI read and 1 for the SPI write operation.

Table 3: SPI transaction types

SPI transaction type	Description
Fast command transaction	Write a single octet command specified by the “Fast Command” field
Short addressed transaction	Read or write any of the 32 register file (IDs) in the range 0x00 to 0x1f as specified by the 5-bit base address field.
Full addressed transaction	Read or write any of the 32 register files, with a sub-address (an octet offset index) in the range 0x00 to 0x7F as specified by the 7-bit sub-address field
Masked write transaction	A write only transaction allowing changes to sub-fields within any register file with sub-address. This transaction includes an AND-ing mask to clear selected bits in the destination register and an OR-ing mask to set selected bits in the destination register. This command removes the need for the host to do a *read-modify-write* when it needs to set a sub-field in a register. **Note:** The masked write transaction should not be used on “write-1-to-clear” event status bits.

SPI transactions are enveloped by the assertion of the active low chip select line, SPICSn. The high-to-low assertion (low) of SPICSn initialises the SPI transaction handler so that the QM33100 interprets the next octet(s) as a new transaction header. The low-to-high de-assertion of SPICSn ends the SPI transaction.

The SPI accessible parameters of the QM33100 are organised into 32 separate register file locations and further detailed in section 8 – The QM33100 register set. Every SPI read or write transaction header includes a 5-bit register file ID – ***5-bit base address*** – as shown in Figure 2, that identifies the register file is being accessed by the transaction. Sub-addressing within the selected register file allows efficient access to all the parameters within the QM33100. Depending on the sub-addressing being used, the transaction header is either one or two octets long. These three types of transaction are described in the sub-sections below.

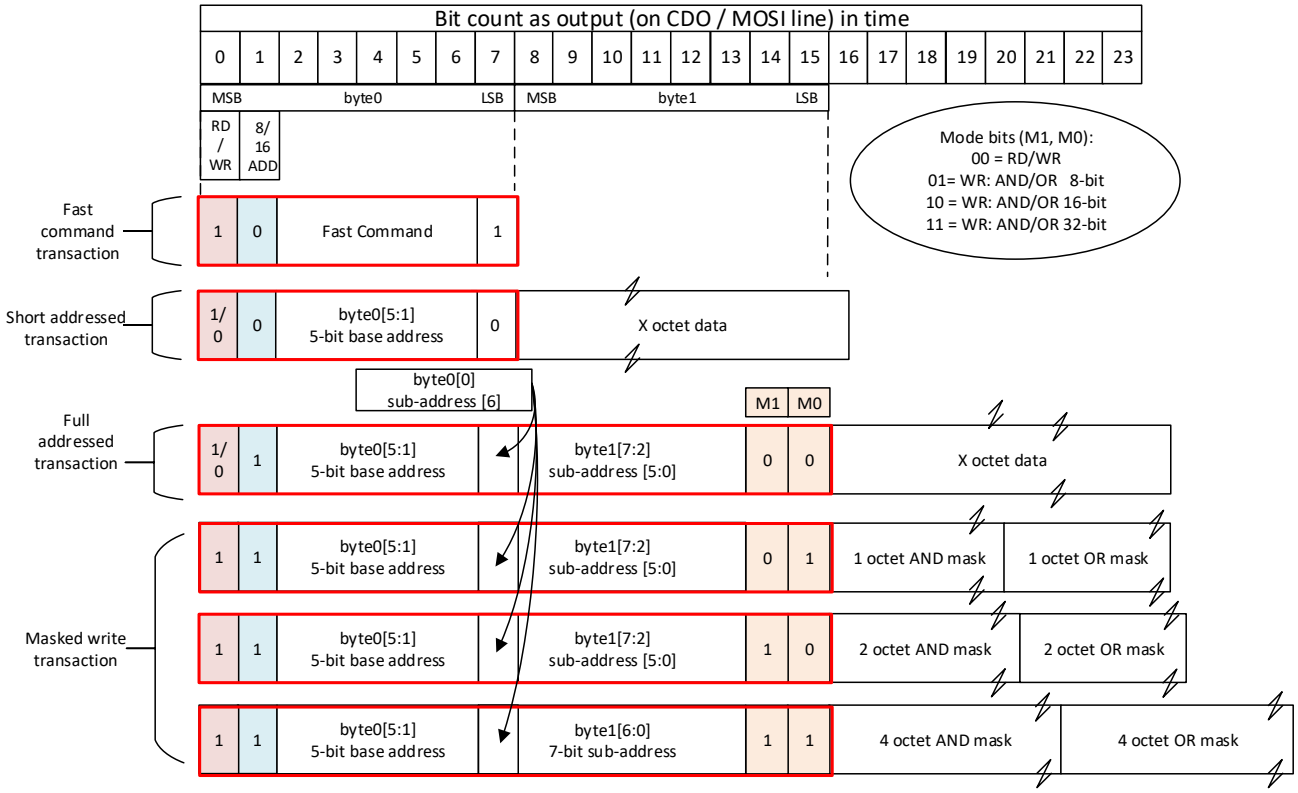

Figure 2: SPI command formatting

2.3.1.2.1 SPI transaction with a 1-octet header – fast command

shows the fields within the one octet transaction header of a fast command SPI transaction. The 5-bit fast command hex code is encapsulated by control bits which specify this SPI header as fast command type. In the fast command is a start transmission command (CMD_TX, hex code 0x1),

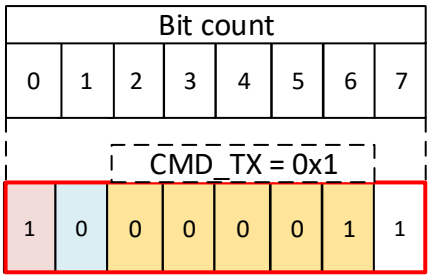

Figure 3: Single octet fast command (start TX) SPI transaction

2.3.1.2.2 SPI transaction with a 1-octet header

Figure 4 shows the fields within the one octet transaction header read/write command. The 5-bit register file ID is encapsulated by control bits which specify this SPI header as a read/write transaction with a single octet header. This example is accessing the DEV_ID register (0x00:00). Only the first octet read from the 32-bit DEV_ID register is shown with value of 0x14. Note this value may be different depending on the IC version see § 8.2.2.1 – Sub-register 0x00:00 – Device Identifier.

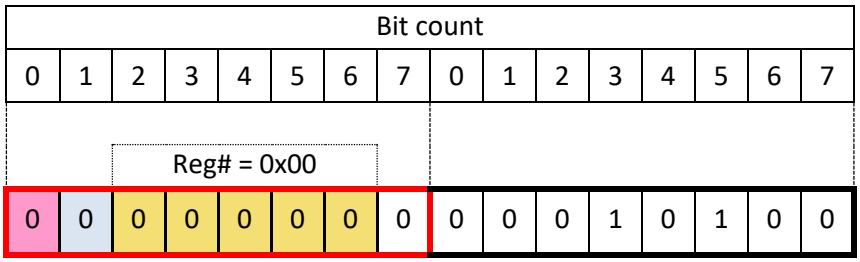

Figure 4: Reading first octet from DEV_ID register of the IC

2.3.1.2.3 SPI write transaction with a two-octet header

Figure 4 shows the fields within the two-octet transaction header of an extended address write SPI transaction. The 5-bit base address (0x2) and 7-bit sub address (0x1C) are encapsulated by control bits which specify this SPI header as an extended address write type. Figure 4 shows the octets sent to the device to write 2 bytes to register STS_IV (reg ID 0x2 sub address 0x1C). As this is a write type the mode bits (M1 and M0) are set to 0.

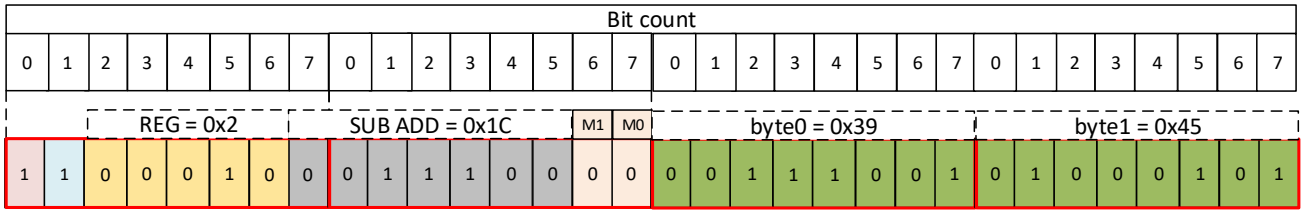

Figure 5: writing two octets to register 0x2 subaddress 0x1C

2.3.1.3 SPI CRC mode

The SPI CRC mode when enabled provides the ability for SPI transactions to have the added protection of a Cyclic Redundancy Check sequence. This mode of operation is disabled by default, but may be enabled (and disabled) via the SPI_CRCEN bit in the SYS_CFG register.

Note: The SPI interface is a local digital interface which should never experience error, so the enabling of SPI CRC checking is generally unwarranted. While SPI CRC checking has a penalty in terms of the added software overhead of the host microprocessor having to calculate a CRC for every SPI write and read transaction, it may be useful for extra reliability reasons, for example where the design has long SPI lines with a possibility of interference, or in the case of design debugging where SPI integrity has not been proven.

When SPI CRC mode is enabled, the QM33100 behaves as follows:

2.3.1.3.1 SPI writes in SPI CRC mode

For SPI writes from the host to the QM33100 when SPI CRC mode is enabled, the IC assumes that each SPI transaction ends with an 8-bit CRC calculated over all the preceding bytes written during the SPI transaction, i.e. from the falling edge of SPICSn. When SPICSn is de-asserted the final CRC byte of the SPI transaction is checked by the IC against a CRC it has generated for the transaction. If the CRC from the host does not match a CRC the IC generates internally, then this is considered to be an error, and the error is signalled to the host via the SPICRCE event bit in the system event status register, SYS_STATUS. This event flag may be used to generate an interrupt if the event is unmasked via the SPICRCE_EN bit in the SYS_ENABLE register.

Note: The write command will complete irrespective of the CRC error check status, which means that the data may be written to the wrong register location, and/or the data may be written incorrectly. The 8-bit CRC itself is not written to the addressed location,

The recommend recovery from a write CRC error is to reset the QM33100 completely, reinitialising and reconfiguring it into the desired operating mode for the application.

2.3.1.3.2 SPI reads in SPI CRC Mode

For host SPI reads of QM33100 registers when SPI CRC mode is enabled, the IC calculates an 8-bit CRC over the whole the SPI transaction, i.e. from the falling edge of SPICSn at the start of reading to its de-assertion at the end of the read. The CRC thus covers the 1 to 3 header bytes written by the host to initiate the read, and all the subsequent bytes output as the host continues the read transaction. The resultant CRC value is placed into the SPI_RD_CRC register in *Sub-register 0x00:18 – SPI CRC read status.*

A host wishing to validate the CRC of an SPI read transaction, must calculate its own CRC value across the header bytes it writes and all the data bytes it subsequently reads for the transaction, and then must read the IC calculated CRC from the SPI_RD_CRC register, and compare it to the host calculated CRC. If these don't match then the host has detected an error in the SPI read.

In the event of an SPI read error, the read can just be repeated, except in the unlikely case that the error changed the read into a write. In that case a write CRC error will be triggered and this can be handled as before by resetting the QM33100 and reinitialising/reconfiguring it into the desired operating mode.

2.3.1.3.3 The SPI CRC polynomial / implementation

The CRC is 8-bits long, based on the CRC-8-ATM which has generator polynomial:

$$G(x) = x^8 + x^2 + x + 1$$

and has a typical implementation as shown in Figure 6, where the circle operator is an exclusive-OR.

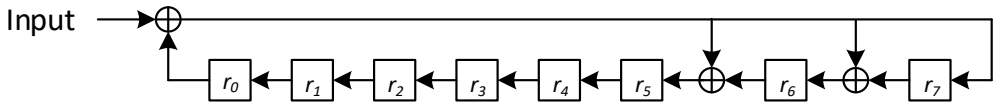

Figure 6: SPI CRC polynomial implementation

In the QM33100 the shift register (r_0 to r_7) is initialised, at the start of each read or write SPI, with the value specified in the SPICRCINIT register, which by default is all zeros.

Note: The source code of the QM33100 device driver and API [3] includes functions to generate and check the SPI CRC along with a simple example that demonstrates their use. The maximum SPI rate supported by QM33100 when the SPI CRC mode is enable is 20 MHz.

2.3.2 Interrupts

The QM33100 can be configured to assert its IRQ pin on the occurrence of one or more status events. The assertion of the IRQ pin can be used to interrupt the host controller and redirect program flow to handle the event.

The polarity of the IRQ pin may be configured via the HIRQ_POL bit in the Sub-register 0x0F:28 – Digital diagnostics test mode control. By default, on power up the IRQ polarity is active high. This is the recommended polarity to ensure the lowest power operation of the QM33100 in SLEEP and DEEPSLEEP device states. This pin will float in SLEEP and DEEPSLEEP states and may cause spurious interrupts unless pulled low.

The occurrence of a status event in the SYS_STATUS register may assert the IRQ pin depending on the setting of the corresponding bit in the SYS_ENABLE register.

By default, on power-up, the SPIRDY interrupt generating event enable SPIRDY_EN is set to 1 so that the interrupt is enabled.

2.3.3 General purpose I/O

The QM33100 provides a number of GPIO pins as listed in the QM33100 Datasheet [5]. These can be individually configured at the user's discretion to be inputs or outputs. The state of any GPIO configured as an input can be read by the host controller over the SPI interface. When configured as an output the host controller can set the state of the GPIO to high or low. Some of the GPIO lines have multiple functions as listed in the QM33100 Datasheet.

The configuration and operation of the GPIO pins are controlled via GPIO_CTRL register. By default, on power-up, GPIO pins 0-3 are reserved and software should switch to preferred mode on startup; GPIO pins 4-7 are configured as inputs, and GPIO8 is configured as IRQ by default.

2.3.4 The SYNC pin

This pin is used for external clock synchronization purposes. See section *7.1 – External Synchronisation* for further details.

2.4 QM33100 operational states

2.4.1 State diagram

The QM33100 has a number of different operational states. These are listed and described in Table 4 below and the transitions between them are illustrated in Figure 7.

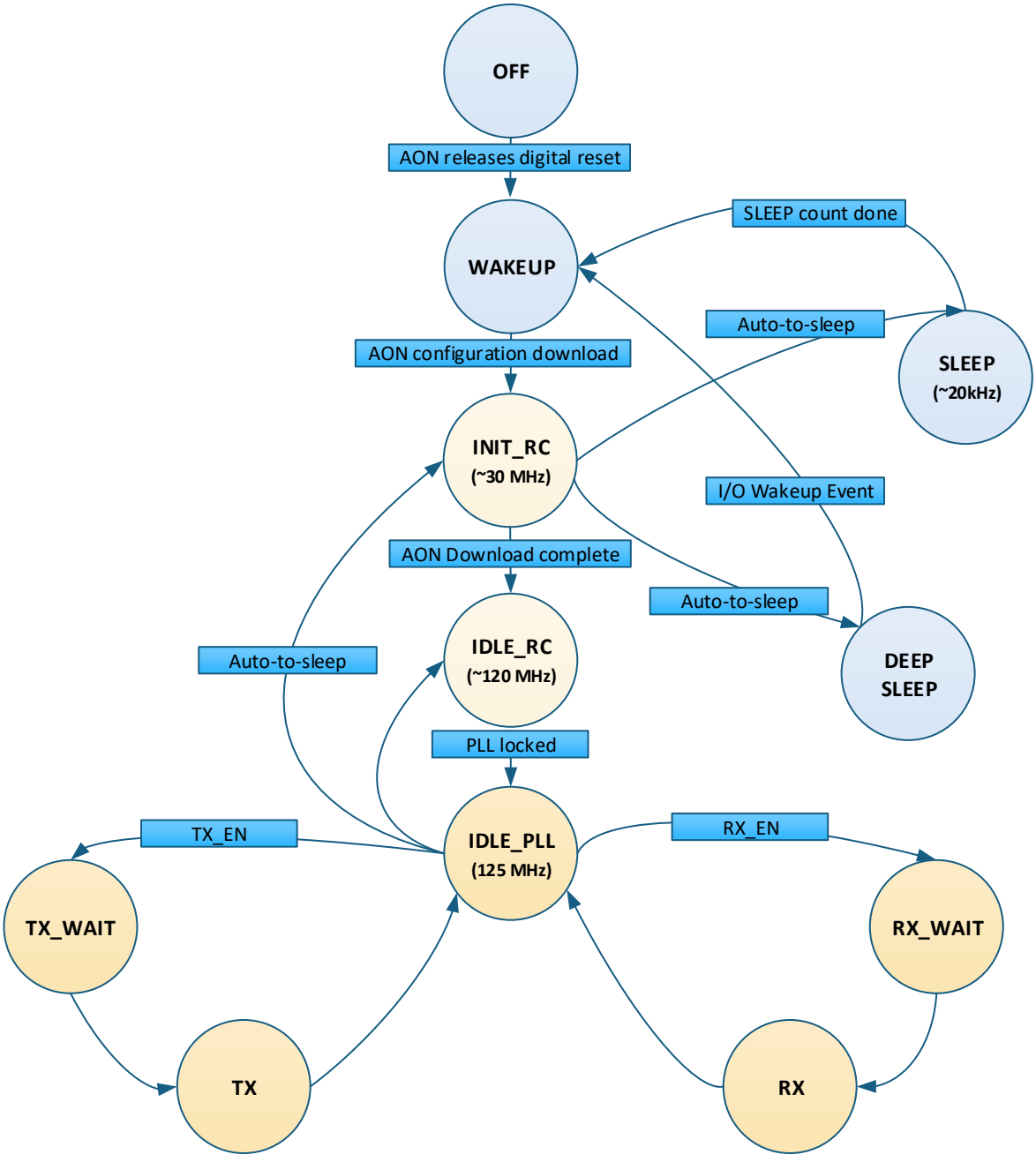

Figure 7: QM33100 state diagram

2.4.2 Overview of main operational states

Table 4: Main QM33100 operational states / modes

State Name	State Description
OFF	In the **OFF** state, the QM33100 is completely powered off, with no voltages applied to any of its input pins. Power consumption = 0 µA. No I/O pins should be driven or power will leak through the I/O cells.
WAKE_UP	During the **WAKE_UP** state, the AON sequencer starts up all of the primary power and clock blocks in order to achieve host comms in the INIT_RC state. The device will automatically transition from the **WAKE_UP** state to the INIT_RC state.
INIT_RC	Lowest power state with SPI access, but limited to 7 MHz. The system is clocked off a divide-by-four of the FAST_RC clock, i.e., the IC runs at approximately 30 MHz.
IDLE_RC	Lowest power state allowing full speed SPI access. The FAST_RC oscillator is used as the system clock source, i.e., the IC runs at approximately 120 MHz.
IDLE_PLL/IDLE	In the **IDLE_PLL** state the QM33100 internal clock generator PLL is locked running at its nominal 125 MHz rate and ready for use but it is gated off to most circuitry to minimize power consumption. In the **IDLE_PLL** state, SPI communications can operate at up to 38 MHz, the maximum SPICLK frequency. In the **IDLE_PLL** state, the analog receives and transmit circuits are powered down. The external host can control the QM33100 to initiate a transmission or reception and thus cause the QM33100 to progress into **TX** or **RX** state respectively. If a delayed TX or RX operation is initiated (see section *3.3 – Delayed transmission* and *4.3 - Delayed receive*) then the QM33100 will enter the **TX_WAIT** or **RX_WAIT** (until the delayed time has elapsed) and then enter **TX** or **RX** state.
SLEEP	In the **SLEEP** state, the IC consumes < 1 µA from the external power supply inputs. All internal LDOs are turned off. In the **SLEEP** state, the QM33100 internal low power oscillator is running and is used to clock the sleep counter whose expiry is programmed to “wake up” the QM33100 and progress into the **WAKE_UP** state. While the IC is in the **SLEEP** state, the external system should avoid applying power to GPIO, SPICLK, or SPICDI pins as this will cause an increase in the leakage current. While the device is in the **SLEEP** state SPI communication is not possible.

State Name	State Description
DEEPSLEEP	With the exception of the **OFF** state, the **DEEPSLEEP** state is the lowest power state of the device. In this state, the IC consumes < 250 nA from the external power supply inputs. In **DEEPSLEEP** all internal circuitry is powered down with the exception of the always-on memory which can be used to hold the device configuration for restoration on wake up Once in **DEEPSLEEP**, the QM33100 remains in this state until the occurrence of a wake up event. This can be either: 1. the **SPICSn** line pulled low or 2. the **WAKEUP** line driven high for the duration quoted in the QM33100 Datasheet [5] (nominally 500 µs). Once the QM33100 has detected a wake up event it progresses into the **WAKE_UP** state. While the IC is in **DEEPSLEEP** state, the external system should avoid applying power to GPIO, SPICLK, or SPICDI pins as this will cause an increase in leakage current. Note: Asserting the RSTn pin (hardware reset) will also take the device out of DEEPSLEEP, however, the device will be fully reset in that case. While the device is in the **DEEPSLEEP** state SPI communication is not possible.
TX_WAIT	This state is very like the **IDLE_PLL** state except the IC is implementing a delay prior to transmission, typically aiming to transmit the RMARKER at a specified time. At the appropriate moment the TX analog blocks are turned on and the device transitions into the **TX** state.
RX_WAIT	This state is very like the **IDLE_PLL** state except the IC is implementing a delay prior to turning on the receiver. At the appropriate time the RX analog blocks are turned on and the device transitions into the **RX** state.
TX state	In the **TX** the QM33100 actively transmits a packet (generally containing the contents of the transmit buffer) on the configured RF channel with the configured transmit parameters (PRF, data rate, preamble code, etc.) Once the transmission is complete the QM33100 will go back to the **IDLE_PLL** state, following which it may enter **RX** state (e.g. if using wait for response command) or go to a low-power state depending on the programmed configuration. If the ATX2SLP bit is set (in SEQ_CTRL) the QM33100 will enter the **SLEEP** or **DEEPSLEEP** state automatically, see “Auto-to-sleep” path in Figure 7 (as long as no host interrupts are pending).

State Name	State Description
RX state	In the **RX** state, the receiver is active, either hunting for preamble or (once it has detected preamble) actively receiving preamble searching for the start of frame delimiter (SFD), and subsequently receiving the rest of the packet. In the **RX** state, the RF synthesizer and all RX blocks are active. After an event that ends the reception, (either a good frame/packet reception or some error or timeout event that aborts reception) the QM33100 will return to the **IDLE_PLL** state unless ARX2SLP bit is set (in SEQ_CTRL) in which case the IC will enter the **SLEEP** or **DEEPSLEEP** state automatically, see "Auto-to-sleep" path in Figure 7 (as long as no host interrupts are pending). Note that it is not possible to be in the RX and TX states simultaneously – the QM33100 is a half-duplex transceiver device.

2.4.3 Clock periods and frequencies

The chipping rate specified for the HRP UWB PHY by the IEEE 802.15.4 standard [1] is 499.2 MHz, and all IC system clocks are referenced to this frequency. Where the system clock frequency is quoted as 125 MHz, this is an approximation to the actual 124.8 MHz system clock frequency (crystal 38.4 MHz × 13 ÷ 4). Similarly, where the system clock period is quoted as 8 ns, this is an approximation to the actual period of $1/(124.8\times10^6)$ seconds.

The 1 GHz PLL clock, where referenced, is an approximation to its actual frequency of 998.4 MHz.

A 63.8976 GHz sampling clock is associated with ranging for the IEEE802.15.4 standard [1], where a 15.65 picosecond time period is referred to, it is an approximation to the period of this clock.

2.4.4 Pulse repetition frequency (PRF)

The PRF values of 16 MHz and 64 MHz quoted in this document are approximations to the values dictated by the IEEE802.15.4 standard [1]. PRF mean values are slightly higher for SHR compared to the PHR and data parts of the packet. Please refer to [1] for full details of peak and mean PRFs.

2.4.5 Data rate

Where a data rate of 6.8 Mb/s is referred to, this is equivalent to the 6.81 Mb/s data rate in [1]. Note that the data rates quoted are (rounded) nominal values based on the data symbol rate multiplied by the Reed-Solomon (RS) coding rate, 0.87 which is 330/378 because the RS coding adds 48 parity bits for every 330 bits of data (or part thereof). Please refer to [1] for more details.

2.5 Power On Reset (POR)

When the external power source is applied to the QM33100 for the first time (cold power-up), the internal Power-On Reset (POR) circuit compares the externally applied supply voltage (VDD1) to an internal power-on

threshold (approximately 1.5 V), and once this threshold is passed the AON block is released from reset and the external device enable pin EXTON is asserted. This is shown in Figure 8.

Then the VDD2a and VDD3 supplies are monitored and once they are above the required voltage as specified in the Datasheet (2.2 V and 1.4 V respectively), the fast RC oscillator (FAST_RC) and crystal (XTAL Oscillator) will come on within 500 µs and 1 ms respectively. **However, if the time for VDD2a or VDD3 exceeds 10 ms then the device will need to be reset once these supplies are up.** Please refer to the Datasheet [5] for more details.

The QM33100 digital core will be held in reset until the crystal oscillator is stable. Once the digital reset is de-asserted the digital core wakes up and enters the **INIT_RC** state, (see Figure 7 and Figure 8). Then once the configurations stored in AON and OTP have been restored (into the configuration registers) the device will enter the **IDLE_RC** state**IDLE_RC**. Then the host can set the AINIT2IDLE configuration bit in SEQ_CTRL and the IC will enable the CLKPLL and wait for it to lock before entering the **IDLE_PLL** state.

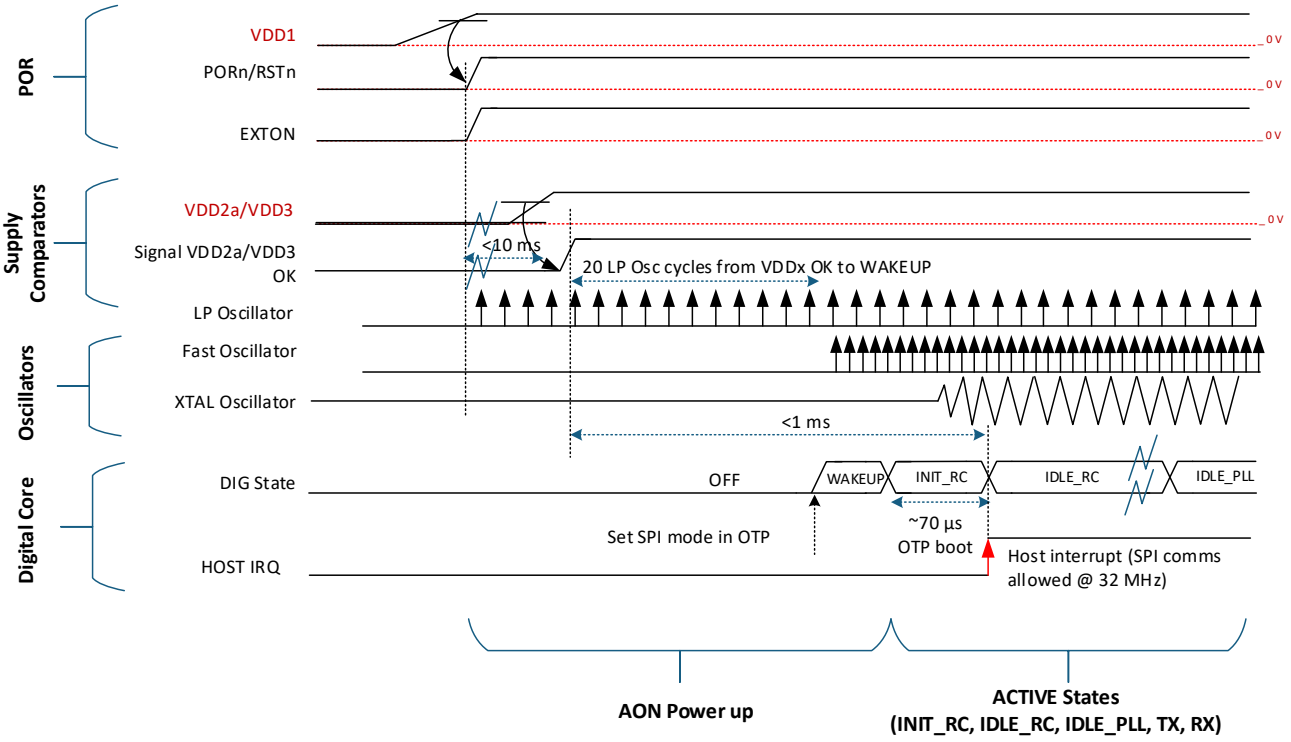

Figure 8: Timing diagram for cold start POR

2.5.1 SLEEP and DEEPSLEEP

In the very low power **DEEPSLEEP** state, the IC is almost completely powered down except for a small amount of "always-on" memory necessary to maintain IC configurations. This is the lowest power mode of the IC where the power drain is approximately 200 nA. To wake the IC from **DEEPSLEEP** state requires an

external agent to assert the **WAKE_UP** input line or the external host microprocessor to initiate an SPI transaction to assert the SPICSn input.

The QM33100 also includes a low power **SLEEP** state where the IC can wake itself from sleeping as a result of the elapsing of a sleep timer (see 8.2.11.6.1 for sleep timer configuration) that is running from a low-powered oscillator internal to the QM33100 IC. In this **SLEEP** state, the power drain is approximately 1 µA. The IC will wake from the **SLEEP** state when the sleep timer elapses, or the **WAKE_UP** or **SPICSn** inputs may be used to wake the device before the timer elapses.

The frequency of the low power oscillator is dependent on process variations within the IC but is typically around 20 kHz. There are facilities within the IC to measure the frequency of this LP oscillator and also to trim it.

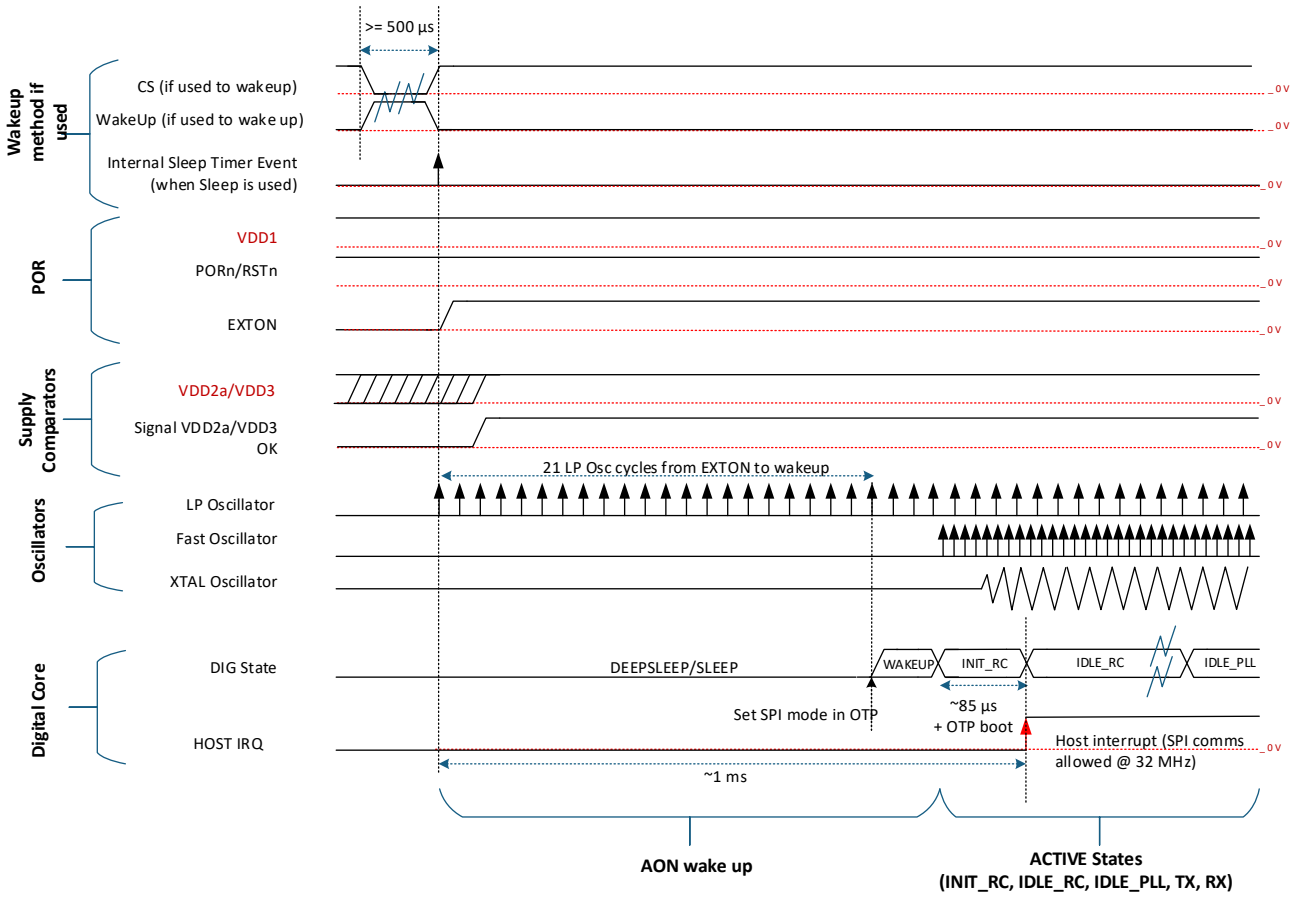

Figure 9: Timing diagram for warm start

2.5.1.1 Waking from sleep

Waking from **SLEEP** and **DEEPSLEEP** states can happen in the following ways:

- Driving the **WAKEUP** pin high for approximately 500 µs, (assuming WAKE_WUP configuration bit is set in AON_CFG).

- Driving the **SPICSn** pin low for approximately 500 µs, (assuming the WAKE_CSN configuration bit is set in AON_CFG). This can be achieved by doing a dummy SPI read of sufficient length.

  Note: When using the SPICSn pin to wake up the device, it is important that the SPICDO line is held low for the duration of the SPICSn to ensure that a spurious write operation does not occur.

In addition return from **SLEEP** also occurs when

- The internal sleep timer counter expires, (assuming the WAKE_CNT configuration bit is set in AON_CFG along with an appropriate SLEEP_TIM value).

In all three wake-up cases, the device is returned to the **IDLE_PLL** state if the AINIT2IDLE configuration bit in SEQ_CTRL register was set when the device configuration was uploaded prior to entering a sleep state, otherwise, the device will stay in **IDLE_RC**. Additional state transitions can be automatically enacted thereafter depending on configurations.

Asserting the RSTn pin (hardware reset) will also take the IC out of **SLEEP** or **DEEPSLEEP** states, however, the device will be fully reset in that case.

2.5.1.2 Configuration register preservation

Prior to entering the **SLEEP** or **DEEPSLEEP** states the main QM33100 configuration register values are saved (copied) into the Always-On (AON) memory, and upon waking, prior to exiting the **INIT_RC** state the saved values are restored from the AON memory.

Power is maintained to the AON memory at all times, even in **SLEEP** and **DEEPSLEEP** states. The copying of configuration data (saving or restoring) and boot up time of the OTP takes ~85 µs to complete (when restoring from **SLEEP** and **DEEPSLEEP** states, see Figure 9). Restoration of configurations during the **WAKE_UP** state is only done if the ONW_AON_DLD configuration bit is set in AON_DIG_CFG.

Note: The host should wait for the restoration of configurations to be completed before using SPI to avoid internal IC conflicts that may lead to the corruption of the configuration values. The preferred option is to wait for the assertion of the SPIRDY interrupt.

Table 5: Configurations maintained in the AON memory array

Configuration Register	Configuration Register
AON_DIG_CFG [23:0]	GPIO_MODE [31:0]
XTAL [7:0]	DTUNE0 [15:0]
PLL_CAL [7:0]	RX_SFD_TOC [15:0]
PANADR [31:0]	PRE_TOC [15:0]
SYS_CFG [31:0]	DTUNE3 [31:0]
FF_CFG [15:0]	
TX_FCTRL [47:0]	PLL_CFG [15:0]
DREF_TIME [31:0]	CIA_CONF [31:0]
RX_FWTO [23:0]	FP_CONF [31:0]
SYS_ENABLE [47:0]	IP_CONF [31:0]
TX_ANTD [15:0]	STS_CONF_0 [31:0]
ACK_RESP_T [31:0]	STS_CONF_1 [31:0]
TX_POWER [31:0]	SEQ_CTRL [31:0]
CHAN_CTRL [15:0]	TXFSEQ [31:0]
AES_IV0 [31:0]	LED_CTRL [15:0]
AES_IV1 [31:0]	RX_SNIFF [31:0]
AES_IV2 [31:0]	DGC_CFG [15:0]
AES_IV3 [31:0]	RF_SWITCH [31:0]
DMA_CFG [31:0]	[15:0]
AES_KEY [127:0]	CIA_ADJUST [15:0]
STS_CFG [15:0]	
STS_KEY [127:0]	
STS_IV [127:0]	

2.5.1.3 Loading LDO calibration data from the OTP

When waking from **SLEEP** or **DEEPSLEEP** it is necessary to load the *LDOTUNE_CAL* value that is programmed into OTP during IC production test calibration.

To confirm if the *LDOTUNE_CAL* has been programmed first read the OTP address 0x4. If this is non-zero (only the first byte needs to be checked) then the device has been calibrated, and the value should be written to the LDOTUNE register.

2.6 Default configuration on power up and after a reset

QM33100 is a highly configurable transceiver with many features. The register default (reset) values have been selected with the intention of minimizing the amount of user configuration required. The default configuration is summarised in Table 6.

Table 6: Default QM33100 operational configuration

Parameter	Default Value
Channel	5 (C_F is 6489.6 MHz)
Data Rate	6.8 Mb/s
PHR Rate	850 kb/s
PRF	64 MHz
Preamble Length	64 symbols
Preamble Code	9
Scrambled Timestamp Sequence (STS)	off
STS Sequence Length	n/a
SFD	IEEE802.15.4z [2] length 8

Channel numbers and preamble codes are as specified in the standard, IEEE802.15.4 standard [1]. Some further details are given below on the specifics of the default device configuration. For full details please refer to the register map where the default value of each register is given,

2.6.1 Default system configuration

Much of the system configuration is configured in the SYS_CFG register, please see section *Sub-register 0x00:10 – System configuration* for a full description of the register contents and defaults.

By default, interrupt polarity is active high and all interrupts are disabled, see HIRQ_POL in the DIAG_TMC register for interrupt polarity and the SYS_ENABLE and SYS_STATUS registers for interrupt configuration and information, see sections *Sub-register 0x00:3C – System event enable* mask and *Sub-register 0x00:44 – System event status*.

GPIOs are all set to mode 0 (as defined in the GPIO_MODE register), their default function is shown in Table 7.

Table 7: GPIO default functions

GPIO Pin	Default Function	Direction at Reset
GPIO0/RXOKLED	Reserved	Input
GPIO1/SFDLED	Reserved	Output
GPIO2/RXLED	Reserved	Input

GPIO Pin	Default Function	Direction at Reset
GPIO3/TXLED	Reserved	Input
GPIO4/COEXIN/EXTTXE(PDOA_SW_0)	GPIO4	Input
GPIO5/COEXOUT/EXTRXE(PDOA_SW_1)	GPIO5	Input
GPIO6/EXTRXE/PDOA_SW_2	GPIO6	Input
SYNC/GPIO7/PDOA_SW_3	SYNC	Input
IRQ/GPIO8	IRQ	Output

Note: For further details about Pin configuration, see the QM33100 Datasheet [5].

Frame wait timeout (see SYS_CFG register bit RXWTOE and *Sub-register 0x00:34 – Receive frame wait timeout period*) and preamble detection timeout (see *Sub-register 0x06:04 – Preamble detection* timeout count) are off, whilst SFD detection timeout (see *Sub-register 0x06:02 – SFD detection* timout count) is on.

Other SYS_CFG register settings such as automatic receiver re-enable (RXAUTR) and MAC functions such as frame filtering (FFEN), double buffering (DIS_DRXB) and automatic acknowledgment (AUTO_ACK) are all off by default. Automatic CRC generation in the MAC frame data is on (DIS_FCS_TX).

External synchronization and the use of external power amplifiers are deactivated by default, see sections *7.1 – External Synchronisation* and *7.2 – External power amplification*.

2.6.2 Default channel configuration

Channel 5, preamble code 9, and 64 MHz PRF are set by default in the CHAN_CTRL register, see *Sub-register 0x01:08 – Channel control* for more information.

The transmit data rate is set to 6.8 Mb/s in the TX_FCTRL register, see TXBR field in *Sub-register 0x00:20 – Transmit frame control*. The receive data rate is never set it can be decoded from the PHR bits.

2.6.3 Default transmitter configuration

Transmit RF channel configurations are set for channel 5 by default – see Sub-register 0x07:1C – RF TX control register 2. The transmit preamble symbol repetition length is 64 symbols, see *Sub-register 0x00:20 – Transmit frame control,* TXPSR field for configuration details.

2.6.4 Default receiver configuration

Receiver RF channel configurations are set for channel 5 by default, to match the transmitter.

Digital receiver tuning registers are configured by default for 64 MHz PRF, 6.8 Mb/s data rate, and a preamble symbol repetition of length 64. See *Sub-register 0x06:00 – Digital* RX tuning register for programming details.

The CIARUNE bit (CIA run enable) is enabled by default, which means that the CIA algorithm will execute on every packet reception, which in turn will calculate the accurate time of arrival. If the running CIA is not required then the CIARUNE control in *Sub-register 0x11:08 – Sequencing control* can be turned off (set to zero). This may be useful in a data communications only application, to save power and time.

2.7 UWB channels and preamble codes

The IEEE802.15.4 standard [1] has 16 defined channels/bands for the HRP UWB PHY. The QM33100 supports the subset of these listed in Table 8 below. Depending on the channel and the pulse repetition frequency (PRF) the IEEE802.15.4 standard [1] HRP UWB PHY defines a choice of two or four preamble codes. The standard defined preamble code options are also listed in Table 8. The combination of channel number and preamble code gives what the standard terms a *complex channel*.

Table 8: QM33100 supported UWB channels and recommended preamble codes

Channel number	Center frequency (MHz)	Bandwidth (MHz)	Preamble Codes (16 MHz PRF)	Preamble Codes (64 MHz PRF)
5	6489.6	499.2	3, 4	9, 10, 11, 12
9	7987.2	499.2	3, 4	9, 10, 11, 12

The preamble codes specified by the standard for use on a particular channel were chosen to have a low cross correlation factor with each other with the intention that the complex channels could operate independently from each other as separate networks. In practice, as there is still some cross correlation, there will be some break-through between different codes especially in conditions of close proximity with long preambles.

The IEEE802.15.4 standard [1] includes a feature called dynamic preamble select (DPS), where devices switch to using one of the DPS specific preamble codes for the ranging exchange, and perhaps a different one for each direction of communication. The idea is to make it more difficult to eavesdrop or spoof, by randomly changing the DPS preamble codes in a mutually agreed sequence only known to the valid participants. This is supported by the QM33100 where at 64 MHz PRF the preamble codes additionally available for DPS are: 13, 14, 15, 16, 21, 22, 23 and 24.

2.8 Data modulation scheme

The UWB specified in the IEEE802.15.4 standard [1] is sometimes called impulse radio UWB because it is based on high speed pulses of RF energy. During the PHR and Data parts of the packet, information bits are signalled by the position of the burst, in a modulation scheme termed burst position modulation (BPM).

Each data bit passes through a convolution encoder to generate a “parity” bit used to set the phase of the burst as either positive or negative, this component of the modulation is termed binary phase-shift keying (BPSK). Figure 10 shows how the convolutional encoder contributes to this BPM/BPSK modulation. A coherent receiver (i.e. one tracking carrier timing and phase) such as the one in the QM33100 can determine this burst phase and use it in a Viterbi decoder to get an additional 3 dB of coding gain, thereby extending the operational range of the modulation.

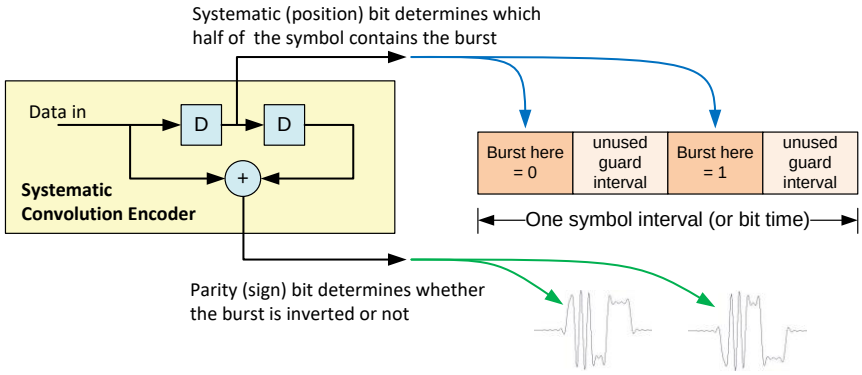

Figure 10: BPM/BPSK data and PHR modulation

In addition the quarter symbol interval is sub-divided into 2, 4, or 8 sub-intervals and a pseudo random sequence used to determine both the burst shape and which of the sub-intervals are actually used for the burst transmission. This gives more immunity to interference and whitens the output spectrum allowing a higher signal power to be used in the transmitter.

Forward error correction (FEC) is also included in the PHR and Data parts of the packet. The 19-bit PHR includes a 6-bit single-error-correct double-error-detect (SECDED) code and the data part of the packet has a Reed Solomon (RS) code applied. Both SECDED and RS codes are termed "systematic" meaning that the data can be recovered without using the codes (and of course not benefitting from them in that case), e.g. by a non-coherent receiver. The 850 kb/s and 6.8 Mb/s user data rates quoted, include allowance for the 0.87 average RS coding rate. The PHR is not RS coded so for example at the 850 kb/s nominal rate the PHR is actually sent at 975 kb/s.

2.9 Synchronisation header modulation scheme

The Synchronisation Header (SHR) consists of the preamble sequence and the SFD (start of frame delimiter). In contrast to the BPM/BPSK modulation used for the PHR and data, the synchronisation header is made up of single pulses. The preamble symbol period is divided into approximately 500 "chip" time intervals, (496/508 depending on 16/64 MHz PRF[1]), in which either a negative or a positive pulse may be sent, or no pulse. The "chip" interval is 499.2 MHz, a fundamental frequency within the UWB PHY, and so the resultant symbol times are thus $^{496}/_{499.2}$ µs for 16 MHz PRF, and $^{508}/_{499.2}$ µs for 64 MHz PRF, (see Table 9 below).

The sequence of pulses sent during the preamble symbol interval is determined by the preamble code. The standard defines 8 preamble codes of length-31 for use at 16 MHz PRF and 16 preamble codes of length-127 for use at 64 MHz PRF. The standard nominates particular codes for particular channels so that at 16 MHz PRF there are just two to choose from per channel, while at 64 MHz PRF there is a choice of four codes per channel. The length-31 codes are spread by inserting 15 zeros after each pulse to give the 496 chip times per symbol while the length-127 codes are spread by inserting 3 zeros after each pulse to give the 508 chip times per symbol. The preamble length and duration is defined by how many times (i.e. for how many symbols) the sequence is repeated. This is determined by the configuration of the number of preamble symbol repetitions (PSR).

[1]The QM33100 supports average pulse repetition frequencies of 16 MHz and 64 MHz

Table 9: Preamble parameters

Mean PRF (MHz)	#Chips Per Symbol	Preamble Symbol Duration (ns)
16 nominal	496	993.59
64 nominal	508	1017.63

The standard defines PSR settings of 16, 64, 1024 and 4096. The QM33100 supports these (although it will not receive packets with preamble length below 32 symbols) and in addition supports PSR settings of 128, 256, 512, 1536 and 2048.

The preamble sequence has a property of perfect periodic autocorrelation[2] which in essence allows a coherent receiver to determine the exact impulse response of the RF channel between transmitter and receiver. This brings two important benefits. Firstly, it allows the receiver make use of the received energy from multiple paths, turning multipath from an interference source into a positive affect extending operating range. Secondly, it lets the receiver resolve the channel in detail and determine the arrival time of the first (most direct) path, even when attenuated, which brings precision advantages for RTLS applications.

Note: The QM33100 includes a packet format specified by new IEEE802.15.4z amendment [2] which incorporates a cryptographically generated scrambled timestamp sequence (STS) that can be used to obtain an RX timestamp that has improved integrity in terms of its robust to accidental or deliberate interference, e.g. as a result of packet collisions, for more details of this please refer to section 6 below.
The SFD marks the end of the preamble and the precise start of the switch into the BPM/BPSK modulation of the PHR. The time-stamping of this event is deterministic in terms of symbol times and it is this in conjunction with determining the first arriving ray within that symbol time that allows the accurate time-stamping needed for precision RTLS applications. The standard specifies the SFD, which consists of the preamble symbols either not sent, or sent as normal or sent inverted (i.e. positive and negative pulses reversed) in a defined pattern 8 symbol times long for 850 kb/s and 6.81 Mb/s data rates. The length-8 SFD sequence is: 0, +1, 0, -1, +1, 0, 0, -1. The IEEE802.15.4z amendment also specifies length-8 SDF without zeros, (-1, -1, -1, +1, -1, -1, +1, -1), which gives improved performance in a coherent receiver such as the QM33100 that supports it.

2.10 PHY header: standard data frame length

The PHY header (PHR) is modulated using the BPM/BPSK modulation scheme defined in section 2.8 above, but it does not employ the Reed Solomon code used for data, instead is employs a 6-bit SECDED parity check sequence as part of its 19-bit length.

[2] V. P. Ipatov, "Ternary sequences with ideal autocorrelation properties," Radio Eng. Electron. Phys., vol. 24, pp. 75–79, 1979

Bit 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
R1	R0	L6	L5	L4	L3	L2	L1	L0	RNG	EXT	P1	P0	C5	C4	C3	C2	C1	C0
Data Rate		Frame Length							Ranging frame	Header Extension	Preamble Duration		SECDED Check Bits					

Figure 11: PHR bit assignment

Figure 11 above shows the bits of the PHR. These are transmitted bit-0 first in time. The QM33100 fills in the Data Rate, Frame Length, Ranging frame, and Preamble Duration fields of the PHR based on the user configuration of the appropriate parameters in TX_FCTRL and generates the SECDED sequence accordingly. The Header Extension bit of the PHR is always zero and is reserved by IEEE for future extensions.

2.11 Extended PHY header: extended data frame length

Standard IEEE 802.15.4-2020 UWB packets can carry up to 127 bytes of payload, see Figure 11: PHR bit assignment. The QM33100 also supports a mode of operation with frame lengths up to 1023 bytes. This mode of operation is enabled via the PHR_MODE selection bits of *Sub-register 0x00:10 – System configuration*, which changes the defifinition of the PHR bits. While this makes the PHR non-compliant with IEEE 802.15.4 this PHR format is actually defined as an option in the IEEE 802.15.8 standard.

In this mode the PHY header (PHR) is redefined to carry the 3 extra bits of frame length. In order to communicate extended length frames between two QM33100 devices both ends must be set to the long frame PHY header mode via the PHR_MODE selection bits of *Sub-register 0x00:10 – System configuration*. If the setting is only at one end of a link any attempt at communication will fail with PHR errors being reported. When this long frame mode is selected, the QM33100 will be unable to successfully communicate with any device operating with the IEEE 802.15.4 standard frame encoding, and because the SECDED error check sequence of the PHR in this long frame mode is inverted this will cause PHR error events in advance of any attempt to receive the payload.

Note that the probability of an error occurring within a frame increases as the frame length is increased, and as a result of this increasing the frame length may or may not improve system throughput depending on the error rate and the need to retransmit frames when there is an error.

In long frame mode only the high order bit of the TXPSR value from TX_FCTRL is sent in the PHR and reported in the RXPSR value in RX_FINFO.

The PHR encoding for the extended length frames is shown below in Figure 15:

Bit 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	14	16	17	18
R1	R0	L9	L8	L7	L6	L5	L4	L3	L2	L1	L0	P0	S5	S4	S3	S2	S1	S0
Data Rate		Frame Length										Preamble Duration	SECDED Check Bits					

Figure 12: PHR bit assignment extended length frames

The Data Rate field has the same encoding as used for the IEEE802.15.4 standard PHR.

The frame length field L9-L0 is an unsigned 10-bit integer number that indicates the number of octets in the PSDU from the MAC sub-layer. Note that the high order bit of the length is transmitted first in time.

A single bit, P0, provides the Preamble Duration field, indicating the length of the SYNC portion of the SHR shown in Table 11.

Table 10: Preamble duration field values in extended length frame PHR

P0	Preamble length for BPM-BPSK modulation mode
0	64 to 1024 symbols
1	1536 to 4096 symbols

The preamble duration field, TXPSR, may be used by a receiver to set the value of the preamble duration for an ACK frame to 64, 128, 256, 512, 1024, 1536, 2048 and 4096 symbols. Alternatively, the FINE_PLEN field may be used to set any multiple of 8 as a preamble length from 32 to 2048. The application may use the count of received preamble symbols, as reported in IP_DIAG12 register, to additionally inform the choice of preamble length for any response frames.

The SECDED field, S5–S0, is a set of six parity check bits that are used to protect the PHR from errors caused by noise and channel impairments. The SECDED calculation is the same as that defined in IEEE802.15.4 standard [1] except the bits C5–C0 are inverted to get S5–S0 as follows:

S0 = *NOT* (C0), S1 = *NOT* (C1), S2 = *NOT (*C2), S3 = *NOT* (C3), S4 = *NOT* (C4) and S5 = *NOT* (C5)

This is as specified in the IEEE 802.15.8 standard for frames up to 1023 octets.

3 Message transmission

3.1 Basic transmission

The transmission of packets is one of the basic functions of the QM33100 transceiver. The packets can be sent both with and without data payload. The QM33100 supports four packet formats, the IEEE802.15.4 standard [1] packet format, and three new IEEE802.15.4z amendment [2] defined formats where a scrambled timestamp sequence (STS) is introduced into the packet as shown in Figure 13. When the STS modes are enabled the QM33100 transmitter will insert the STS in the position shown, and the receiver will expect it to be present accordingly. The packet begins with a synchronisation header consisting of the preamble and the SFD (start of frame delimiter). The PHY header (PHR) defines the length (and data rate) of the data payload part of the packet. The STS when inserted allows for secure timestamping, and ranging, see § 6.

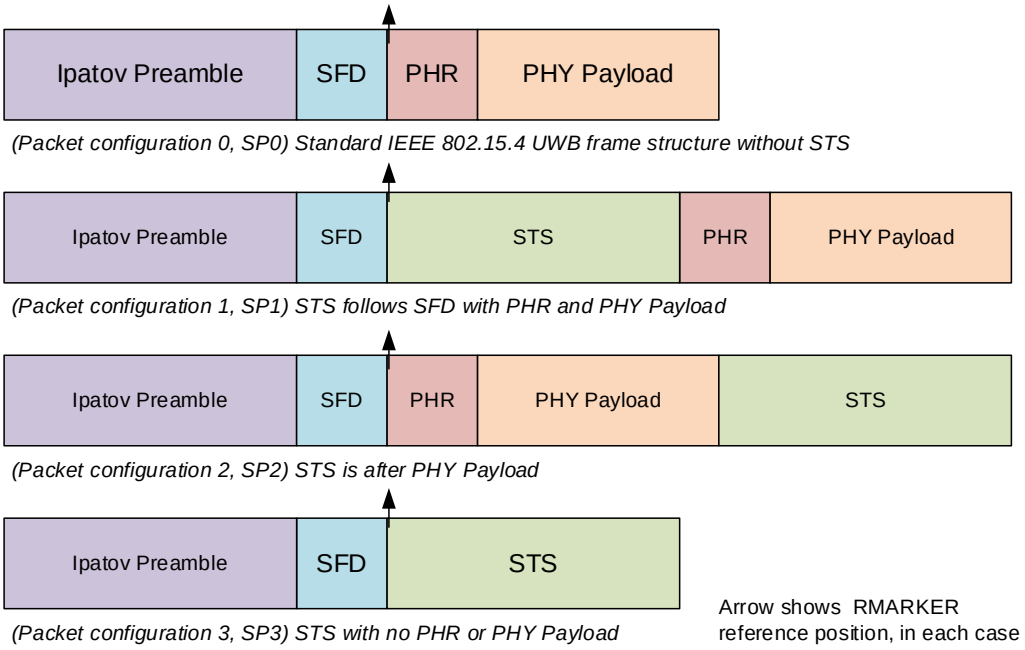

Figure 13: Packet formats

The QM33100 basic transmit sequence is as shown in Figure 14, beginning in the **IDLE_PLL** state awaiting instruction from the host controller.

In order to transmit, the host controller must write data for transmission to TX_BUFFER, and select the preamble length, frame length, data rate and PRF in the Transmit Frame Control and Channel Control registers, TX_FCTRL and CHAN_CTRL. The PRF will be set according to the chosen preamble code (TX_PCODE). Assuming all other relevant configurations have already been made, the host controller initiates the transmission by issuing a TX command (e.g. CMD_TX). Once transmission is initiated, the QM33100 sends the complete packet of preamble, SFD, PHR and the PHY Payload (MAC Frame). The STS will be included optionally depending on the configured packet format (see Figure 13). In the STS packet configuration 3 (SP3) the PHR and PHY Payload is omitted.

As an aid to MAC layer framing, the IC calculates the FCS (CRC) during the transmission of the (MAC) data from the TX_BUFFER based on the frame length (TXFLEN) and automatically appends it.

The end of transmission is signalled to the host via the TXFRS event status bit in SYS_STATUS, and the QM33100 returns to **IDLE_PLL** mode to await new instructions.

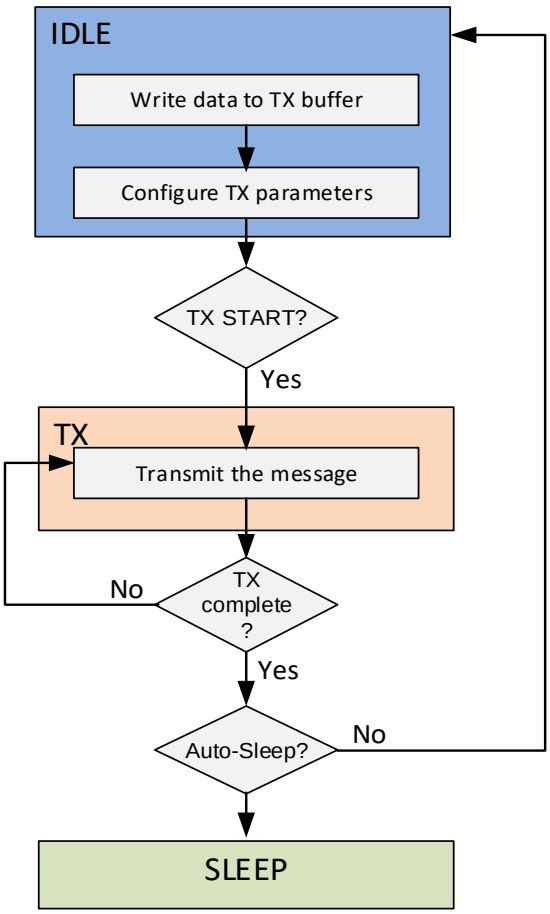

Figure 14: Basic transmit sequence

Further transmission features are described in the following sections:

- Transmit message time-stamping – see section *3.2 – Transmission timestamp*.
- Delayed transmission – see section *3.3 – Delayed transmission*.
- Long transmit frames – see section **3.4 – Extended length data frames**.

3.2 Transmission timestamp

During packet transmission the event nominated to time-stamp termed the RMARKER , where the **RMARKER** is defined in the IEEE standard to be the time when the beginning of the first symbol following the SFD is at the local antenna, or more precisely the peak pulse location associated with the first chip following the SFD.

The QM33100 digital transmit circuitry takes note of the system clock counter as the RAW transmit timestamp at the point when it begins sending the first chip following the SFD. It then adds to this the transmit antenna delay (configured in TX_ANTD) to get the antenna adjusted transmit time-stamp that it writes to the TX_STAMP field of TX_TIME.

See also section *10.3 – IC calibration – antenna delay.*

3.3 Delayed transmission

For delayed transmission, the transmit time is programmed into DX_TIME and then the delayed transmission is initiated by issuing the CMD_DTX command. Alternatively a delayed transmission can be achieved by programming a "reference time" into the DREF_TIME and an offset into the DX_TIME, after which delayed transmission command, CMD_DTX_REF, is issued instead.

One of the design goals of delayed transmission was that the specified transmission time would be predictable and aligned with the transmit timestamp. This was achieved in that the transmission time specified is the time of transmission of the **RMARKER** (not including the TX antenna delay), that is the raw TX time, TX_RAWST before the antenna delay is added. This allows for the time of transmission of a message to be pre-calculated and embedded in the message being transmitted.

Note: The value programmed into DX_TIME (and in DREF_TIME) is in time units of 4 ns, or more precisely $2 \div (499.2 \times 10^6)$. To calculate the time of transmission of the RMARKER at the antenna, the low 9 bits of the delayed TX time should be zeroed before adding the TX antenna delay, and the high 32-bits of the 40-bit result written into DX_TIME, (or a reference time programmed into DREF_TIME and an offset into the DX_TIME). Note: the least significant bit of DX_TIME is ignored so its effective resolution of is really $4 \div (499.2 \times 10^6)$ or approximately 8 ns.

In performing a delayed transmission, i.e., after the CMD_DTX is issued, the QM33100 calculates an *internal start time* for when to begin sending the preamble to make the RMARKER raw timestamp agree with the programmed transmit time. The QM33100 then remains in **TX_WAIT** state until the system time (SYS_TIME) reaches the correct point to turn on the transmitter and begin preamble.

One use of delayed transmission (and reception), is to control the response times in two-way ranging, (described in *APPENDIX 1: Two-way* ranging), and especially to allow the prediction and embedding of the TX timestamp (or response delay) into the transmitted message to reduce the number of messages necessary for a ToF measurement. Delayed transmission (and reception) also helps to minimise the response times to save power, however in working towards this the host microprocessor may sometimes be late invoking the delayed TX, i.e., where the system clock has passed the specified start time (i.e. *internal start time* mentioned above) and then the IC would have to complete almost a whole clock count period before the start time is reached. The HPDWARN event status flag in SYS_STATUS warns of this "lateness" condition so that during application development the delay may be chosen large enough to generally avoid this lateness. The HPDWARN status flag also serves to facilitate detection of this late invocation condition so that recovery measures may be taken should it ever occur in deployed product. For delayed transmission it is the *internal start time* mentioned above that is used when deciding whether to set the HPDWARN event for the delayed transmit. As long as the preamble start time is in the near future, the HPDWARN event flag will not be set. If a long delay was intended then HPDWARN flag can be ignored and the transmission will begin at the allotted time. If a long delay was not intended then the transmission can be stopped by issuing transceiver off command CMD_TXRXOFF.

3.4 Extended length data frames

Standard IEEE 802.15.4-2015 UWB packets can carry up to 127 bytes of payload. The QM33100 also supports a mode of operation with frame lengths up to 1023 bytes. This mode of operation is enabled via the PHR_MODE selection bits of *Sub-register 0x00:10 – System configuration*, which changes the defifinition of the PHR bits. While this makes the PHR non-compliant with IEEE 802.15.4 this PHR format is actually defined as an option in the IEEE 802.15.8 standard.

In this mode the PHY header (PHR) is redefined to carry the 3 extra bits of frame length. In order to communicate extended length frames between two QM33100 devices both ends must be set to the long frame PHY header mode via the PHR_MODE selection bits of *Sub-register 0x00:10 – System configuration*. If the setting is only at one end of a link any attempt at communication will fail with PHR errors being reported. When this long frame mode is selected, the QM33100 will be unable to successfully communicate with any device operating with the IEEE 802.15.4 standard frame encoding, and because the SECDED error check sequence of the PHR in this long frame mode is inverted this will cause PHR error events in advance of any attempt to receive the data payload.

Note that the probability of an error occurring within a frame increases as the frame length is increased, and as a result of this increasing the frame length may or may not improve system throughput depending on the error rate and the need to retransmit frames when there is an error.

In long frame mode only the high order bit of the TXPSR value from TX_FCTRL is sent in the PHR and reported in the RXPSR value in RX_FINFO.

The PHR encoding for the extended length data frames is shown below in Figure 15:

Bit 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	14	16	17	18
R1	R0	L9	L8	L7	L6	L5	L4	L3	L2	L1	L0	P0	S5	S4	S3	S2	S1	S0
Data Rate		Frame Length										Preamble Duration	SECDED Check Bits					

Figure 15: PHR encoding extended length frames

The Data Rate field has the same encoding as used for the IEEE802.15.4 standard [1] PHR.

The frame length field L9-L0 is an unsigned 10-bit integer number that indicates the number of octets in the PSDU from the MAC sub-layer. Note that the high order bit of the length is transmitted first in time.

A single bit, P0, provides the Preamble Duration field, indicating the length of the SYNC portion of the SHR shown in Table 11.

Table 11: Preamble duration field values in extended length frame PHR

P0	Preamble length for BPM-BPSK modulation mode
0	64 to 1024 symbols
1	1536 to 4096 symbols

The preamble duration field, TXPSR, may be used by a receiver to set the value of the preamble duration for an ACK frame to 64, 128, 256, 512, 1024, 1536, 2048 and 4096 symbols. Alternatively, the FINE_PLEN field may be used to set any multiple of 8 as a preamble length from 32 to 2048. The application may use the count of received preamble symbols, as reported in IP_DIAG12 register, to additionally inform the choice of preamble length for any response frames.

The SECDED field, S5–S0, is a set of six parity check bits that are used to protect the PHR from errors caused by noise and channel impairments. The SECDED calculation is the same as that defined in IEEE802.15.4 standard [1] except the bits C5–C0 are inverted to get S5–S0 as follows:

S0 = *NOT* (C0), S1 = *NOT* (C1), S2 = *NOT (*C2), S3 = *NOT* (C3), S4 = *NOT* (C4) and S5 = *NOT* (C5)

This is as specified in the IEEE 802.15.8 standard for frames up to 1023 octets.

4 Message reception

4.1 PHY reception

The reception of a packet is enabled by a host request enabling of the receiver. This can be done while the device is in either IDLE_RC or **IDLE_PLL** state. If the device was in IDLE_RC it will firstly clalibrate, enable PLL and switch to **IDLE_PLL** state and then goin into **RX** state. However, prior to the first RX enable following device power up, the receiver needs to be calibrated. This is done by performing RX calibration, for more details on this see the dwt_pgf_cal() API [3] and RX_CAL register. The RX calibration is automatically done on wakeup as long as ONW_PGFCAL bit is set. It is also recommended that the receiver is re-calibrated if the operating temperature changes by 20 °C.

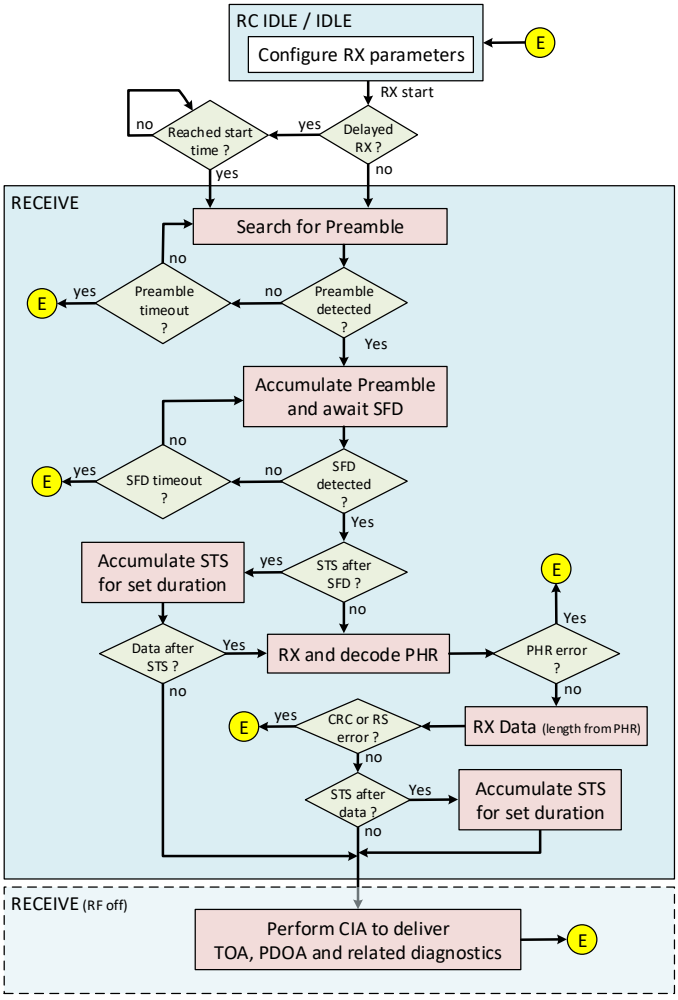

Figure 16: Receive sequence

To enable the receiver, the host issues an RX start command (see section on Fast Commands). The receiver will start by searching for preamble continually until preamble has been detected or acquired, then a demodulation will be attempted. A preamble detection timeout may be set to allow the receiver to stop searching for preamble after a desired period (which is programmable in PRE_TOC). A receive sequence is shown in Figure 16.

4.1.1 Preamble detection

The preamble sequence is detected by cross-correlating in preamble acquisition chunks (PACs) which are a number of preamble symbols long. The size of chunk used is selected by the PAC configuration in DTUNE0. The PAC size should be selected depending on the expected preamble size. A larger PAC size gives better performance when the preamble is long enough to allow it, but if the PAC is too large for the preamble length the receiver performance will be impaired, or fail to work at the extremes – (e.g. a PAC of 32 will never receive packets with just 32 preamble symbols). Table 12 gives the recommended PAC size configuration to use in the receiver depending on the preamble length being used in the transmitter.

Table 12: Recommended PAC size

Expected preamble length of packets being received	Data Rate	Recommended PAC size
32	6.81 Mb/s	4
≥ 64	6.81 Mb/s	8
≥ 128	850 kb/s	16

Aborting reception with use of preamble detection timeout (PRE_TOC) is very useful following sending a message where a response is being awaited. Here if the preamble is not detected then the awaited response is not coming. The preamble detection time-out can be used to abandon the reception at the earliest possible time, saving power.

4.1.2 Preamble accumulation

Once the preamble sequence is detected, the receiver begins accumulating correlated preamble symbols and building a channel impulse response (CIR), while in parallel looking for the SFD sequence (a particular sequence of preamble symbols, see § *2.9 – Synchronisation header modulation scheme* for details).

4.1.3 SFD detection

The detection of SFD is a key event in the reception of a packet, because it marks the RMARKER, which is time-stamped (see section *4.1.7 – RX message timestamp*), and it marks the change from preamble demodulation to the BPM/BPSK demodulation of the PHR and data or STS demodulation if enabled.

It is possible to abort reception if the SFD is not detected within a certain time after preamble is detected. This functionality is configured via DRX_SFDTOC. This SFD detection timeout guards against false detection of preamble (which has a finite chance of happening) that could otherwise lead to a prolonged period receiving nothing. By default, the SFD detection timeout is 65 symbol times (to match the default preamble length of 64 symbols and a PAC size of 8). It is not recommended to disable the SFD detection timeout.

The SFD sequence is either 8 or 16 symbols long for the supported data rates of 6.8 Mb/s or 850 kb/s. The type of the SFD sequence used is defined by SFD_TYPE configuration bits in CHAN_CTRL register, and are described in Table 23.

4.1.4 STS reception

If the STS is enabled, see Figure 13, the receiver will construct another CIR from that. The time-of-arrival (ToA) can also be derived from this CIR estimate.

The STS itself consists of a series of equally spaced pulses whose polarity is derived from an AES128 based cryptographically secure pseudo random number generator as specified in the IEEE802.15.4z amendment.

This can help when constructing a secure ranging environment (see 6 Secure ranging / timestamping) and it also improves immunity to packet collisions on the air.

4.1.5 PHR demodulation

The main role of the PHY Header (PHR) is to convey the length of the data portion of the packet, and to indicate the data rate being employed for data demodulation. See paragraph 2.10 - PHY header and paragraph 2.11 for details of the PHY header. For data rates of 850 kb/s and 6.8 Mb/s the PHR is modulated / demodulated as per the 850 kb/s data rate (note that because Reed Solomon encoding is not applied to the PHR, its bit rate is approximately 1 Mb/s). If the PHR is indicating 850 kb/s then the data demodulation continues at this rate, but if the PHR is indicating 6.8 Mb/s then the demodulation changes to this rate at the end of the PHR as data demodulation begins.

It is also possible to configure the PHR rate to be the same as the data rate, i.e. to use 6.8 Mbit/s with PHR_6M8 configuration bit in SYS_CFG, 8.2.2.4.

4.1.6 Data demodulation

Section *2.8 – Data modulation scheme* describes the modulation scheme. In the receiver a Viterbi decoder is used to recover the data bits (this is also used for PHR reception) which are then passed through the Reed Solomon decoder to apply, if it can, any further corrections. Every octet thus received is passed through a CRC checker that checks the frame against the transmitted FCS.

As the data octets are received they may also be parsed by the frame filtering function if enabled, see section *5.4– Frame filtering* for more details.

Successful reception of a frame is signalled to the host via the RXFR and RXFCG event status bits in SYS_STATUS. Other status bits in this register may be used to flag reception of other parts of the frame or, events indicating failure, i.e. RXPTO (Preamble detection Timeout), RXSTO (SFD timeout), RXPHE (PHY Header Error), RXFSL (Reed Solomon error), RXFTO (Frame wait timeout), etc.

Features related to reception	Section
Receive message time-stamping	*4.1.7 – RX message timestamp.*
Delayed reception	*4.3 - Delayed receive*
Receiving long frames	*3.4–Extended length data frames*

4.1.7 RX message timestamp

The IEEE802.15.4 standard [1] nominates the time when the RMARKER arrives at the antenna as the significant event that is time-stamped, also shown in Figure 13.

The QM33100 digital receiver circuitry takes a coarse timestamp of the symbol in which the RMARKER event occurs and adds in various correction factors to give a resultant adjusted time stamp value, which is the time at which the RMARKER arrived at the antenna. This includes subtracting the receive antenna delay as configured in the RXANTD register (in CIA_CONF) and adding the correction factor determined by the first path (leading edge) detection algorithm embedded in the QM33100. The resulting fully adjusted RX timestamp is written into RX_STAMP, IP_TOA and STS_TOA registers.

See also section *10.3 – IC calibration – antenna delay*, and section *6 – Secure ranging / timestamping*.

4.2 TDoA and PDoA support

The two-antenna port variants of QM33100 chip can measure the phase of incoming signal. This information can be used to help determine the direction of arrival and location of the transmission when combined with knowledge of the antenna response. More details of how this is performed will be given in an associated application note. This section will focus on the information provided by the device itself.

Depending on the variant of the device there will be one or two antenna ports. Devices with two antenna ports are referred to as PDoA parts while the others are non-PDoA parts (see Table 1). The two variants have different capabilities when it comes to Phase Difference of Arrival (PDoA) support.

For the “PDoA” parts, if the packet contains an STS then, depending on the configured mode, the device can compute the PDoA. There are two methods for computing the PDoA: PDoA Mode 1 which is the fira compliant mode [7]Fira Consortium - https://www.firaconsortium.org/Fira Consortium - https://www.firaconsortium.org/Fira Consortium - https://www.firaconsortium.org/ and PDoA Mode 3.

PDoA Mode 1 functions with any packet containing an STS but is less accurate than PDoA Mode 3. PDoA Mode 3 requires the STS length (in units of 512 chips, ~1 µs) to be an integer multiple of 128 (128, 256, 512). It is more accurate than Mode 1 but the packets will be longer (see PDOA_MODE in SYS_CFG register on how to configure the required mode).

In either of the PDoA modes the chip will also compute the Time Difference Of Arrival (TDoA) between the two channel estimates that are used to determine the PDoA. If this TDoA is large, e.g. greater than 1 ns, then the PDoA should be considered to be untrustworthy. This is probably due to harsh channel conditions resulting in an ambiguous first ray estimate.

4.3 Delayed receive

In delayed receive operation the receiver turn-on time is programmed into DX_TIME and then the delayed receiving is initiated by issuing CMD_DRX command. . Otherwise a delayed reception can be achieved by programming a “reference time” into the DREF_TIME and an offset into the DX_TIME, after which delayed reception command, CMD_DRX_REF, is issued instead.

The QM33100 remains in **IDLE_PLL** state until the system time (SYS_TIME) reaches the value programmed in DX_TIME (or time programmed into DREF_TIME plus an offset into the DX_TIME) and then the IC receiver is turned on. This point marks the start time for any programmed timeouts that apply to the reception process, i.e. the preamble detection timeout (which is set and enabled by PRE_TOC) and the frame wait timeout (which is enabled by the RXWTOE configuration bit in *Sub-register 0x00:10 – System configuration*, and whose period is programmed in RX_FWTO).

The benefit of delayed receive is that the receiver can be turned on at just the right moment to receive an expected response, especially when that response is coming from a QM33100 employing delayed transmit to send the response message at a precise time. This saves power because the **IDLE_PLL** state power consumption while counting down to the time to perform the RX enable is significantly less than the **RX** state power consumption while waiting and searching for the preamble of a packet reception.

One use of delayed receive, and especially delayed transmission, is in two-way ranging, (described in *APPENDIX 1: Two-way* ranging), to control the response times. On the receive side turning on the receiver just in time to receive the response message helps save power since the IC will remain in **IDLE_PLL** state until it is time to turn on the receiver. Minimising the response time and time spent in **IDLE_PLL** state also saves power. It is possible for the host microprocessor to be late invoking the delayed TX or RX, so that the system clock is beyond the specified start time and then the IC would have to complete almost a whole clock count period before the specified start time is reached. The HPDWARN event status flag in SYS_STATUS warns of this "lateness" condition so that during development a delay may be chosen large enough to generally avoid this lateness. The HPDWARN status flag also serves to facilitate detection of this late invocation condition so that recovery measures may be taken should it ever occur in deployed product.

4.4 Double receive buffer

The QM33100 has a pair of receive buffers (RX_BUFFER_0 and RX_BUFFER_1) offering the capability to receive into one of the pair while the host system is reading previously received data from the other buffer of the pair. For example, this is useful in a TDoA RTLS anchor node application where it is desired to have the receiver turned on for as much time as possible to avoid missing any tag blink messages. A number of ancillary registers (timestamps, quality indicators and status bits) are also doubly-buffered. The registers that are part of this "RX double-buffered swinging-set" (SET_1 and SET_2) and are listed in Table 47. To enable logging of data to these diagnostic sets, RDB_DMODE must be set to at least 1 to log minimal set of registers.

Note that double buffering can also be used with the SP3 packets, configured via the CP_SPC field in *Sub-register 0x00:10 – System configuration*, where even though the frame contains no payload, the receive timestamp and other diagnostic information is doubly buffered.

4.4.1 Enabling double-buffered operation

By default the QM33100 operates in a single buffered mode that is appropriate for many applications. When using double-buffered mode it is normal to also configure the QM33100 to automatically re-enable the receiver (moving on to the other buffer of the swinging set) as soon as it has completed receiving any previous frame. Double-buffered receiving is enabled by setting the DIS_DRXB bit to zero, (in *Sub-register*

0x00:10 – System configuration). The RX auto-re-enable function is enabled by setting the RXAUTR bit to 1 (in *Sub-register 0x00:10 – System Configuration*).

The QM33100 may be operated in double buffered mode without the automatic re-enabling of the receiver, in which case the host needs to manually re-enable the receiver. The receiver can be enabled in advance of processing the previously received frame. This operation will reduce the amount of time for which the receiver may be actively listening for frames on the air, but will prevent both buffers being full (at the same time) and will prevent overflows. This simplifies the double-buffer operation, see sections 4.4.2 and 4.4.3.

4.4.2 Operation of double buffering

In non-double buffer mode the QM33100 will write the received frame data into RX_BUFFER_0. When the double buffer mode is enabled the QM33100 will alternate writing frames to RX_BUFFER_0 and RX_BUFFER_1. The associated RX_FINFO and diagnostic data is stored in two register sets: SET_1 and SET_2 as listed in Table 47. The RDB_STATUS register informs the host which register set is ready and available. For example, when RX_BUFFER_0 frame reception is complete the IC will set the RXFR0 with RXFCG0 in RDB_STATUS register interrupting the host and the IC will move on to receive into the other buffer of the double-buffered swinging-set. Once the host has finished accessing a register set is should issue toggle buffer command, CMD_DB_TOGGLE, to let the IC know that the particular buffer is “free” again and that it can be used for the next received frame. The host should also clear the RDB_STATUS register bits relating to the processed buffer. Figure 17 below is a flow chart showing the operation of double buffering in the receiver.

Reception of a frame with good CRC will set the relevant RDB_STATUS register bits. When RX auto-re-enable function is enabled, in the case that a received packet is rejected by frame filtering, bad CRC or another error, the RDB_STATUS register bits will remain unset and the current buffer will be reused for the next incoming packet. If RX auto-re-enable function is disabled the host will need to respond to the RX error event and re-enable the receiver.

4.4.3 Overrun

With RX auto-re-enable function enabled in the receiver an overrun condition may occur if the host side is not keeping up with the arrival rate of frames. For example, say the IC receives a frame into the first buffer, moves on to the other buffer and receives a second frame in it. The IC will then try and receive next frame into the first buffer again, but if the host has not completed reading the data from this first buffer (i.e. not yet issued the CMD_DB_TOGGLE command) then the IC will not write data to the buffer and will flag this as an overrun condition (RXOVRR event in SYS_STATUS). The overrun condition occurs at the point the receiver finishes processing a good PHR and needs to write the first octet of data to the RX buffer. This event is detected by the IC and reported in the RXOVRR status bit in *Sub-register 0x00:44 – System Event Status*.

When a receiver overrun occurs, the frame reception in progress will be aborted and, assuming RX auto-re-enable is enabled (by RXAUTR) the receiver will begin looking for preamble again. The overrun condition will be cleared as soon as the host issues the CMD_DB_TOGGLE command. The host should clear the RXOVRR status bit. Receiver overrun events are also counted in *Sub-register 0x0F:0E –* RX overrun error counter, assuming that counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event counter control*.

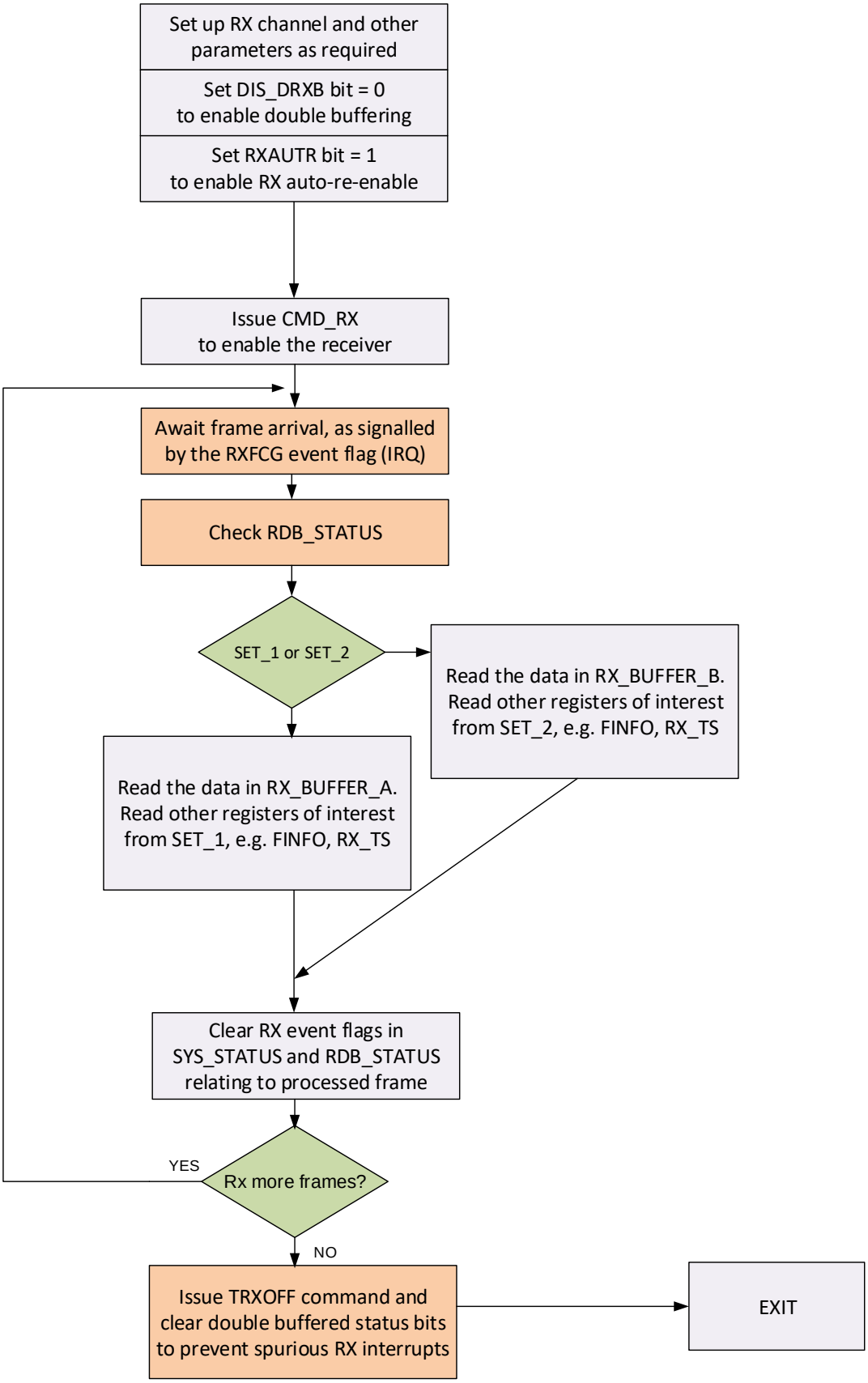

Figure 17: Flow chart for using double RX buffering

4.5 Low-Power SNIFF mode

Low-Power SNIFF mode is a lower power preamble hunt mode, also known as pulsed preamble detection mode (PPDM), where the receiver (RF and digital) is sequenced on and off rather than being on all the time. These on and off times are configurable in 8.2.15.6 Sub-register 0x11:1C – SNIFF mode and have default values of zero, disabling the feature. Using SNIFF mode causes a reduction in sensitivity depending on the ratio and durations of the on and off periods.

In SNIFF mode the QM33100 alternates between the **RX** state (on) and the **IDLE_PLL** (off) state Figure 18 shows a simplified view of the state transitions during SNIFF mode.

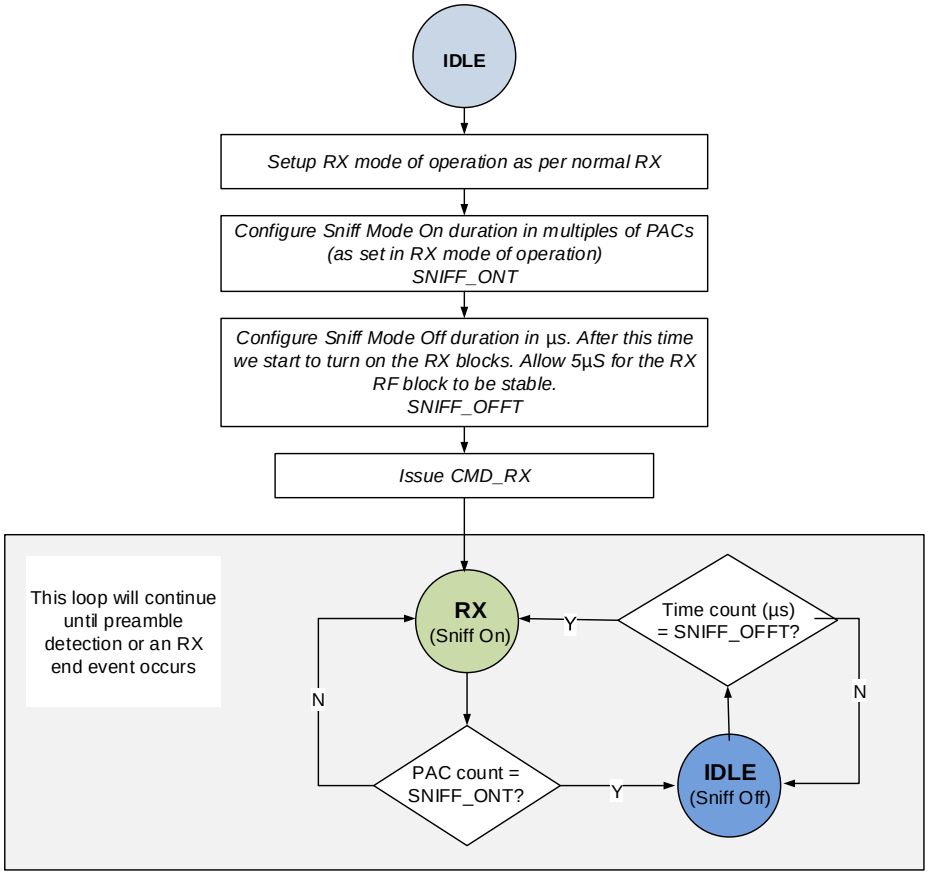

Figure 18: State transitions during SNIFF mode

4.5.1 SNIFF mode power profiles

In SNIFF mode the QM33100 alternates between the **RX** (on) and the **IDLE_PLL** (off) states. To enable SNIFF mode two parameters SNIFF_ON (sniff on time) and SNIFF_OFF (the off time) need to be configured in Sub-register 0x11:1C – SNIFF mode. The on duration is programmed in units of PAC, (these are described in section 4.1.1 - Preamble detection), and must be set to at a minimum value of 2 for functional preamble detection. The SNIFF_ON counter automatically adds 1 PAC unit to the total PAC count so the programmed value should always be 1 less than the desired total. The off duration is programmed in units of 1 µs. When

both on and off durations are programmed with non-zero values SNIFF will be operational from the next RX enable.

As an example if the PAC size is 8 symbols, (this is approximately 8 µs), and we want to have a 50:50 on-off duty cycle, then we could set SNIFF_ON to its minimum of 2 PAC intervals (by programming the counter with a value of 1) and the SNIFF_OFF to a value of 16 µs.

Figure 19 below shows the power profile associated with SNIFF mode where the IC wakes up from **SLEEP** and progress into the repeated **IDLE_PLL**-**RX**-**IDLE_PLL**-**RX**… duty-cycle of the pulsed preamble detection mode. A timeout ends this and the QM33100 is returned to **SLEEP**.

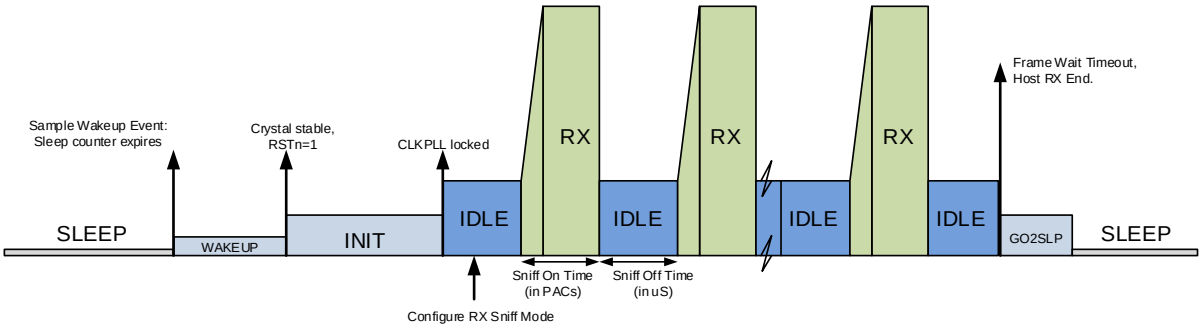

Figure 19 Power profile for SNIFF where a packet is not received

Figure 20 below, shows a power profile for SNIFF mode, similar to figure 19 except in this case preamble is detected on the second period of RX sampling, and the QM33100 completes the reception of the packet.

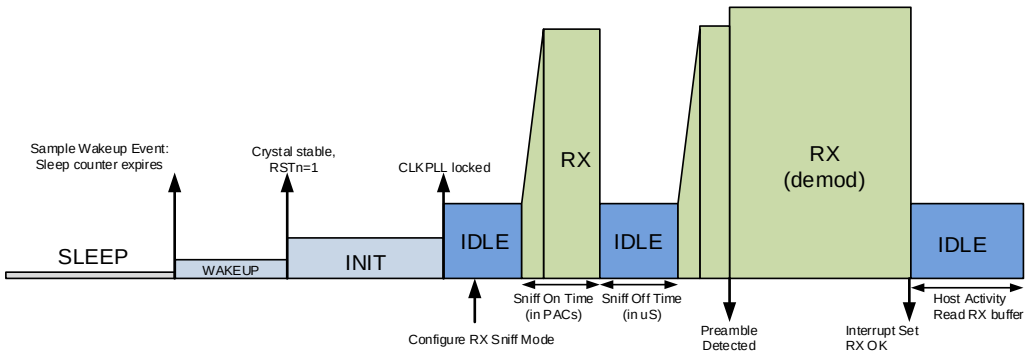

Figure 20 Power profile for SNIFF where a packet is received

4.6 Diagnostics

The QM33100 includes the following diagnostic aids: -

- The ability to drive LEDs to show TX and RX activity, which may be useful during product development and in non-battery powered devices. The LED driving feature is an option on GPIO lines, and is configurable via Register file: 0x05 – GPIO control and status. Please refer to the register description for details of the supported functionality.

Access to accumulator – of use during product development diagnostics. This is provided via:

- Register file: 0x15 – Accumulator CIR memory. Please refer to its description for details.

- RX packet quality indications – of use for both product development diagnostics and for working diagnostics, e.g. for network management or for deciding on confidence level for an RTLS or ranging measurement. These are available through the diagnostics registers of Register files: 0x0C, 0x0D, 0x0E – CIA Interface. Please refer to section *4.7 – Assessing the quality of reception and the RX timestamp* for details.

4.7 Assessing the quality of reception and the RX timestamp

The QM33100 receiver is capable of receiving messages under many different conditions. In some circumstances it can be useful to assess the quality of the received signals and any timestamp data based on them.

The following details the elements of receive status reported by the QM33100 that may be used to assess the quality of a received message and any related timestamp.

Note: These items use diagnostics provided by the channel impulse analyser (CIA) algorithm, which operates on the accumulated correlation of the repeated symbols preamble sequence and/or the accumulated correlation of the scrambled timestamp sequence (STS) used for secure timestamping, see *§ 6 – Secure ranging / timestamping*. The timestamps based on preamble and STS sequences can be assessed following these rules:

- Each ToA has an associated status word, see section 8.2.13. This word contains the results of several confidence tests on the first path estimate. If any of the bits in the status word is set, then the corresponding confidence test has failed and so the conditions may have resulted in lower confidence in the ToA.
- It is possible to compute an estimated receive power figure – for the purposes of this discussion this will be called RX_POWER (see section 4.7.2). It is also possible to compute an estimated power for just the first path signal – for the purposes of this discussion this will be called FP_POWER (see section 4.7.1). Using these two calculations it may be possible to say whether the channel is line-of-sight (LOS) or non-line-of-sight signal (NLOS).
- It is possible to compare the locations of the first path and the peak path in the CIR estimate. In LOS channels the peak path will be close to the first path.
- Where possible the system should use a scrambled time sequence (STS). This allows the QM33100 produce at least 2 estimates of the CIR and so compute 2 independent ToA estimates. As both are measuring the same physical parameter, they should be very close to each other. The CIA algorithm will compute the difference between these estimates and test it against a predefined threshold. If the difference is too great, then an error flag will be raised, and the ToA should not be trusted.

4.7.1 Estimating the signal power in the first path

An estimate of the power in the first path signal may be calculated (in dBm). Depending on the receiver configuration two different formulae are used to do this.

If the RX_TUNE_EN bit is set in DGC_CFG register, described in 8.2.4.1 (DGC):

$$First\ Path\ Power\ Level = 10 \times \log_{10}\left(\frac{{F_1}^2 + {F_2}^2 + {F_3}^2}{N^2}\right) + (6 \times D) - A\ \ dBm$$

If the RX_TUNE_EN bit is not set in DGC_CFG register, described in 8.2.4.1 (No DGC):

$$First\ Path\ Power\ Level = 10 \times \log_{10}\left(\frac{{F_1}^2 + {F_2}^2 + {F_3}^2}{N^2}\right) - A\ \ dBm$$

Where:

- F_1 = the *First Path Amplitude (point 1)* magnitude value (it has 2 fractional bits),
- F_2 = the *First Path Amplitude (point 2)* magnitude value (it has 2 fractional bits),
- F_3 = the *First Path Amplitude (point 3)* magnitude value (it has 2 fractional bits),
- N = the number of preamble symbols accumulated, or accumulated STS length
- D = the DGC_DECISION, treated as an unsigned integer in range 0 to 7
- A = is the constant:
  - 113.8 for a PRF of 16 MHz, or
  - 121.7 for a PRF of 64 MHz Ipatov preamble or
  - 120.7 for a PRF of 64 MHz STS.

The values for F1, F2, F3 and N can be read from the register file described in 8.2.13. F1, F2 and F3 can be read from IP_DIAG_2, IP_DIAG_3, IP_DIAG_4, or STS_DIAG_2, STS_DIAG_3, STS_DIAG_4, or STS1_DIAG_2, STS1_DIAG_3, STS1_DIAG_4, and N can be read from: IP_DIAG_12 , STS_DIAG_12 or STS1_DIAG_12, depending on the CIR used. D is read from DGC_DECISION described in DGC_DBG register, see section 8.2.4.2.

Note: These readings will not be available if the CIA is configured to compute minimal diagnostics, see MINDIAG bit in CIA_CONF register.

4.7.2 Estimating the receive signal power

It is possible to calculate an estimate of the receive power level (in dBm). Depending on the receiver configuration two different formulae are used to do this.

If the RX_TUNE_EN bit is set in DGC_CFG register, described in 8.2.4.1 (DGC):

$$RX\ Level = 10 \times \log_{10}\left(\frac{C \times 2^{17}}{N^2}\right) + (6 \times D) - A\ \ dBm$$

If the RX_TUNE_EN bit is not set in DGC_CFG register, described in 8.2.4.1 (no DGC):

$$RX\ Level = 10 \times \log_{10}\left(\frac{C \times 2^{17}}{N^2}\right) - A\ \ dBm$$

Where:

- C = the *Channel Impulse Response Power* value,
- N = the *Preamble Accumulation Count* value,
- D = the DGC_DECISION, treated as an unsigned integer in range 0 to 7
- A = is the constant:
  - 113.8 for a PRF of 16 MHz, or

- 121.7 for a PRF of 64 MHz Ipatov preamble or
- 120.7 for a PRF of 64 MHz STS.

The values for C and N can be read from the register file described in 8.2.13. C can be found in: IP_DIAG_1, STS_DIAG_1, or STS1_DIAG_1, and N can be read from: IP_DIAG_12 , STS_DIAG_12 or STS1_DIAG_12, depending on the CIR used. D is read from DGC_DECISION described in DGC_DBG register, see section 8.2.4.2.

Figure 21 shows the typical relationship between the actual receive power and the power estimated by this technique.

Note: Received signal power is sometimes referred by the term received signal strength indication (RSSI)

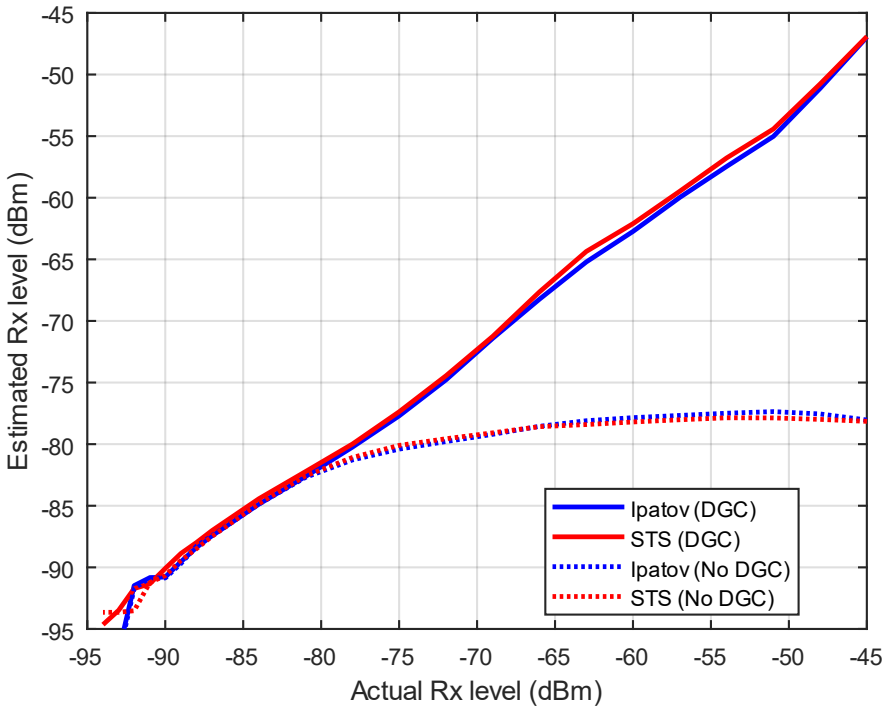

Figure 21: Estimated RX level versus actual RX level

5 Media Access Control (MAC) hardware features

The QM33100 has an incorporated MAC features on a transmitter and a receiver side to free the host processor from unnecessary interrupts. This helps to increase the overall performance of the system built with QM33100. The sections below describe the features for media access control (MAC) that have been implemented in the QM33100.

5.1 MAC level processing in the QM33100

The QM33100 transmits data from the TX_BUFFER in a frame with data length as specified in the TXFLEN field of the TX_FCTRL, inserting the 2-octet FCS as the last two octets of the data payload. The QM33100 will not do any other MAC level transmit processing. It is up to the host system software to prepare the correctly formatted frame conforming to the IEEE802.15.4 standard [1] MAC if this is required.

On the receive side, the QM33100 will validate the FCS of the received frame, and can parse frames complying with IEEE802.15.4 standard [1] to validate and accept only those as configured by the frame filtering configuration bits in the FF_CFG register (as described in section 5.4 below). The QM33100 can also optionally respond to the acknowledgement request bit set in the frame control field, of correctly addressed Data Frames or MAC Command frames, by sending an IEEE802.15.4 standard [1] acknowledgement frame (as described in section 5.5 below). Note: This only applies to the immediate acknowledgment (Imm-Ack) used to acknowledge Data frames or MAC command frames with the Frame Version field set to 0b00 or 0b01.

The QM33100 will deliver the received data frame in the RX_BUFFER_0 with its data length reported by the RXFLEN field of the RX_FINFO, and other than the RX activities mentioned in the paragraph above the QM33100 will not do any additional MAC level receive processing. It is up to the host system software to correctly parse the received frame according to the IEEE802.15.4 standard [1] MAC definition and take whatever additional action is prescribed by the standard, if this is required.

5.2 General MAC message format

The MAC message occupies the *PHY Payload* portion of the UWB packet as shown in Figure 13. This may be up to 127 octets in length according to the standard, and up to 1023 octets when employing the QM33100's long frame mode, see section **3.4** – **Extended length data frames**. The general structure of a MAC message consists of a header that identifies the frame, followed by a variable length (possibly zero) payload typically from the upper layers but sometimes (as in the case of MAC command frames) generated within the MAC itself, and finally ended by the MAC footer which is the FCS (Frame Checking Sequence) CRC used to detect transmission errors. Figure 22 shows the components of the MAC message frame in more detail, indicating the number of octets in each component.

The MAC header is parsed by the QM33100 as part of the frame filtering function to determine if the destination address matches the IC's address information programmed in *Sub-register 0x00:04 – Extended* Unique Identifier and *Sub-register 0x00:0C – PAN Identifier and Short Address* (or if the frame is a broadcast message). This parsing of receive frame is based on the contents of the Frame Control field (at the start of the MAC header).

PHY Payload								
MAC Header (MHR)							MAC Payload	MAC Footer (MFR)
Frame Control	Sequence Number	Destination PAN Identifier	Destination Address	Source PAN Identifier	Source Address	Aux Security Header	Frame Payload	FCS
2 octets	1 octet	0 or 2 octets	0, 2 or 8 octets	0 or 2 octets	0, 2 or 8 octets	0, 5, 6 10 or 14 octets	Variable number of octets	2 octets

Figure 22: General MAC message format

5.3 Cyclic redundancy check

The QM33100 includes a CRC generation function capable of automatically calculating and appending the 16-bit CRC frame check sequence (FCS) at the end of each transmitted frame.

The QM33100 also includes a CRC checking function capable of automatically calculating the 16-bit CRC frame check sequence (FCS) during frame reception and comparing this calculated CRC with the final two octets of the received frame to check that the calculated CRC matches with CRC transmitted by the frame's originator. A mismatch between received and calculated CRC typically indicates that the received frame contains errors (generally handled by discarding the received frame). At the end of the frame reception as reported via the RXFR event status bit, the result of the CRC comparison is reported by either RXFCG or the RXFCE status bit being set, i.e. depending on whether or not the CRCs matched. These three status bits are SYS_STATUS.

Where a CRC is not required it is possible to disable the CRC transmission by employing the setting DIS_FCS_TX (disable FCS transmission) bit in SYS_CFG. This might be done when using a different MAC layer protocol.

5.4 Frame filtering

Frame filtering is a feature of the QM33100 IC that can parse the received data of frame types defined in the IEEE802.15.4 standard [1], (and also listed in Table 13 below) identifying the frame type and its destination address fields, match these against the IC's own address information, and only accept frames that pass the filtering rules.

Table 13: Frame type field values

Frame Type Field (Frame Control bits 2 to 0)	Frame
0, 0, 0	Beacon
0, 0, 1	Data
0, 1, 0	Acknowledgement
0, 1, 1	MAC command
1, 0, 0	Reserved
1, 0, 1	Multipurpose

1, 1, 0	Fragment or Frak
1, 1, 1	Extended

The frame filtering functionality allows the IC to be placed into receive mode and only interrupt the host processor when a frame arrives that passes the frame filtering criteria. When frame filtering is disabled all frames with good CRC are accepted, typically to interrupt the host with event status indicating a frame has been received with good CRC (i.e. the RXFR and RXFCG event status bits are set in SYS_STATUS). When frame filtering is enabled the frame filtering rules have to be passed before these event status (interrupt) bits are set. See section 4.1 PHY reception for general details of reception.

Frame filtering is enabled by the FFEN configuration bit in SYS_CFG register. The frame filter is further configured by the various configuration bits in the FF_CFG register. This register contains ten additional configuration bits (FFAB, FFAD, FFAA, FFAM, FFAR, FFAMP, FFAF, FFAE, FFBC and FFIB) for fine filtering control of the frame types.

5.4.1 Frame filtering rules

If frame filtering is enabled frames will be accepted or rejected based on the following rules:

- The particular frame type must be allowed for reception:
  - The FFAB configuration bit must be set to allow a Beacon frame to be received.
  - The FFAD configuration bit must be set to allow a Data frame to be received.
  - The FFAA configuration bit must be set to allow an Acknowledgment frame to be received.
  - The FFAM configuration bit must be set to allow a MAC command frame to be received.
  - The FFAR configuration bit must be set to allow Reserved frames to be received, only a FCS check is performed on these.
  - The FFAMP configuration bit must be set to allow Multipurpose frames to be received.
  - The FFAF configuration bit must be set to allow Fragmented/Frak frames to be received, only a FCS check is performed on these.
  - The FFAE configuration bit must be set to allow Extended frames to be received, only a FCS check is performed on these
- The frame version field must be 0x00, 0x01 or 0x02
- The Destination PAN ID if present must:
  - Be the broadcast PAN ID (0xFFFF)
  - Or match the PAN_ID programmed in PANADR register.
- The Destination Address if present must:
  - Be the (short 16-bit) broadcast address (0xFFFF)
  - Or be a short (16-bit) address matching the SHORTADDR programmed in PANADR register.
  - Or be a long (64-bit) address matching the EUI_64.
- If the frame is a Beacon frame then the Source PAN ID must match the PAN_ID programmed in PANADR register, (or be 0xFFFF)

- If only the source address is present, in a data or MAC command frame, then the frame will only be accepted if the IC is configured to be a coordinator, (via the FFBC configuration bit in FF_CFG) and the Source PAN ID matches the PAN_ID programmed in PANADR register.

- If frame has no destination PAN ID and destination address, if will only be accepted (treated as though it is addressed to the broadcast PAN ID and broadcast short (16-bit) address) if the device is configured to allow implicit broadcast (via the FFIB configuration bit in FF_CFG).

- The FCS (CRC) must be correct for the frame to be accepted.

Note:
When QM33100 IC is configured for frame filtering to behave as PAN coordinator (FFBC is set to 1), it should accept frames with only source addressing fields when the destination PAN ID matches the PAN_ID programmed in PANADR register. The QM33100 IC chip is not correctly handling this case as specified by the IEEE802.15.4 standard [1] and the chip will incorrectly reject these frames.
This issue can be avoided by setting the FFIB configuration and then analyse the RX packet in software to discard any frame that don't match the PAN_ID programmed in PANADR register.

5.4.2 Frame filtering notes

The frame filtering does not take any notice of the Security Enabled field, in the frame control, so it is up to the host software to decode any security information and accept/reject the frame is it sees fit.

The decisions on frame rejection/acceptance with respect to illegal frame control octets is made after the first two octets of data are decoded, and at the end of reception of the address fields (as specified by the frame control octets) for the relevant addressing rules. When a frame is rejected, the reception is aborted immediately and the rejection is reported by the AFFREJ event bit in SYS_STATUS register.

While frame filtering can save some work on the part of the host system, prolonged listening with the QM33100 receiver on is a relatively power-hungry activity best employed only on equipment with a mains powered source.

All the configuration bits related to frame filtering are found in FF_CFG register.

Once the FFAA configuration bit is set, all acknowledgement frames are accepted, and the IC does not make any distinction between the 5-octet Imm-Ack (with frame version of 0 or 1) or the more complex Enh-Ack (with frame version of 2). It is up to the host software expecting an Enh-Ack to parse the received frame to identify the Enh-Ack and validate its addressing and other fields accordingly.

When enhanced beacon frames are used, they will be accepted even if the beacon's source PANID does not match the one programmed in the device.

NOTE: For Multipurpose frames the IEEE standard [1] says that Frame version field values other than 0x00 are reserved. The QM33100 does not correctly check the Frame version field of received Multipurpose frames. The host software MAC should validate the version of any Multipurpose frames received and handle the frame accordingly.

5.5 Automatic acknowledgment

The automatic acknowledgement functionality of the QM33100 allows the IC to automatically send an acknowledgement frame when a frame is received and validated that includes an acknowledgement request. The automatic acknowledgement functionality only operates when frame filtering is enabled and automatic acknowledgement is enabled.

In order for automatic acknowledgement to operate:

- Frame filtering must be enabled and the received data or MAC command frame must be correctly addressed and pass through the receive frame filtering, (see section *5.4 – Frame filtering* for details of frame filtering configuration).
- The ACK request bit in the frame control field of the received frame must be set.
- Auto-acknowledgement must be enabled by the AUTO_ACK configuration bit in SYS_CFG register.
- The received frame is an Data frames or a MAC command frame with the Frame Version field set to 0b00 or 0b01; the destination address and PAN ID matches the programmed PANID and programmed short address or exrended addess; and, AR (Acknowledgement Request) bit is set.

When these conditions are met the QM33100 will at the end of the reception automatically transition into transmit mode to send the 5-octet MAC acknowledgement frame as defined by IEEE802.15.4 standard [1].

The enhanced acknowledgment (Enh-Ack) when called for by Data or a MAC command frames with the Frame Version field set to 0b02, is not automatically generated. The host software is responsible for decoding/desecuring these frame version 2 frames, interpreting any Information Elements (IEs), and generating the Enh-Ack enhanced with any IEs necessary and securing it before transmission. The AAT bit will not be set.

5.5.1 Automatic ACK turnaround time

The IEEE802.15.4 standard [1] specifies a 12 symbol +/- 0.5 symbols turnaround time for ACK transmission. In the QM33100 this period is configurable via the ACK_TIM parameter in ACK_RESP_T. It should also be noted that running the CIA analysis will delay ACK response. To speed up ACK transmission FAST_AAT bit can be set or where the RX timestamp is not required, the CIA analysis may be disabled by clearing both CIA_IPATOV and CIA_STS configuration bits in *Sub-register 0x00:10 – System configuration*

5.5.2 Frame pending bit

The standard IEEE802.15.4 standard [1] MAC includes a frame pending bit in the frame control at the start of each frame. This bit can be set to indicate more data is coming or in the case of acknowledging a Data Request MAC command frame to indicate that the responding node has data to send to the node soliciting the ACK. Please refer to the standard [1] for details of this. The QM33100 will automatically determine whether to set the frame pending bit in the automatically-generated ACK frames based on:

- the received frame being a Data Request MAC command frame.
- the address of the device sending the MAC command (Data Request) frame matches one of the four 16-bit addresses programmed into LE_PEND_01 or LE_PEND_23 registers and the data pending bits are set in FF_CFG register: LE0_PEND – LE3_PEND.
- the address of the device sending the MAC command is 16-bits and SSADRAPE bit is set
- the address of the device sending the MAC command is 64-bits and LSADRAPE bit is set
- security bit is not set in Frame Control and frame version is 0 or 1

5.5.3 Host notification

The AAT status bit (SYS_STATUS) indicates that an acknowledgement has been requested. When the FAST_AAT bit is disabled the AAT bit is set at the same time as the RXFCG status event (indicating a good CRC at the end of frame reception). However if FAST_AAT bit is enabled the AAT bit is set at the same time as the RXFR status event.

If automatic acknowledgement is enabled then the AAT bit can be used during receive interrupt processing to detect that acknowledgement is in progress and thus avoid taking any action until the transmission of the acknowledgement is completed, and signalled by the TXFRS (Transmit Frame Sent) event.

Note: If automatic acknowledgement is not enabled, then the AAT status bit should be ignored.

5.6 Transmit and automatically wait for response

The QM33100 has the ability to automatically turn on its receiver after a transmission has completed in order to receive a response. This may also include an optional delay configuration between the end of the transmission and the enabling of the receiver. This is controlled by any of the TX and wait-4-response commands (TXW4R, TXW4RCCA, TXW4RDLY, TXW4RDLYREF, TXW4RDLYTS, and TXW4RDLYRS) and the W4R_TIM parameter in the Sub-register 0x01:00 – Acknowledgement time and response time.

Note: If the response that is received is a correctly addressed data (or MAC command) frame requesting an acknowledgement, and assuming frame filtering and automatic acknowledgement are enabled, then the QM33100 will transmit the ACK before it transitions into **IDLE_PLL** state.

5.7 Pseudo Clear Channel Assessment (CCA) mechanism

For Wireless Sensor Networks application, most of the MAC protocols rely on Clear Channel Assessment (CCA) to avoid collisions with other packets in the air. This consists in sampling the air for a short period to see if the medium is idle before transmitting. For most radios this involves looking for the RF carrier, but for UWB where this is not possible, one approach is to just look for preamble to avoid conflicting transmissions, since any sending of preamble during data will typically not disturb those receivers who are demodulating in data mode.

QM33100 has a simple Clear Channel Assessment (CCA) mechanism feature that can be employed before a packet transmission. It works by sample the air for a small amount of time (as configured by PRE_TOC) to see if preamble can be detected, then if preamble is not seen the transmission is initiated, otherwise a failed

transmission is reported with CCA_FAIL event and the host can then defer the transmission typically for a random back-off period after which transmission is again attempted with CCA.

QM33100 will return to **IDLE_PLL** for the back-off period and do not receive the packet whose preamble was detected, since the MAC (and upper layer) wants to transmit and not receive at this time.

To transmit a packet with CCA check CMD_CCA_TX command is used. If the device has detected a clear channel, the packet will be sent and TXFRS event reported. Although the detection of clear channel is as a result of preamble timeout counter expiring, the RXPTO event will not be raised in this case.

Although the CCA can be used to avoid collisions with other packets on the air, the mechanism is only looking for preamble thus will not detect PHR or data phases of the packet.

This CCA mechanism may have little benefit, but costs energy drain by turning on the receiver for a period before every transmission. The aloha CCA mechanism of simply sending may be more efficient, since even in collision cases message delivery can succeed. It is recommended that careful analysis and experimentation be undertaken for the target use cases to decide on this point.

5.8 Data confidentiality and authenticity

IEEE802.15.4 standard [1] supports the following security services: data confidentiality, data authenticity and replay protection. QM33100 has an advanced encryption standard (AES) engine that can perform cryptographic transformation on received frames or prior to transmission of frames. For this purpose the AES engine can use a TX Buffer, Rx Buffer or a separate scratch buffer as shown in Figure 23.

The AES engine performs encryption/decryption and authentication of data and can either perform the operation within the same buffer or transfer the data (during the process) from one buffer to another (e.g. from RX to TX, or from scratch buffer to TX or STS_KEY register).

The AES engine can also decrypt key data (i.e. an encrypted STS_KEY) into the STS_KEY register. This means that the host can write encrypted STS_KEY over SPI into scratch RAM memory and then run the QM33100 AES engine to decrypt this into the STS_KEY register prior to use of the secure ranging modes (see6 Secure ranging / timestamping).

The AES-DMA engine is fully configurable by the host from a source/destination/sub-address/burst-size point of view. The completion of the AES operation (AES_DONE event) as well as error conditions such as an AES decrypt tag error or DMA error (AES_ERR event) are signalled in the system status register (SYS_STATUS). There is also an AES status register (AES_STS) which gives more details about the performed AES operation.

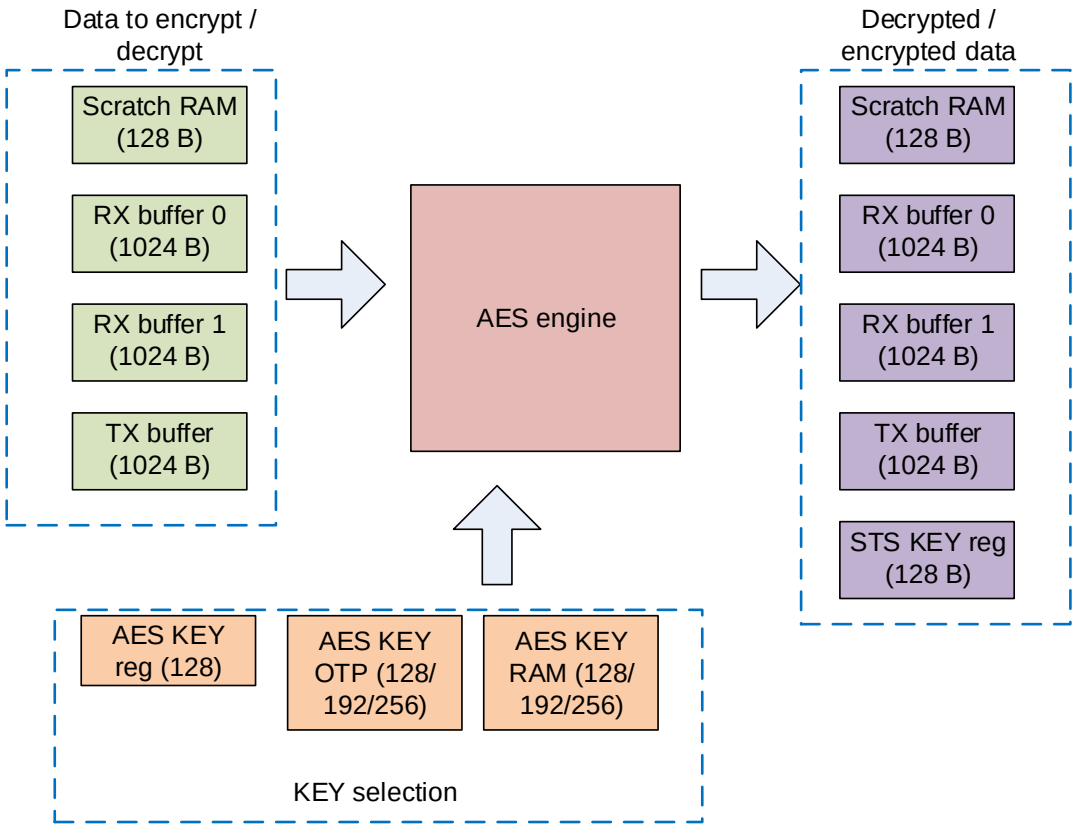

Figure 23: AES engine source/destination buffers

5.8.1 AES Configuration

Before the encryption or decryption of the data, the AES engine needs to be configured with:

- the MIC size (TAG_SIZE), which can be 0, 4, 6, 8, 10, 12, 14 or 16 bytes
- the KEY size (128, 192 or 256-bits) and location (KEY can be stored in QM33100 registers, or OTP, or AES KEY RAM). If a KEY from QM33100 registers or AES KEY RAM is used then it needs to be written to AES_KEY register or AES KEY RAM location.
- the type of operation (encryption or decryption)
- the type of AES core to use: GCM or CCM*

5.8.2 General TX Encryption flow

The following steps are taken to transmit an encrypted data (assuming above configuration steps are already done):

- Write the IV value in the AES_IV registers: AES_IV0, AES_IV1, AES_IV2, and AES_IV3. GCM uses 96-bit IV value and CCM* 128-bit.
- Specify source buffer (SRC_PORT) from which the plaintext data is taken and destination buffer (DST_PORT) into which the encrypted data is placed.
- Load the selected AES key. When using CCM* mode, AES key should be loaded each time prior to starting of AES engine. This key update is needed even if the key remains the same.

- Start the AES process (DMA + encryption) by setting the AES_START bit. Wait for AES_DONE or AES_ERR status events.
- Repeat the above steps to encrypt more data, e.g. when using the scratch buffer the max amount of data that can be processed at a time is 127 bytes.
- Once complete, the host can issue start TX command to transmit the encrypted data frame.

5.8.3 General RX Decryption flow

The following steps are taken to decrypt received encrypted data (assuming above configuration steps are already done):

- Write the IV value in the AES_IV registers: AES_IV0, AES_IV1, AES_IV2, and AES_IV3. GCM uses 96-bit IV value and CCM* 128-bit.
- Specify source buffer (SRC_PORT) from which the encrypted data is taken and destination buffer (DST_PORT) into which the plain text data is placed. For example encrypted RX data is available in RX buffer, which can then be decrypted into the same RX buffer.
- Load the selected AES key. When using CCM* mode, AES key must be loaded each time prior to starting of AES engine. This key update is needed even if the key remains the same.
- Start the AES process (DMA + decryption) by setting the AES_START bit. Wait for AES_DONE or AES_ERR status events.
- Read resulting plaintext RX data from the buffer
- Repeat the above steps to decrypt more data, e.g. when using the scratch buffer then the max amount of data that can be processed at a time is 127 bytes.

5.8.4 Sample use cases

5.8.4.1 Encrypt TX data prior to transmission

These are the steps to encrypt TX data prior to transmission:

- Configure the AES KEY, e.g. the host writes the desired AES KEY into the AES_KEY register.
- Configure the AES core type (CCM* or GCM), source and size of the KEY,
- Select encryption mode and MIC size
- Construct the nonce, and set up header and payload buffer pointers (source (SRC_PORT) and destination (DST_PORT)). For example host can write plaintext data into the TX buffer and then select the destination as the TX buffer and the data will be encrypted into the same buffer. Otherwise host could write the plaintext data into the scratch memory and then instruct the AES engine to encrypt it into the TX buffer.
- Run the AES engine and check status
- Transmit the encrypted packet

5.8.4.2 Decrypt RX data after packet reception

These are the steps to decrypt RX data following a packet reception:

- Following good packet reception, the host needs to read the received data and check the frame format, security and then configure the AES engine to decrypt the payload.
- Configure the AES KEY, e.g. the host writes the desired AES KEY into the AES_KEY register.
- Configure the AES core type (CCM* or GCM), source and size of the KEY,

- Select encryption mode and MIC size
- Construct the nonce, and set up header and payload buffer pointers (source (SRC_PORT) and destination (DST_PORT)). For example host can set the destination to be the same as the source, i.e. the same RX buffer. This allows the encrypted data in the buffer to be overwritten by the decrypted (plaintext) data.
- Run the AES engine and check status
- Read out the decrypted data

5.8.4.3 Decrypt STS KEY into STS KEY registers

These are the steps to decrypt encrypted STS KEY (transferred over SPI into the scratch buffer) into the STS_KEY register e.g. prior to transmission or reception of packets with STS:

- The host and QM33100 should have same AES KEYs programmed (they should be paired). The QM33100 KEY is stored in OTP.
- The host would then encrypt the STS KEY and transfer it over SPI into the scratch buffer, there should be no header but the MIC should be present
- Configure the AES core type (CCM* or GCM), source and size of the KEY, and the MIC size
- Construct the nonce, and set up header and payload buffer pointers (source and destination). The source (SRC_PORT) is the scratch buffer and destination (DST_PORT) STS_KEY register.
- Run the AES engine and check AES status events (AES_DONE, AES_ERR)

5.8.5 Note on STS KEY decryption

Consider a following example. A Secure Element host (SE) which derives STS KEY (prior to encryption) as: sts_key[16] = {**0xaa**, 0xbb, 0xcc, 0xdd, 0x11, 0x22, 0x33, 0x44 …. 0x55, 0x66, 0x77, 0x88}, i.e. the 128-bit STS_KEY = 0x**aa**bbcc.....667788.

The SE encrypts this STS_KEY (with STS KEY encryption KEY (STS_KEK)) and writes it into the device (scratch buffer), then the device needs to decrypt it with a matching STS_KEK from e.g. OTP and write the decrypted STS KEY into STS KEY registers (STS_KEY).

The DMA engine inside AES block can be configured as either big endian (CP_END_SEL = 0) or little endian (CP_END_SEL = 1), see DMA_CFG register. When big endian is used, the MSB of the STS KEY is in the lowest address otherwise the MSB is the highest address.

Considering, as an example, if sts_enc = {**0xaa**, 0xbb, 0xcc, 0xdd, 0x11, 0x22, 0x33, 0x44 …. 0x55, 0x66, 0x77, 0x88}, 0xaa is sent first to AES engine. Thus the data is with big endian format and is decrypted into STS_KEY register as shown in Table 14 (**0xaa** is in the lowest address location):

Table 14: Decrypted STS KEY bytes (with big endian format)

Address	Value		
02:0C	**0xaa**	02:14	0x00
02:0D	0xbb	02:15	0xee

02:0E	0xcc	02:16	0xff
02:0F	0xdd	02:17	0x00
02:10	0x11	02:18	0x55
02:11	0x22	02:19	0x66
02:12	0x33	02:1A	0x77
02:13	0x44	02:1B	0x88

For the little endian format, the STS KEY data will look like, as shown in Table 15 (**0xdd** is in the lowest address location):

Table 15: Decrypted STS KEY bytes (with little endian format)

Address	Value	Address	Value
02:0C	**0xdd**	02:14	0x00
02:0D	0xcc	02:15	0xff
02:0E	0xbb	02:16	0xee
02:0F	0xaa	02:17	0x00
02:10	0x44	02:18	0x88
02:11	0x33	02:19	0x77
02:12	0x22	02:1A	0x66
02:13	0x11	02:1B	0x55

But what wis actually required is that the STS KEY data looks like, as shown in Table 16:

Table 16: Decrypted STS KEY bytes (with little endian format and swapped by SE prior to encyption)

Address	Value	Address	Value
02:0C	**0x88**	02:14	0x44
02:0D	0x77	02:15	0x33
02:0E	0x66	02:16	0x22
02:0F	0x55	02:17	0x11
02:10	0x00	02:18	0xdd

02:11	0xff	02:19	0xcc
02:12	0xee	02:1A	0xbb
02:13	0x00	02:1B	0xaa

Thus the SE must re-arrange the data before encryption, the bytes (of the derived STS KEY) need to be swapped: byte[15]=byte[0]; byte[14]=byte[1] etc... prior to encryption with STS KEK.

5.8.6 Note on source and destination buffers

When specifying a source and destination buffer address (and offset within the buffer) the buffer size should be sufficient for the DMA transfer. If the size is smaller than the transfer, the DMA transfer error (AES_ERR) may not be correctly flagged by the device. In some circumstances the internal AES block can lock up and the only way to recover is to perform a reset of the device (soft or hard)).

To avoid such errors it is require to always ensure that the buffers and sizes of transfers should be properly configured, i.e.:

Header length + Payload length + MIC length < Total buffer size.

6 Secure ranging / timestamping

One of the main features of the QM33100 (differentiating it from the DW1000) is its secure timestamping ability.

The timestamping ability of the DW1000 is based on accumulated correlation of the repeated symbols of the 802.15.4 standard preamble sequence. Please refer to the IEEE 802.15.4 standard [1], for more details of the packet format. When the SFD is detected a coarse receive timestamp is taken and then the accumulator state, which essentially provides the radio channel impulse response, CIR, is used to find the first arriving ray of RF energy and adjust the RX timestamp to give the sub-nanosecond precision achieved by the IC.

Since the CIR accumulation is based on a repeated symbol of a known sequence of pulses defined by preamble code, it is possible for an attacker to send this same symbol (set of pulses) with a time offset which makes it appear like an earlier arriving ray in the CIR, and thus confuse the IC into reporting an earlier arrival time for the message. This could, for instance, foreshorten a ranging result, thus fooling an access system into thinking the user is closer than he actually is.

To remove the possibility of such an attack the QM33100 has the ability to include a scrambled sequence in the packet, which can be used to obtain an RX timestamp that cannot be attacked in this way. This scrambled timestamp sequence (STS) is generated in accordance with the IEEE 802.15.4z amendment [2], and consists of a sequence of randomised pulses generated by an AES-128 CPRNG (cryptographic pseudo-random number generator) block in counter mode. This is also termed a deterministic random bit generator (DRBG). Only valid transmitters and receivers know the correct seed (i.e., key and data) to generate the sequence for transmission and for reception to cross correlate and accumulate to produce a CIR estimate from which to determine the RX timestamp.

When the STS modes are enabled the QM33100 transmitter will insert the STS in the position shown in Figure 13, and the receiver will expect it to be present accordingly. The STS is generated by the AES-128 block output and is determined by a seed consisting of a 128-bit key, and a 128-bit nonce (a number that should only be used once). The nonce is updated during the STS generation by incrementing the counter once for every 128 pulses produced. When transmitting and receiving devices are using aligned seeds (i.e., same key and nonce) with the counter values aligned, the receiver generates a secure version of the CIR by correlating with the matching scrambled pulse sequence. The resultant timestamp is thus secure against attack.

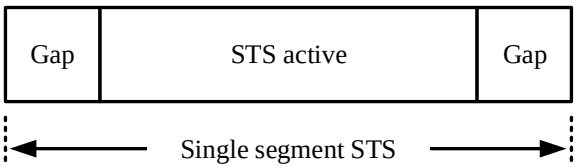

Figure 24: STS structure

As shown in Figure 24, the STS consists of an active segment bracketed at either end by a gap (of 512 chips or approx. 1 µs). The length of the STS (active segment) is specified in units of 512 chips (~1 µs). In BPRF mode, where the STS pulse spacing is 8 chips, the mandatory STS length of 64 units has an active segment of 4096 pulses. For convenience we sometimes call this 512-chip (~1 µs) period an *STS symbol*.

To contrast the three STS packet configuration options:

- STS Packet Configuration mode 1 has the STS directly after the SFD which means it is early (and in a deterministic position). This allows the CIA to start processing the STS CIR earlier (assuming it is not processing the preamble CIR), while the IC continues to receive the PHR and the data payload. This mode saves time and energy (as compared to configuration mode 2). However data reception performance may be poorer when there is a misalignment of the STS sequences, i.e., where the transmitter's seed info (including counter) does not match the receiver's giving poor correlation.

- STS Packet Configuration mode 2 has the STS at the end of the packet. This has the benefit that the receiver can receive the data frame as normal even if it has misalignment of the STS sequence generation seeds between the transmitter and the receiver. Also the frames sent by a device using this mode can be received by a device employing the original IEEE802.15.4 standard [1] UWB frame structure, which might be an advantage in some circumstances. In using STS Packet Configuration mode 2 the frame payload can carry information allowing the receiving node to determine the correct alignment of the STS generation seed (key + nonce data) that it can achieve alignment for future messages. Obviously, transmission of sensitive information like the STS key and nonce counter state needs to be encrypted/protected so that an attacker cannot learn this secret.
  N.B. In this mode, for correct operation it is required that the data length is non-zero.

  Note: This mode can potentially be attacked. The mechanism has the attacker corrupting the true PHR to signal a longer data payload than the true payload, adding data payload after the true payload making sure that the CRC of the longer frame is correct, while also recording the true STS of the valid transmitter and playing it at the end of the longer payload but altering its symbol alignment so that it appears to arrive earlier in time resulting in a shorter time-of-flight estimate than would otherwise be the case. This attack can be easily thwarted by: (a) ensuring that the true message length always known, i.e. the length is either a constant or is included securely encrypted in the frame data payload, and then (b) having the receiver discard and ignore frames of the wrong length. Simply encrypting the data payload should also be sufficient to thwart this attack, because, since the attacker does not know the encryption key, it will be unable to generate a valid MIC (message integrity check code) for the longer data payload, and then the receiver's incoming security processing should simply discard and ignore the invalid frame.

- STS Packet Configuration mode 3 has STS directly after the SFD (similar to configuration mode 1), however in this mode the UWB packet ends with STS and there is no PHR or PHY payload (MAC frame) following. This could save time and energy (as compared to configuration modes 1 and 2).

The STS packet configuration is configured via the CP_SPC field in *Sub-register 0x00:10 – System configuration*.

The STS length is configured via the CPS_LEN field in the STS_CFG register. The STS length is programmable in steps of 8 units (i.e., each step is ~8 µs, or 8 x 512 chips). The minimum length supported is 32 STS symbols (one STS symbol is made up of 512 chips), and the maximum supported length is 2048 STS symbols. The IEEE 802.15.4z mandates only a single length of 64 x 512 chips, (or 64 x 64 pulses @ 64 MHz PRF), which is approximately 64 µs duration. As noted the equivalent PRF during STS is 64 MHz.

The 128-bit STS key is programmed into STS_KEY, while the 128-bit initial value for the nonce is loaded into the device via STS_IV.

When the QM33100 is enabled to transmit or receive a packet incorporating an STS, the AES-128 block generates a randomised set of pulses of positive and negative polarity, the CPRGN is shown in Figure 25 below, advancing the counter by one step for every 128 pulses generated. The counter would thus advance by 32 for each completed transmission, or reception, of the IEEE 802.15.4z mandated 64 x 64 pulse BPRF mode STS.

To achieve secure ranging/timestamping both parties of a two-way ranging exchange should be configured the same way, i.e. to use the same STS length and PRF and the same key and nonce values for the generation of their STS sequences.

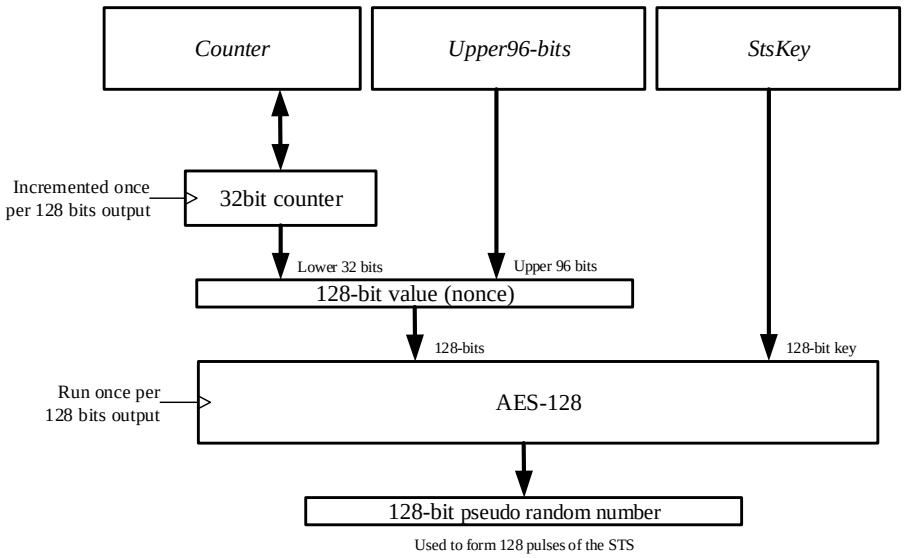

Figure 25: AES in counter mode based CPRNG

Since the counter automatically advances as the STS is generated in the transmitter and receiver, the counter state in two units can remain aligned as messages are sent and received in a typical two-way ranging exchange. Where alignment is lost, or to maintain alignment in multi-node systems the counter state should be reprogrammed by the setting the new value into the STS_IV register. Optionally, the counter state could be sent from one device to another to update the STS_IV register. Clearly any communication of the seed (key and/or IV) would have to be done securely to avoid them becoming known to an attacker. Note also that during transmission and reception of the STS the state of the counter may be in flux as it generates the scrambled symbol sequence, so before any reading or writing of the STS_IV register, it is recommended that the QM33100 is not active with any transmission and reception activities.

When STS is included in the packet there are two separate accumulations of CIR in the receiver. The first accumulation occurs during the preamble reception and the second occurs during the STS reception. Both these CIR may be read through Register file: 0x15 – Accumulator CIR memory. Access to the CIR data is not needed for ranging measurements, however this ACC_MEM may be of interest to the system design engineers to visualise the radio channel for diagnostic purposes.

The accurate receive timestamp for a received frame is computed by finding the first arriving path or ray within the accumulated CIR, which is the function of the leading edge algorithm, also called the channel

impulse analyser (CIA) algorithm. This algorithm processes the CIR (in the accumulator) to find the leading edge and estimate the RX timestamp for a received frame. The CIA may be applied to CIR resulting from preamble and/or STS sequences. The CIA is configured and controlled through the Sub-registers in CP_CONF0, CP_CONF1, IP_CONF0 and IP_CONF1.

Assuming it is enabled, once both the preamble and STS CIRs have been received, the CIA begins running on the preamble first followed by the analysis of the STS CIR. The CIA delivers the receive timestamp for a received frame through the RX_STAMP field in Sub-register 0x00:60 – Receive time stamp. To ensure the integrity of the receive timestamp based on the STS, the user must check that the CP_TOAST quality status indicator (in CP_TS) is not indicating any issues with the STS CIR analysis.

NB: The host should not attempt to read ACC_MEM until the CIA has finished its processing, (as signalled by the CIADONE event status flag), since this may give rise to a conflict with the CIA accesses to the CIR memory which may interfere with the generation of a good RX timestamp result.

Note: It is also very important to assess that quality of the STS accumulation before accepting that a receive timestamp CP_TOA value is sufficiently good enough to be trusted in a two-way ranging exchange. This is a unitless value that summarizes the results of the STS quality algorithm that is monitoring the reception of the STS. For high signal levels this measurement should be very high but its value will reduce as the signal level reduces. An ACC_QUAL value (in STS_STS) that is <60% of the STS length indicates that the reception of the STS was too poor for the first path analysis to obtain a reliable ToA and so the received timestamp based on STS should not be trusted.

For example for a preamble sequence using 64 MHz PRF (i.e. TX_PCODE is in range 9-12) and an STS *length of 128 symbols (i.e. CPS_LEN is set to 15), the* STS *based receive timestamp (CP_TOA) value should only be trusted when the* STS *accumulation quality (CPS_ACCQUAL) value is greater than ≥ 77 (i.e. 60 % of 128).*

As mentioned above STS uses a continually varying sequence. This means that a colliding packet will not line up with the desired signal. As a result, the ToA will be unaffected. If security is not a concern, then overhead of key management and counter control is undesirable. For this reason, the QM33100 includes a special STS mode that uses a code optimized for ToA performance. This is known as the Super Deterministic Code (SDC). If SDC is enabled (by CP_SDC bit), then the STS will be populated with the SDC code. Since it is a time varying sequence optimized for ToA performance it will better tolerate packet collisions without requiring any key management.

It is important to remember that the SDC mode does not provide security but will increase confidence in the ToA when the on-air packet density is high. For this reason, we would recommend that an SDC based STS is used when security is not a requirement.

7 Other features of the IC

7.1 External Synchronisation

This feature is used to synchronise QM33100 with external clocks, events or with other QM33100's. For example, this would be required in a TDoA RTLS system employing wired clock synchronisation of the anchor nodes.

The QM33100 external synchronisation feature allows the following functions:

a) The ability to reset the internal system counter in a deterministic way with respect to the assertion of the SYNC input pin and an external 38.4 MHz clock supplied on the EXTCLK pin.

b) The ability to initiate the transmission of a frame in a deterministic way with respect to the assertion of the SYNC input pin and an external 38.4 MHz clock supplied on the EXTCLK pin.

c) The ability to synchronise receive time stamping to an external counter.

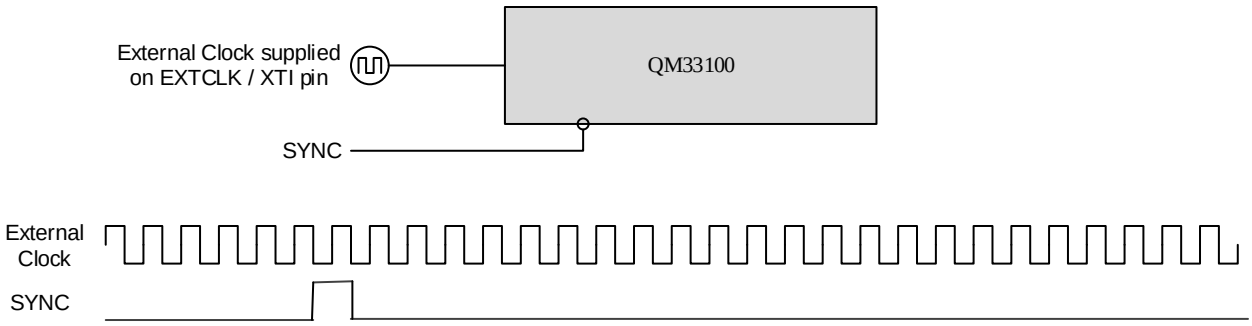

Figure 26: QM33100 External Synchronisation Interface

The SYNC input pin must be source synchronous with an external 38.4 MHz frequency reference clock supplied on the EXTCLK pin. The SYNC input pin is sampled on the rising edge of EXTCLK. Refer to the QM33100 Datasheet for setup and hold times of the SYNC pin. The SYNC input provides a common reference point in time to synchronise the QM33100 with the accuracy necessary to achieve high resolution location estimation.

7.1.1 One Shot Timebase Reset (OSTR) Mode

One Shot Timebase Reset (OSTR) mode allows a reset to be applied to the timebase counter used for timestamping in QM33100 at a deterministic and predictable time relative to a synchronisation event. Any given device will reset the counter at a repeatable time to within 300ps (typically less than 100ps) variation. Process variation between parts introduces a deterministic error that can be calibrated out as part of the necessary calibration process to compensate for cable transmission delays in a wired synchronization system. When several QM33100s are driven by the same reference clock and an external SYNC signal, their internal timebases can be synchronised very accurately (allowing for the deterministic delays associated with the distribution network for the reference clock and SYNC signal).

To configure QM33100 for OSTR mode, the OSTR_M bit in the EC_CTRL register is set and the OSTS_WAIT value is set to the desired delay value. When a counter running on the 38.4 MHz external clock and initiated on the rising edge of the SYNC signal equals the OSTS_WAIT programmed value, the QM33100 timebase counter will be reset. See *Register file: 0x04 – External sync control* and RX calibration for register details.

At the time the SYNC signal is asserted, the clock PLL dividers generating the QM33100 125 MHz system clock are reset to ensure that a deterministic phase relationship exists between the system clock and the asynchronous 38.4 MHz external clock. For this reason, the OSTS_WAIT value programmed will dictate the phase relationship and should be chosen to give the desired phase relationship, as given by OSTS_WAIT modulo 4. An OSTS_WAIT value of 33 decimal is recommended, but if a different value is chosen it should be chosen so that OSTS_WAIT modulo 4 is equal to 1, i.e. 29, 37, and so on.

7.2 External power amplification

In some geographic regions for certain situations (e.g. for emergency first responder use in ETSI UWB regulations for EU) it is permitted to send at +20 dB above the normal UWB regulation levels. To achieve this with the QM33100 it is necessary to employ external amplification of the transmitted signal. The QM33100 provides signals (using the GPIO lines in a special mode) to control the turn-on of an external power amplifier and to control an external analog switching of the transmitter and receiver signal paths appropriately. This mode of operation utilises the QM33100 pins , and as configured via the fields of the GPIO_MODE register.

Care should be taken when using this feature to ensure that necessary regulatory requirements have been fulfilled.

There is a separate application note giving details of the external power amplification. This includes the circuit diagram, details of configuration and various design considerations that apply. Please consult with Qorvo's applications support team for details.

7.3 Using the on-chip OTP memory

The QM33100 has a small amount of one-time-programmable (OTP) memory intended for device specific configuration or calibration data. Some areas of the OTP memory are used to save device calibration values determined during QM33100 testing, while other OTP memory locations are intended to be set by the customer during module manufacture and test.

For example, an OTP memory area is reserved for customers to programme the EUI that is loaded into EUI_64 as the IC comes out of reset (see EUI_64 for details of the EUI functionality).

This section lists the OTP memory areas defining their functionality and describes the algorithm for programming values into the OTP memory, and how to read values from the OTP memory. Access to OTP memory is achieved using Register file: 0x0B – OTP memory interface.

7.3.1 OTP memory map

The OTP memory locations are as defined in Table 17. The OTP memory locations are each 32-bits wide, OTP addresses are word addresses so each increment of address specifies a different 32-bit word.

Table 17: OTP memory map

Address	Size (Used Bytes)	Byte [3]	Byte [2]	Byte [1]	Byte [0]	Programmed By
0x00	4	64 bit EUID				Customer
0x01	4					
0x02	4	Alternative 64bit EUID (Selected via reg/SR register)				Customer
0x03	4					
0x04	4	LDOTUNE_CAL				Prod Test
0x05	4					
0x06	4	{“0001,0000,0001”, “CHIP ID 5 nibbles (20 bits)”}				Prod Test
0x07	4	{“0001” , “LOT ID – 7 nibbles (28bits)”}				Prod Test
0x08	4	-	Vbat @ 3.0 V [23:16]	Vbat @ 3.62 V [15:8]	Vbat @ 1.62 V [7:0]	Prod Test
0x09	2				Temp @ 22 °C [7:0]	Prod Test
0x0A	0	BIASTUNE_CAL				Prod Test
0x0B	4	Antenna Delay – RFLoop				Prod Test
0x0C	4	AoA Iso CH9 RF2->RF1	AoA Iso CH9 RF1->RF2	AoA Iso CH5 RF2 -> RF1	AoA Iso CH5 RF1->RF2	Prod Test
0x0D	0	W.S. Lot ID [3]	W.S. Lot ID [2]	W.S. Lot ID [1]	W.S. Lot ID [0]	Prod Test
0x0E	0			W.S. Lot ID [5]	W.S. Lot ID [4]	Prod Test
0x0F	0		W.S. Wafer Number	W.S. Y Loc	W.S. X Loc	Prod Test
0x10	4					Customer
0x11	4					Customer
0x12	4					Customer
0x13	4					Customer
0x14	4					Customer
0x15	4					Customer
0x16	4					Customer
0x17	4					Customer
0x18	4					Customer
0x19	4					Customer
0x1A	4					Customer
0x1B	4					Customer
0x1C	4					Customer
0x1D	4					Customer
0x1E	2				XTAL_Trim[6:0]	Customer
0x1F					OTP Revision	Customer
0x20	4	RX_TUNE_CAL: DGC_CFG0				Prod Test
0x21	4	RX_TUNE_CAL: DGC_CFG1				Prod Test
0x22	4	RX_TUNE_CAL: DGC_CFG2				Prod Test
0x23	4	RX_TUNE_CAL: DGC_CFG3				Prod Test

0x24	4	RX_TUNE_CAL: DGC_CFG4	Prod Test
0x25	4	RX_TUNE_CAL: DGC_CFG5	Prod Test
0x26	4	RX_TUNE_CAL: DGC_CFG6	Prod Test
0x27	4	RX_TUNE_CAL: DGC_LUT_0 – CH5	Prod Test
0x28	4	RX_TUNE_CAL: DGC_LUT_1 – CH5	Prod Test
0x29	4	RX_TUNE_CAL: DGC_LUT_2 – CH5	Prod Test
0x2A	4	RX_TUNE_CAL: DGC_LUT_3 – CH5	Prod Test
0x2B	4	RX_TUNE_CAL: DGC_LUT_4 – CH5	Prod Test
0x2C	4	RX_TUNE_CAL: DGC_LUT_5 – CH5	Prod Test
0x2D	4	RX_TUNE_CAL: DGC_LUT_6 – CH5	Prod Test
0x2E	4	RX_TUNE_CAL: DGC_LUT_0 – CH9	Prod Test
0x2F	4	RX_TUNE_CAL: DGC_LUT_1 – CH9	Prod Test
0x30	4	RX_TUNE_CAL: DGC_LUT_2 – CH9	Prod Test
0x31	4	RX_TUNE_CAL: DGC_LUT_3 – CH9	Prod Test
0x32	4	RX_TUNE_CAL: DGC_LUT_4 – CH9	Prod Test
0x33	4	RX_TUNE_CAL: DGC_LUT_5 – CH9	Prod Test
0x34	4	RX_TUNE_CAL: DGC_LUT_6 – CH9	Prod Test
0x35	4	PLL_LOCK_CODE	Prod Test
0x36 – 0x5F		UNALLOCATED	Customer
0x60	1	**QSR Register (Special function register)**	Reserved
0x61	4	Q_RR Register [7:0]	Reserved
0x62 – 0x77	4	UNALLOCATED	Customer
0x78	4	AES_KEY[127:96] (big endian order)	Customer
0x79	4	AES_KEY[95:64] (big endian order)	Customer
0x7A	4	AES_KEY[63:32] (big endian order)	Customer
0x7B	4	AES_KEY[31:0] (big endian order)	Customer
0x7C	4	AES_KEY[255:224] (big endian order)	Customer
0x7D	4	AES_KEY[223:192] (big endian order)	Customer
0x7E	4	AES_KEY[191:160] (big endian order)	Customer
0x7F	4	AES_KEY[159:128] (big endian order)	Customer

The QM33100 supports three AES_KEY format of length 128, 192 or 256 bits. Hence, the address 0X78 to 0X7F can be written with either two 128 bits AES_KEY, one 192 bits AES_KEY or one 256 bits AES_KEY. The AES_KEY must be written as specified in Table 17: OTP memory map, in big endian order.

The QSR (“Special function register”) is a 32-bit segment of OTP that is directly readable via the register interface upon power up. To program the SR register follow the normal OTP programming method but set the OTP address to 0x60. As this is part of OTP boot sequence the new value will be present in the QSR register following the next boot up sequence. The value of the SR register can be directly read back at address Register file: 0x0B – OTP memory interface.

Table 18: OTP_SRDAT Register

Bit	Function
31:4	Reserved. Defaults to all “0”. If programming the OTP_SRDATA register these bits must be set to “0”.
3:2	**SPI_SR_EN[1:0] :** Set to “01” to enable bits [1:0] to be used
1	**SPI_SR_PH :** Set SPI Phase mode to this value if bits [4:3] are set to “01”
0	**SPI_SR_POL :** Set SPI Polarity mode to this value if bits [3:2] are set to “01”

The various tune values (LDOTUNE_CAL, BIASTUNE_CAL etc.) can be automatically loaded from OTP into the respective registers. This will ensure the device is using optimal calibration and tuning parameters for given configurations. The automatic loading (kicking) of these values is described insection 8.2.12.3 Sub-register 0x0B:08 – OTP configuration.

It is left up to the customer, but in Qorvo’s evaluation kits which have been calibrated, the XTAL_Trim and OTP Revision values are programmed into OTP memory addresses 0x1e and 0x1f. Then in the dwt_initialise() API [3] the XTAL trim value is copied to XTAL_TRIM.

7.3.2 Programming a value into OTP memory

The programming of the OTP requires a sequence of steps please see API functions [3].

7.3.3 Reading a value from OTP memory

The reading of an OTP memory address consists of a number of steps. Please see the API function [3]: dwt_otpread(uint16 address, uint32 *array, uint8 length) of how to read OTP memory.

7.4 Measuring IC temperature and voltage

The QM33100 is equipped with a low speed 8-bit SAR A/D convertor which can be configured to sample values from an internal IC temperature sensor and from a battery voltage monitor on the VDD1 power supply input. These readings can be manually run under host control or they can be configured to be run automatically each time the QM33100 enters the **WAKE_UP** state. This automatic mode allows the temperature and voltage to be read while the device is in a low power state, which allows to read the temperature without internal heat up and the unloaded battery voltage. More details can be found in Register file: 0x08 – Transmitter calibration block section and the description of the ONW_RUN_SAR bit of Sub-register 0x0A:00 – AON on wake configuration.

7.5 The brownout detector

The QM33100 incorporates a brownout detection feature to warn the system about the condition of the IC supply voltage dropping below a pre-set brownout threshold of approx. 1.5 volts, which will reset the IC. The role of the brownout detector is to detect drops in the supply during the higher current TX or RX modes. This should not happen in normal operation when there is sufficient power supply current and capacitance available, but could happen in a battery powered design with insufficient remaining charge or insufficient

capacitance available to supply the current needs of the device. A brownout means that the IC circuits have insufficient supply for correct operation likely to result in poor performance or operational issues.

The elements involved in the brownout detection feature are:

- The VWARN event flag bit in SYS_STATUS. If a brownout event is detected this bit will get set and it will remain set until it is explicitly cleared by writing a 1 to it, or the IC is reset.

- The VWARN event can also be used to trigger an interrupt to the host microprocessor if the corresponding mask bit, VWARN_EN, is set in Sub-register 0x00:3C – System event enable mask.

Enabling the voltage comparator will set the VWARN event flag bit in SYS_STATUS register. This initial VWARN event should be ignored and the event flag should be cleared (by writing 1 to VWARN). Subsequent VWARN events will signal drop in the supply voltage.

7.6 Timers

The QM33100 incorporates a timer block with two timers that run from the external 38.4 MHz crystal / reference clock.

Each timer has a configuration to select one of eight division factors (applied to the reference input) to individually set the clocking rate for the timer – see TIMER0_DIV and TIMER1_DIV.

Each timer has a separate 21-bit terminal count configuration register, TIMER0_CNT and TIMER1_CNT.

Each timer can operate in a single shot mode where it stops when it reaches its programmed terminal count or can operate in a repeating mode where it automatically restarts counting from zero when the terminal count is reached – see TIMER0_MODE and TIMER1_MODE configuration bits.

Each timer has a mode where it can stop counting in response to the assertion of the GPIOIRQ event status bit – see TIMER0_GPIO and TIMER1_GPIO configuration bits.

Each timer has a separate control to start it counting, see TIMER0_START and TIMER1_START bits, but both timers share a common reset control TIM_RST

which stops them both counting.

Each timer has an event flag, TIMER0 and TIMER1, to signal the occurrence of its significant events, (i.e. the count reaching the programmed terminal count or the timer stopping due to GPIOIRQ), and separate mask bits TIMER0_EN and TIMER1_EN to enable the event to generate an interrupt. These events are also counted, see TIMER0_EVC and TIMER1_EVC.

Each timer has a configuration to gate its events into the coexistence block, where it can cause the assertion of the coexistence output pin – see TIMER0_COEX and TIMER1_COEX configuration bits.

The running count of each timer may be read from TIMER0_CNT and TIMER1_CNT respectively when the appropriate control for this operation, TIMER0_RDC and TIMER1_RDC respectively, is set.

7.6.1 Dual-timer block – reset and soft reset

By default the timer block state is not reset by a general soft reset of the IC, but this can be changed by setting the TIM_RST_MODE

configuration to one. The TIM_RST

control can be used to reset the timer block at any time. When TIM_RST_MODE

is zero, the soft reset will reset the host interface registers and bits, including those associated with the timers, but the timer block will keep the internally latched parameters, with the following operation:

Following the general soft reset the IC will return to the **IDLE_RC** state, where the timers are not clocked. To resume counting the host must either turn on the PLL LDO (bit 4) in the LDO_CTRL register to enable the crystal oscillator, or transition the IC into the **IDLE_PLL** state (using the AINIT2IDLE bit). The host will need to account for any time lost between the reset and the resumption of the clock counting.

The soft reset will reset the terminal count values TIMER0_CNT and TIMER1_CNT but will not reset the terminal counts latched internally to the timer block. The host software should reprogram these terminal counts as necessary before next starting the timer.

The soft reset will clear the TIMER0_EVC and TIMER1_EVC events counters, but the previous values are preserved in the timer block, so that at the next event the counter will be updated to the preserved value plus one.

The soft reset will clear the timer interrupt enable mask bits TIMER0_EN and TIMER1_EN, so these will need to be reenabled if needed.

8 The QM33100 register set

The QM33100 is controlled by an associated host microcontroller system using the SPI interface to access a series of registers within the device. The QM33100 register set includes configuration registers, status registers, control registers, data buffer registers, and diagnostic registers. Section *8.1 – Register map overview* gives an overview of the register layout and then section *8.2 – Detailed register description* describes each individual parameter in detail. There is also a set of single octet commands to initiate certain IC activities (e.g. TX, RX, etc) which are described in secion *9 – Fast Commands*. The SPI transaction formats are described in section *2.3 –* .

8.1 Register map overview

The register map overview is given in Table 19. This lists the registers in address order, by register file ID, giving the register file length in octets, its type (RO = Read-Only, RW = Read & Write, SRW = Special Read Write – see individual register descriptions for details about how the Read/Write access is special),and a brief high level description of the register. Section 8.2 gives a detailed description of each register.

Note: When writing to any of the QM33100 registers care must be taken not to write beyond the documented length of the selected register and not to write to any of the reserved register locations. Doing so may cause the device to malfunction.

Table 19: Register map overview

ID	Mnemonic	Description
0x00 – 0x01	GEN_CFG_AES	This is the part of the main register bank and Advanced Encryption Standard configuration
0x02	STS_CFG	Scrambled timestamp sequence configuration
0x03	RX_TUNE	Receiver tuning parameters
0x04	EXT_SYNC	External synchronisation control
0x05	GPIO_CTRL	Peripheral register bus access - GPIO control
0x06	DRX	Digital receiver tuning and configuration
0x07	RF_CONF	Analog RF configuration
0x08	RF_CAL	Transmitter calibration block
0x09	FS_CTRL	Frequency synthesiser control block
0x0A	AON	Always-On register set
0x0B	OTP_IF	One Time Programmable memory interface
0x0C – 0x0E	CIA	Channel Impulse Analysis control and diagnostic block
0x0F	DIG_DIAG	Digital diagnostics
0x11	PMSC	Power Management System Control block

ID	Mnemonic	Description
0x12	RX_BUFFER_0	Receive data buffer
0x13	RX_BUFFER_1	Second receive data buffer
0x14	TX_BUFFER	Transmit data buffer
0x15	ACC_MEM	Read access to Channel Impulse Response data
0x16	SCRATCH_RAM	Scratch RAM
0x17	AES RAM	AES key RAM
0x18	SET_1, SET_2	Double buffer diagnostic sets
0x1D	INDIRECT_PTR_A	Indirect pointer A buffer
0x1E	INDIRECT_PTR_B	Indirect pointer B buffer
0x1F	IN_PTR_CFG	Indirect pointer access configuration, and “fast” status register

8.2 Detailed register description

8.2.1 Terminology

Section 8.1 gives an overview of the QM33100 register set presenting all top level register file ID addresses in Table 19. This section describes in detail the contents and functionality of these register files and their sub-registers in separate sub sections. In each case the row from Table 19 is reproduced with a hexadecimal register ID, its length, type, mnemonic and one line description as follows:

ID	Length (octets)	Type	Mnemonic	Description

This is followed by a description of the parameters within that register file. All parameters are presented with format REG:RR:SS, where RR is register file ID (5-bits) and SS (7-bits) is the sub address. Where a register is made up of individual bits or bit-fields these are identified with mnemonic and default values as follows:

REG:RR:SS – Mnemonic – one line description																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<bits or bit-fields identified by a quoted mnemonic>																															
<default power-on-reset values are quoted as bits or values>																															

Then the fields or bits identified are described individually in detail.

Because many parameters are 4-octets long, the default presentation of the register values is as a 32-bit value. This may be sub-divided into fields of various bit widths down to single bit values. It should be noted that when reading these values via the SPI interface the octets are output least significant octet first. Also of note is that the indexed addressing modes allow individual octets to be accessed – a technique that may be employed to reduce SPI traffic when only part of a register needs to be read or written to.

Note: unused or reserved registers return 0xDEADDEAD when read. Unused or reserved bits/ bit fields within registers return the appropriate bits / bit fields from 0xDEADDEAD.

Each register file is described below:

8.2.2 Register file: 0x00-0x1 – General configuration registers and AES

ID	Length (octets)	Type	Mnemonic	Description
0x00 – 0x01	226	-	GEN_CFG_AES	Scrambled Timestamp sequence configuration and Status registers

Register map register files 0x00 and 0x01 are concerned with the use of the various device configurations and AES block configuration. It contains a number of sub-registers. An overview of these is given by Table 20. Each of these sub-registers is separately described in the sub-sections below.

Table 20: Register file: 0x00-0x1 – General configuration registers overview

Register file ID	OFFSET in Register	Mnemonic	Description
0x00	0x00	DEV_ID	Device Identifier
0x00	0x04	EUI_64	Extended Unique Identifier
0x00	0x0C	PANADR	PAN Identifier and Short Address
0x00	0x10	SYS_CFG	System Configuration
0x00	0x14	FF_CFG	Frame Filter Configuration
0x00	0x18	SPI_RD_CRC	SPI CRC read status
0x00	0x1C	SYS_TIME	System Time Counter
0x00	0x20	TX_FCTRL	Transmit Frame Control
0x00	0x28	DX_TIME	Delayed Send or Receive time
0x00	0x30	DREF_TIME	Delayed Send or Receive Reference time
0x00	0x34	RX_FWTO	Receive Frame Wait Timeout period
0x00	0x3C	SYS_ENABLE	System Event Enable Mask
0x00	0x44	SYS_STATUS	System Event Status Register
0x00	0x4C	RX_FINFO	RX Frame Information
0x00	0x60	RX_TIME	Receive Time Stamp
0x00	0x70	TX_TIME	Transmit Time Stamp
0x00	0x78	TX_RAW	Unadjusted/raw TX timestamp
0x00	0x7C	TX_ANTD	16-bit Delay from Transmit to Antenna
0x01	0x00	ACK_RESP_T	Acknowledgement Time and Response Time
0x01	0x04	TX_POWER	TX Power Control
0x01	0x08	CHAN_CTRL	Channel Control Register

Register file ID	OFFSET in Register	Mnemonic	Description
0x01	0x0C	LE_PEND_01	Low Energy device address 0 and 1
0x01	0x10	LE_PEND_23	Low Energy device address 2 and 3
0x01	0x14	SPI_COLLISION	SPI Collision Status
0x01	0x18	RDB_STATUS	RX Double Buffer Status
0x01	0x20	RDB_DIAG	RX Double Buffer Diagnostic Configuration
0x01	0x30	AES_CFG	AES Configuration
0x01	0x34	AES_IV0	The 3rd IV word for the AES GCM/CCM* core
0x01	0x38	AES_IV1	The 2nd IV word for the AES GCM/CCM* core
0x01	0x3C	AES_IV2	The 1st IV word for the AES GCM/CCM* core
0x01	0x40	AES_IV3	The 4th IV word for the AES GCM/CCM* core
0x01	0x44	DMA_CFG	The DMA Configuration Register
0x01	0x4C	AES_START	Start AES operation
0x01	0x50	AES_STS	The AES Status
0x01	0x54	AES_KEY	The 128-bit KEY for the AES GCM/CCM* core

8.2.2.1 Sub-register 0x00:00 – Device Identifier

ID	Length (octets)	Type	Mnemonic	Description
00:00	4	RO	DEV_ID	Device Identifier – includes device type and revision information

Register file: 0x00-0x1 – General configuration registers, sub-register 0x00 is the device identifier. This is hard-coded into the silicon. The value in this register is read-only and cannot be overwritten by the host system. The device ID will be changed for any silicon updates. The DEV_ID register is ideal to use in the host µP to validate that the SPI interface is operational. It is expected that the host system will validate that the device ID is the expected value, supported by its software, before proceeding to use the IC. The DEV_ID register contains the following sub-fields:

REG:00:00 – DEV_ID – Device Identifier																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RIDTAG																MODEL								VER				REV			
1	1	0	1	1	1	1	0	1	1	0	0	1	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	0	0

The fields of the DEV_ID register identified above are individually described below:

Field	Description of fields within Sub-register 0x00:00 – Device Identifier
REV reg:00:00 bits:3–0	Revision: This number will be updated for minor corrections and changes in device's operation
VER reg:00:00 bits:7–4	Version: This number will be updated if a new version is produced that has significant differences from the previous version. There are currently two versions; 0xDECA03**0**4 and 0xDECA03**1**4, the non-PDoA and PDoA device respectively.
MODEL reg:00:00 bits:15–8	The MODEL identifies the device type. The QM33100 is device type 0x03.
RIDTAG reg:00:00 bits:31–16	Register Identification Tag. It is planned that this will remain constant for all Qorvo parts. The value is 0xDECA in hex.

For the production QM33100 the Device ID is set to 0xDECA0304 or 0xDECA314. The register descriptions in this user manual relate to that QM33100 device and are not valid for any earlier sample parts.

8.2.2.2 Sub-register 0x00:04 – Extended Unique Identifier

ID	Length (octets)	Type	Mnemonic	Description
00:04	8	RW	EUI_64	Extended Unique Identifier – the 64-bit IEEE device address

Register file: 0x00-0x1 – General configuration registers, sub-register 0x04 of register file 0x00 is the Extended Unique Identifier register. For IEEE802.15.4 standard [1] compliance every device should have a unique 64-bit device identifier. The high-order 24-bits of the EUI are a *company identifier* assigned by the IEEE Registration Authority, (see http://standards.ieee.org/develop/regauth/oui/), to the manufacturer. The low 40-bits of the EUI are the *extension identifier* uniquely chosen by the manufacturer for each device manufactured and never repeated. The resultant EUI is a globally unique identifier. It is expected that manufacturers who need to comply with this requirement will register with the IEEE Registration Authority and generate and maintain their own EUI *extension identifier* numbering space to ensure its uniqueness for every device made.

Manufacturers may store the EUI externally to the QM33100 or as an alternative the QM33100 has a one-time-programmable memory area that may be programmed with the EUI during product manufacturing. Please refer to section *7.3 – Using the on-chip memory* for details of programming values into OTP.

If needed the host can read the value from the OTP and program it into this EUI register.

Certain IEEE802.15.4 standard [1] defined frames use a 64-bit source address. The software (MAC) generating such frames is expected to insert the EUI within the frame before the frame is written to the QM33100's transmit buffer.

The EUI register is used by the Receive Frame Filtering function, see section 5.4 details. When frame filtering is operational the QM33100 decodes each received frame according to the IEEE802.15.4 standard [1] MAC

rules and any 64-bit destination address present must match the EUI register before the frame will be accepted.

The 8-octets of the Extended Unique Identifier may be accessed as a single 8-octet access to the EUI register file starting at index 0. The bytes of the EUI are output/input in the following order:

REG:00:04 – EUI – Extended Unique Identifier									
7	6	5	4	3	2	1	0	**Octet Index**	**Description**
0xHH								0	Bits 7 to 0 of the extension identifier
0xHH								1	Bits 15 to 8 of the extension identifier
0xHH								2	Bits 23 to 16 of the extension identifier
0xHH								3	Bits 31 to 24 of the extension identifier
0xHH								4	Bits 39 to 32 of the extension identifier
0xNN								5	Bits 7 to 0 of the OUI (manufacturer company ID)
0xNN								6	Bits 15 to 8 of the OUI (manufacturer company ID)
0xNN								7	Bits 23 to 16 of the OUI (manufacturer company ID)

The ordering of octets read from the Extended Unique Identifier register is designed to be directly compatible with the octet ordering of the 64-bit source address fields of IEEE802.15.4 standard [1] MAC frames easing the task of inserting it into a frame for transmission.

8.2.2.3 Sub-register 0x00:0C – PAN Identifier and Short Address

ID	Length (octets)	Type	Mnemonic	Description
00:0C	4	RW	PANADR	PAN Identifier and Short Address

Register file: 0x00-0x1 – General configuration registers, sub-register 0x0C of register file 0x00 contains two 16-bit parameters, the *PAN Identifier* and the *Short Address*. When the QM33100 is powered up or reset both the PAN Identifier and the Short Address in this register are reset to the value 0xFFFF. The host software (MAC) should program the appropriate values into this register if it wishes to use the QM33100's receive frame filtering or automatic acknowledgement generation functions.

In an IEEE802.15.4 standard [1] personal area network (PAN), the PAN coordinator node determines the PAN Identifier for the network, and assigns it and short 16-bit addresses to devices (nodes) associating with the PAN. The nodes in the PAN then should (at the MAC layer) use their assigned short address as the source address and include it along with the PAN Identifier in the frames they transmit. When a node receives a frame it should only process those with a destination address and PAN Identifier which matches their assigned node address and network ID.

When the receive frame filtering and automatic acknowledgement functionality is operational the QM33100 decodes each received frame according to the IEEE802.15.4 standard [1] MAC specification and when it determines that a 16-bit destination address is present in the frame, the QM33100 will compare the destination address with the short address value programmed in this register before accepting/acknowledging the frame, and will similarly only accept received frames when the PAN Identifier in the frame matches the PAN Identifier programmed in this register. See sections 5.4 and 5.5 for details of the frame filtering and automatic acknowledgement functionality.

The PANADR register contains the following sub-fields:

REG:00:0C –PANADR – PAN Identifier and Short Address																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAN_ID																SHORTADDR															
0xFFFF																0xFFFF															

The host software (MAC) only needs to program this register if it is using the QM33100's receive frame filtering and automatic acknowledgement generation functions. The sub-fields are:

Field	Description of fields within Sub-register 0x00:0C – PAN Identifier and Short Address
SHORTADDR reg:00:0C bits:15–0	Short Address. The host software needs to program this register if it is using the QM33100's receive frame filtering functionality, with or without the automatic acknowledgement generation function. The short address is typically assigned to a node by the coordinator function at MAC (or higher) layer as part of network association. The value may alternatively be pre-defined in a closed network where the network association phase is being skipped.
PAN_ID reg:00:0C bits:31–16	PAN Identifier. The host software needs to program this register if it is using the QM33100's receive frame filtering functionality, with or without the automatic acknowledgement generation function. The PAN ID is typically assigned as part of network association. A predefined PAN ID might be used in a closed network where the network association phase is being skipped.

8.2.2.4 Sub-register 0x00:10 – System configuration

ID	Length (octets)	Type	Mnemonic	Description
00:10	4	RW	SYS_CFG	System configuration bitmap

Register file: 0x00-0x1 – General configuration registers, sub-register 0x10 of register file 0x00 is the system configuration register. This is a bitmapped register. Each bit field is separately identified and described below. The SYS_CFG register contains the following bitmapped sub-fields:

REG:00:10 – SYS_CFG – System Configuration bit map																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	ARM_ROM_OPT				CIA_IRQ_ONLY	COEX_IN_POL	COEX_OUT_MODE		COEX_IN_MODE		-	FCS_TYPE	FAST_AAT	PDOA_MODE		CP_SDC	-	CP_SPC		AUTO_ACK	RXAUTR	RXWTOE	CIA_STS	CIA_IPATOV	SPI_CRCEN	PHR_6M8	PHR_MODE	DIS_DRXB	DIS_FCE	DIS_FCS_TX	FFEN
0	0	0	0	0	0	0	0	0	0	0	0	0	1			0	0	0	1	0	0	0	1	1	0	0	0	1	0	0	0

The fields of the SYS_CFG register identified above are individually described below:

Field	Description of fields within Sub-register 0x00:10 – System configuration
FFEN reg:00:10 bit:0	Frame Filtering Enable. This bit enables the frame filtering functionality in the QM33100 receiver. The frame filtering is designed to follow the rules defined in the IEEE802.15.4 standard [1]. When frame filtering is enabled receive frames must pass the frame filtering rules before being considered as a good frame. This includes the destination address matching the PAN_ID and SHORTADDR as defined in *Sub-register 0x00:0C – PAN Identifier and Short Address*, or the long 64-bit defined by *Sub-register 0x00:04 – Extended Unique Identifier*. These addresses and the other frame filtering control bits of *Sub-register 0x00:14 – Frame filter configuration* should be configured correctly before enabling frame filtering with this FFEN bit. Section 5.4 describes frame filtering in more detail.
DIS_FCS_TX reg:00:10 bit:1	Disable auto-FCS Transmission. If this bit is not set then the QM33100 automatically calculates and appends the two Frame-Check-Sequence bytes at the end of each transmitted frame (packet configurations 0, 1,and 2, see Figure 13: Packet formats). Normally DIS_FCS_TX is not set and when transmission is started the QM33100 calculates the FCS on the octets fetched from the TX buffer, and automatically appends the two-byte FCS sequence at the end of the frame. The FCS sequence follows the IEEE802.15.4 standard [1] polynomial, $x^{16} + x^{12} + x^5 + 1$, also known as CRC-16-CCITT or CRC-16 ITU-T. When DIS_FCS_TX is set the QM33100 will not append the FCS to the data frame but instead fetches the two bytes from the TX buffer. The frame length is determined by the TXFLEN field of *Sub-register 0x00:20 – Transmit frame control*. Thus, when DIS_FCS_TX is clear, TXFLEN -2 (frame length minus two) octets are fetched and sent from the TX buffer, and the final two octets sent are the automatically generated FCS bytes. And, when DIS_FCS_TX is set, TXFLEN (frame length) octets are sent from the TX buffer. DIS_FCS_TX may be of use if a non-standard IEEE802.15.4 [1] frame protocol is being employed, and can also be of use to induce a FCS error in the remote receiver during testing.
DIS_FCE reg:00:10 bit:2	Disable frame check error handling. This might be of use for protocols using a different encoding scheme for error handling not based on one defined in IEEE802.15.4 standard [1], but for normal IEEE802.15.4 operation this bit should be set to zero. Setting this bit to one makes the QM33100 treat the frame as valid, ignoring errors in the CRC frame check sequence.
DIS_DRXB reg:00:10 bit:3	Disable Double RX Buffer. The QM33100 has a double buffered receiver allowing reception of a new frame to proceed in one buffer while the host processor is in the process of unloading the last frame received into the other buffer of the buffer pair. The double buffering is enabled when DIS_DRXB is set to 0, and disabled when DIS_DRXB is set to 1. More details on the operation of double buffering are given in section 4.4.
PHR_MODE reg:00:10 bit:4	This configuration allows selection of PHR type to be one of two options. The default setting gives IEEE 802.15.4 standard [1] PHR encoding and a maximum data payload of 127 octets. The other option enables the long frames mode which allows a data payload of up to 1023 octets. In the latter mode the PHR encoding does not follow the IEEE standard. For successful communications between two nodes both must be configured for the same PHR mode. Supported PHR_MODE configurations are: 0 – Standard Frame mode as per IEEE802.15.4 standard [1]. 1 – Long Frame mode encoding as per IEEE802.15.8 standard [4], 0-1023.
PHR_6M8 reg:00:10 bit:5	This configuration sets the PHR rate to match the data rate when 6.81 Mb/s data rate is used. If this is set to 0, then the PHR will be sent at 850 kb/s. This bit is ignored if other data rate (i.e. 850 kb/s) is selected.

Field	Description of fields within Sub-register 0x00:10 – System configuration
SPI_CRCEN reg:00:10 bit:6	Enable SPI CRC functionality. When this bit is set it enables a Cyclic Redundancy Check for SPI read and write transactions, where SPI write access are assumed to have a CRC byte appended that the IC automatically checks and flags any mismatch (error) in the SPICRCERR, event status bit, and SPI read accesses have a CRC computed internally and available in the SPI_RD_CRC register. The host can then check it against the one it calculates for the read data. For more details please refer to see § *1.1.1 –*
CIA_IPATOV reg:00:10 bit:7	Select CIA processing of the preamble CIR. When CIA_IPATOV is 1 (default) it configures the CIA algorithm to analyse the preamble CIR and compute an RX timestamp estimate from it. Upon receipt of aframe, the CIA will be automatically run, assuming CIARUNE in *Sub-register 0x11:08 – Sequencing control* is set to 1 (default). This CIA analysis is started upon SFD detection unless an STS is expected, see *§ 6 – Secure ranging / timestamping*, in which case this processing is delayed until after the STS has been received. When CIA_IPATOV is 0, the CIA algorithm will not run on the preamble CIR. This might be done to save power or processing time when using the STS. If CIARUNE is 0, the CIA_RUN bit, in the DIAG_TMC register, may be used to run the CIA after the packet is received. The RX timestamp estimate resulting from running the CIA on the preamble CIR is written to the IP_TOA field in *Sub-register 0x0C:00 – Preamble receive time stamp and status*, and also to the RX_STAMP field in *Sub-register 0x00:60 – Receive time stamp* but in packet configurations containing an STS, (and when CIA_STS is set), this RX_STAMP value in may be overwritten by the STS based RX timestamp estimate.
CIA_STS reg:00:10 bit:8	Select CIA processing of the STS CIR. For more details of this see *§ 6 – Secure ranging / timestamping*. This CIA_STS configuration bit applies when STS is included in the packet as configured via the CP_SPC configuration. In that case when CIA_STS is 1 (default) it configures the CIA algorithm to analyse the STS CIR and compute an RX timestamp estimate from it. Upon receipt of a packet, the CIA will be automatically run, assuming CIARUNE in *Sub-register 0x11:08 – Sequencing control* is set to 1 (default). This will occur after the STS part of the packet has been received and after the CIA has finished analysis of the preamble sequence based CIR if this is enabled by the CIA_IPATOV configuration bit above. When CIA_STS is 0, the CIA will not analyse the STS CIR. If CIARUNE is 0, the CIA_RUN bit, in DIAG_TMC register, may be used to run the CIA after the packet is received. The RX timestamp estimate resulting from running the CIA on the STS CIR is written to the STS_TOA field in *Sub-register 0x0C:08 – STS receive time stamp and status*, and also to the RX_STAMP field in *Sub-register 0x00:60 – Receive time stamp*.
RXWTOE reg:00:10 bit:9	Receive Wait Timeout Enable. When set RX Enable will initialise an RX_FWTO down count which will disable the receiver if no valid frame is received before the timeout occurs. The timeout period is set in *Sub-register 0x00:34 – Receive frame wait timeout period*. The occurrence of the timeout is signalled by the RXFTO event status bit in *Sub-register 0x00:44 – System event status*.

Field	Description of fields within Sub-register 0x00:10 – System configuration
RXAUTR reg:00:10 bit:10	Receiver Auto-Re-enable. This bit is used to cause the receiver to re-enable automatically. Its operation changes depending on whether the IC is operating in single or double buffered modes. The default value is 0. With this setting, the IC will not automatically re-enable the receiver but will stop receiving and return to idle mode whenever any receive events happen. This includes receiving a frame but also failing to receive a frame because of some error condition, for example an error in the PHY header (as reported by the RXPHE event status bit in Sub-register 0x00:44 – System event status). In such cases if the host wants to re-enable the receiver it must do it explicitly, using CMD_RX command. The operation when RXAUTR = 1 is as follows: (a) Double-buffered mode:After a frame reception event or failure (except a frame wait timeout), the receiver will re-enable to receive another frame.. (b) Single-buffered mode: After a frame reception failure (except a frame wait timeout), the receiver will re-enable to re-attempt reception. For more information on frame reception see section 4 Message reception. **Note:** In double-buffered mode when automatic frame acknowledgement is enabled (by the AUTO_ACK bit below) the receiver will be re-enabled after the ACK frame has been transmitted.
AUTO_ACK reg:00:10 bit:11	Automatic Acknowledgement Enable. Default value 0. This is the enable for the automatic acknowledgement function. See section 5.5 Automatic acknowledgment for details of this operation.
CP_SPC reg:00:10 bits:13-12	STS packet configuration. This field applies to both transmitter and receiver. The IC supports four different STS packet configuration formats (as shown in Figure 13): 0: SP0 – No STS in the packet 1: SP1 – The STS is between the SFD and the PHR. 2: SP2 – The STS is after the data, (i.e. at the end of the packet). 3: SP3 – The STS is after the SFD but there is no PHR or data. For more details on the STS please refer to *§ 6 – Secure ranging / timestamping* and also *§* 3.1 Basic transmission. **N.B. When CP_SPC is 2, for correct operation it is required that the data length is non-zero. This parameter must be set before CP_LOADIV is asserted.**
- reg:00:10 bit:14	This bit is reserved.
CP_SDC reg:00:10 bit:15	This bit, when set, configures the Super Deterministic Code (SDC) mode. If RX timestamp security is not a concern, then overhead of key management and counter control is undesirable. For this reason, the QM33100 includes a special STS mode that uses a code optimized for ToA performance. When this is set the STS KEY/IV values are ignored. The STS generation is using a pre-programmed sequence in the transmitter and the receiver. When this bit is clear, standard IEEE 802.15.4z counter mode is used.

Field	Description of fields within Sub-register 0x00:10 – System configuration
PDOA_MODE reg:00:10 bits:17-16	PDoA mode is configured with this setting: 0x0: this disables the PDoA operation in the QM33100 receiver 0x1: this configures the receiver for PDoA Mode 1 0x2: this configuration is reserved/not supported 0x3: this configures the receiver for PDoA Mode 3 These are further described in 4.2 TDoA and PDoA support
FAST_AAT reg:00:10 bit:18	Enable fast RX to TX turn around mode. When this bit is set the receiver will not wait for CIADONE before it signals end of reception (RXFR). The host can then start processing the received frame. If AUTO_ACK has been configured the AAT bit will also be set to signal ACK is being sent, assuming it has been requested and other frame filtering rules pass.
FCS_TYPE reg:00:10 bit:19	Configure the FCS type. 0= 4z standard 16 bit CRC. 1 = Non-standatd 32 CRC.
COEX_IN_MODE reg:00:10 bit:22-21	Coexistence mode select on what action to take if coex input is asserted. 00 or 10 = Do nothing 01 = Soft Reset 11 = TXRX off.
COEX_OUT_MODE reg:00:10 bits:24–23	Coexistence output mode: The COEX_OUT_MODE field controls the operation of the GPIO5 (or GPIO4) pin when it is configured for use as an output for coexistence signaling. In the case the operation selected by the COEX_OUT_MODE field is: 0b00 – the output is driven high when the transmitter is active, and low otherwise. 0b01 – the output is driven high when the receiver is active, and low otherwise. 0b10 – the output is driven high when either TX or RX is active, and low otherwise. 0b11 – the output is driven high when a timer trigger occurs, i.e., either TIMER0 event occurs (when TIMER0_COEX = 1) or TIMER1 event occurs (when TIMER1_COEX = 1), in this case it is up to the host software to turn off the output at the appropriate time, (e.g. by setting the GPIO to be an output at the logic 0 level). NOTE: This COEX_OUT_MODE field needs to be cleared/set to 0, prior to configuring GPIO operation (e.g. configuring GPIO5 (or GPIO4) for COEX_OUT mode (see MSGP5). Otherwise the GPIO sill be set high immediately upon writing COEX_OUT_MODE field to 0x3.
COEX_IN_POL reg:00:10 bit:25	Coex input polarity 0 = active high to flag coex issue. Note: Set the MFIO mode first or this may cause a coex event to trigger.
CIA_IRQ_ONLY reg:00:10 bit:26	Mask the off chip interrupt lines and only allow the ARM to be interrupted
ARM_ROM_OPT reg:00:10 bit:30-27	General purpose bit to pass info to the ARM ROM.
- reg:00:10 bits:various	These bits are reserved.

8.2.2.5 Sub-register 0x00:14 – Frame filter configuration

ID	Length (octets)	Type	Mnemonic	Description
00:14	2	RW	FF_CFG	Frame filter configuration bitmap

Register file: 0x00-0x1 – General configuration registers, sub-register 0x14 of register file 0x00 is the IEEE802.15.4 standard [1] MAC frame address filter configuration register. This is a bitmapped register. Each bit field is separately identified and described below. The FF_CFG register contains the following bitmapped sub-fields:

REG:00:14 – FF_CFG – Frame Filter Configuration bit map

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																LSADRAPE	SSADRAPE	LE3_PEND	LE2_PEND	LE1_PEND	LE0_PEND	FFIB	FFBC	FFAE	FFAF	FFAMULTI	FFAR	FFAM	FFAA	FFAD	FFAB
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Note: For any of these bits to apply FFEN bit in SYS_CFG must also be set.

Definition of the bit fields within FF_CFG bitmap: -

Field	Description of fields within Sub-register 0x00:14 – Frame filter configuration
FFAB reg:00:14 bit:0	Frame Filtering Allow Beacon frame reception. IEEE802.15.4 standard [1]. Frames begin with three bits, indicating the frame type, b3 to b0. For beacon frames these are binary 000. When FFAB is set to 1 beacon frames will be accepted (assuming all other frame filtering rules are passed) and when FFAB is clear beacon frames will be ignored. Section 5.4 describes frame filtering in detail.
FFAD reg:00:14 bit:1	Frame Filtering Allow Data frame reception. IEEE802.15.4 standard [1] frames begin with three bits, indicating the frame type, b3 to b0. For data frames these are binary 001. When FFAD is set to 1 data frames will be accepted (assuming all other frame filtering rules are passed) and when FFAD is clear data frames will be ignored. Section 5.4 describes frame filtering in more detail.
FFAA reg:00:14 bit:2	Frame Filtering Allow Acknowledgment frame reception. IEEE802.15.4 standard [1] frames begin with three bits, indicating the frame type, b3 to b0. For acknowledgment frames these are binary 010. When FFAA is set to 1 acknowledgment frames will be accepted (assuming all other frame filtering rules are passed) and when FFAA is clear acknowledgment frames will be ignored. Section 5.4 describes frame filtering in more detail.
FFAM reg:00:14 bit:3	Frame Filtering Allow MAC command frame reception. IEEE802.15.4 standard [1] frames begin with three bits, indicating the frame type, b3 to b0. For MAC command frames these are binary 011. When FFAM is set to 1 MAC command frames will be accepted (assuming all other frame filtering rules are passed) and when FFAM is clear MAC command frames will be ignored. Section 5.4 describes frame filtering in more detail.

Field	Description of fields within Sub-register 0x00:14 – Frame filter configuration
FFAR reg:00:14 bit:4	Frame Filtering Allow Reserved frame types. IEEE802.15.4 standard [1] frames begin with three bits, indicating the frame type, b3 to b0. For reserved frames these are binary 100 and would normally be rejected. When FFAR is set to 1 these frames are accepted. Since these frame types are unknown no further frame decoding is done (e.g. no address matching etc.) and the software will thus be responsible for validating and interpreting these frames. Section 5.4 describes frame filtering in more detail. Note that the frame filter does decode the frame control fields to determine the minimum length of the expected frame and will reject the frame if it is too short, see section 5.4.
FFAMULTI reg:00:14 bit:5	Frame Filtering Allow Multipurpose frames. IEEE802.15.4 standard [1] frames begin with three bits, indicating the frame type, b3 to b0. For the multipurpose frames these are binary 101. When FFAMULTI is set to 1 these frames are accepted. When FFAMULTI is set to 0, they will be ignored. Section 5.4 describes frame filtering in more detail. Note that the frame filter does decode the frame control fields to determine the minimum length of the expected frame and will reject the frame if it is too short, see section 5.4.
FFAF reg:00:14 bit:6	Frame Filtering Allow Fragmented/Frack frames. IEEE802.15.4 standard [1] frames begin with three bits, indicating the frame type, b3 to b0. For the fragmented type these are binary 110. When FFAF is set to 1 these frames are accepted. When FFAF is set to 0, they will be ignored. Section 5.4 describes frame filtering in more detail. Note that the frame filter does decode the frame control fields to determine the minimum length of the expected frame and will reject the frame if it is too short, see section 5.4.
FFAE reg:00:14 bit:7	Frame Filtering Allow Extended frames. IEEE802.15.4 standard [1] frames begin with three bits, indicating the frame type, b3 to b0. For the extended frames these are binary 111. When FFAE is set to 1 these frames are accepted. When FFAE is set to 0, they will be ignored. Section 5.4 describes frame filtering in more detail. Note that the frame filter does decode the frame control fields to determine the minimum length of the expected frame and will reject the frame if it is too short, see section 5.4.
FFBC reg:00:14 bit:8	Frame Filtering Behave as PAN Coordinator. When this bit is set (to 1) the device behaves as a PAN coordinator, which means that it will accept frames with only source addressing fields and: a) Source PAN ID must match PAN_ID set, for MAC and data frame types b) Destination PAN ID must match PAN_ID set, for Multipurpose frame type Section 5.4 describes frame filtering in more detail.
FFIB reg:00:14 bit:9	Frame Filtering allow MAC Implicit Broadcast. If this bit is not set (set to 0), then the destination addressing (PAN ID and destination address fields, as appropriate, in the received frame, if present) must either be set to broadcast (0xFFFF) or specific destination address, otherwise the frame will be rejected. If this bit is set to 1, then frames without destination PAN ID and destination address are treated as though they are addressed to the broadcast PAN ID and broadcast short (16-bit) address. Section 5.4 describes frame filtering in more detail.

Field	Description of fields within Sub-register 0x00:14 – Frame filter configuration
LE0_PEND reg:00:14 bit:10	Data pending for device at LE0 address When this is set and the source address of a received MAC command Data Request frame matches the address set in LE_ADDR0 in LE_PEND_01 register, the automatically transmitted ACK will have PEND bit set. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail.
LE1_PEND reg:00:14 bit:11	Data pending for device at LE1 address When this is set and the source address of a received MAC command Data Request frame matches the address set in LE_ADDR1 in LE_PEND_01 register, the automatically transmitted ACK will have PEND bit set. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail.
LE2_PEND reg:00:14 bit:12	Data pending for device at LE2 address When this is set and the source address of a received MAC command Data Request frame matches the address set in LE_ADDR2 in LE_PEND_23 register, the automatically transmitted ACK will have PEND bit set. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail.
LE3_PEND reg:00:14 bit:13	Data pending for device at LE3 address When this is set and the source address of a received MAC command Data Request frame matches the address set in LE_ADDR3 in LE_PEND_23 register, the automatically transmitted ACK will have PEND bit set. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail.
SSADRAPE reg:00:14 bit:14	Short Source Address Data Request ACK with PEND Enable. Setting this bit to 1 means that auto-ACK that is sent as a reply to any MAC command Data Request coming from any node with a short (16-bit) source address will have the PEND bit set. Setting this bit to 0, means that an auto-ACK to a MAC command from a node with short source address will have the PEND bit set if and only if the address matches one of the four 16-bit addresses programmed in LE_PEND_01 and LE_PEND_23 registers and a matching bit in LE0_PEND, LE1_PEND, LE2_PEND and LE3_PEND. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail.
LSADRAPE reg:00:14 bit:15	Long Source Address Data Request ACK with PEND Enable Setting this bit to 1 means that auto-ACK that is sent as a reply to any MAC command Data Request coming from any node with a long (64-bit) source address will have the PEND bit set. Setting this bit to 0, means that an auto-ACK to a MAC command from a node with long source address will not have the PEND bit set. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail.

8.2.2.6 Sub-register 0x00:18 – SPI CRC read status

ID	Length (octets)	Type	Mnemonic	Description
00:18	1	RO	SPI_RD_CRC	SPI CRC read status

Register file: 0x00-0x1 – General configuration registers, sub-register 0x18 of register file 0x00 is the status register for the SPI CRC function. This register contains the SPI read CRC value. This is the CRC value resulting from each SPI read when SPI CRC mode is enabled via the SPI_CRCEN bit in *Sub-register 0x00:10 – System configuration*. For more details of SPI CRC operation please refer to § *1.1.1 –* .

8.2.2.7 Sub-register 0x00:1C – System time counter

ID	Length (octets)	Type	Mnemonic	Description
00:1C	4	RO	SYS_TIME	System time counter (32-bit)

Register file: 0x00-0x1 – General configuration registers, sub-register 0x1C of register file 0x00 is the System Time Counter register. The device's system time and time stamps are designed to be based on the time units which are nominally at 64 GHz, or more precisely 499.2 MHz × 128 = 63.8976 GHz.

The SYS_TIME register only counts in the **IDLE_PLL**, **RX** and **TX** states (were the digital PLL enabled), and it only contains the 32 most significant bits of the device's system time. In all other states the system time counter is disabled and this register is not updated. When it is active the SYS_TIME register is incremented at a rate of 125 MHz and in units of 2. The least significant bit of SYS_TIME is always zero. The internal device's counter wrap period is $2^{40} \div (128 \times 499.2\ \text{MHz}) = 17.2074$ seconds.

8.2.2.8 Sub-register 0x00:20 – Transmit frame control

ID	Length (octets)	Type	Mnemonic	Description
00:24	6	RW	TX_FCTRL	Transmit frame control

Register file: 0x00-0x1 – General configuration registers, sub-register 0x20 of register file 0x00, the transmit frame control register, contains a number of TX control fields. Each field is separately identified and described below. (For a general discussion of transmission please refer to section 3 Message transmission)

REG:00:20 – TX_FCTRL – Transmit frame control (Octets 0 to 3, 32-bits)																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-						TXB_OFFSET										TXPSR				TR	TXBR	TXFLEN									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	1	0	0

REG:00:24 – TX_FCTRL – Transmit frame control (Octets 4 to 5, 16-bits)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																FINE_PLEN								-							
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the TX_FCTRL register identified above are individually described below:

Field	Description of fields within Sub-register 0x00:20 – Transmit frame control
TXFLEN reg:00:20 bits:9–0	Transmit Frame Length. The QM33100 supports data frame lengths up to 1023 bytes. IEEE802.15.4 standard [1] frames can be up to 127 bytes long. The extended frame mode is enabled via the PHR_MODE selection bits of *Sub-register 0x00:10 – System configuration*, please refer to section **3.4 –Extended length data frames** for more details of this IEEE802.15.8 standard [4] mode. The value specified by TXFLEN determines the length of the data portion of the transmitted frame. This length includes the two-octet CRC appended automatically at the end of the frame, unless DIS_FCS_TX (in Sub-register 0x00:10 – System configuration) is use to suppress the FCS. The frame length is also copied into the PHY Header of the TX frame so that the receiving device knows how much data to decode.
TXBR reg:00:20 bit:10	Transmit Bit Rate. This sets the bit rate for the data portion of the frame, if it is set to 0 the data is sent at 850 kb/s and if set to 1 at 6.81 Mb/s.
TR reg:00:20 bit: 11	Transmit Ranging enable. This bit has no operational effect on the QM33100; however it is copied into the ranging bit in the PHY header (PHR) of the transmitted packet, identifying the frame as a ranging frame. In some receiver implementations this may be used to enable hardware or software associated with time-stamping the frame. In the QM33100 receiver the time-stamping of the receive frame does not depend or use the ranging bit in the received PHR.

<table>
<tr><th>Field</th><th>Description of fields within Sub-register 0x00:20 – Transmit frame control</th></tr>
<tr><td>TXPSR

reg:00:20
bits:15-12</td><td>Transmit Preamble Symbol Repetitions (PSR). This sets the length of the transmitted preamble sequence in symbols. Each preamble symbol is approximately 1 µs in duration[3].The two TXPSR bits are copied into the PHY Header. The receiving end is thus made aware of how much preamble was sent. This might inform its choice of preamble length in any reply message. There are four standard preamble lengths defined in the 802.15.4 UWB PHY – these are 16, 64, 1024 and 4096 symbols. The QM33100 has facility via this register setting to to send preambles of additional (non-standard) intermediary lengths. Table 21 below lists the selectable preamble lengths.

Table 21: Preamble length selection

<table>
<tr><th>Bit 15</th><th>Bit 14</th><th>Bit 13</th><th>Bit 12</th><th></th></tr>
<tr><th colspan="4">TXPSR</th><th>Preamble Length</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>64</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1024</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>4096</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>32</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>128</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>1536</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>256</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>2048</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>512</td></tr>
</table>
The bit numbers quoted above are the bit numbers in the TX_FCTRL register.

Note: The FINE_PLEN field below gives additional control of the preamble length (PSR). The choice of preamble length has a bearing on operating range and system performance. The TXPSR bit values not defined in the table above are reserved and should not be used.
When auto ACK feature is used, see AUTO_ACK, the ACK frame will have its PSR set based on this field.</td></tr>
<tr><td>TXB_OFFSET

reg:00:20
bits:25–16</td><td>Transmit buffer index offset. This 10-bit field is used to specify an index in the transmit buffer of the first octet to be transmitted. The TX frame begins with the octet at the TXB_OFFSET index and continues for the number octets specified by the frame length (TXFLEN) less 2 for the CRC. Care should be taken that the TXB_OFFSET plus the frame length does not extend past the end of the TX_BUFFER.</td></tr>
<tr><td>-

reg:00:20, 00:24
bits: various</td><td>These bits are reserved.</td></tr>
</table>

[3] The duration of preamble symbols is 993.59 ns with the 16 MHz PRF setting and 1017.63 ns with for 64 MHz PRF.

Field	Description of fields within Sub-register 0x00:20 – Transmit frame control
FINE_PLEN reg:00:24 bits:15–8	Fine PSR control. This field may be used to get fine control of the length of the transmitted preamble sequence, by specifying the number of preamble symbol repetitions in units of 8 symbols. When FINE_PLEN is zero the preamble length configured in the TXPSR field will be used. When FINE_PLEN is non-zero the preamble length is given by the expression: 8 x (FINE_PLEN + 1), i.e. a FINE_PLEN value of 4 gives a PSR of 40 symbols, and the maximum FINE_PLEN value 0xFF gives a PSR of 2048 symbols. This field allows for tuning the preamble length to trade off message on-air time (and resultant power consumption) versus operational performance to match particular use cases.

8.2.2.9 Sub-register 0x00:28 – Delayed send or receive time

ID	Length (octets)	Type	Mnemonic	Description
00:28	4	RW	DX_TIME	Delayed send or receive time (32-bit)

Register file: 0x00-0x1 – General configuration registers and AES, sub-register 0x28 of register file 0x00, the Delayed Send or Receive Time, is used to specify a time in the future to either turn on the receiver to be ready to receive a frame, or to turn on the transmitter and send a frame. The units are one half of the 499.2 MHz fundamental frequency, (~ 4 ns). The least significant bit of this register is ignored, i.e. the smallest value that can be specified is 2, i.e. 8 ns. Delayed send can be initiated by any of the following commands: CMD_DTX , CMD_DTX_TS, CMD_DTX_RS, CMD_DTX_W4R, CMD_DTX_TS_W4R or CMD_DTX_RS_W4R similarly the delayed receive can be initiated by any of the following commands: CMD_DRX, CMD_DRX_TS or CMD_DRX_RS. For more information see sections 3.3 Delayed transmission and 4.3 Delayed receive

8.2.2.10 Sub-register 0x00:30 – Delayed send or receive reference time

ID	Length (octets)	Type	Mnemonic	Description
00:30	4	RW	DREF_TIME	Delayed send or receive reference time (32-bit)

Register file: 0x00-0x1 – General configuration registers and AES, sub-register 0x30 of register file 0x00, the Delayed Send or Receive Reference Time, is used to specify a time (at which an an event happened, e.g. Beacon was sent) and any value in DX_TIME is added to this register before either the receiver or transmitter are turned on. The unit is one half of the 499.2 MHz fundamental frequency, (~ 4 ns). The least significant bit of this register is ignored, i.e. the smallest value that can be specified is 2, i.e. ~ 8 ns. Delayed send with respect to reference time is initiated by CMD_DTX_REF or CMD_DTX_REF_W4R. Delayed receive with respect to reference time is initiated by CMD_DRX_REF. For more information see section 3.3 Delayed transmission and 4.3 Delayed receive.

8.2.2.11 Sub-register 0x00:34 – Receive frame wait timeout period

ID	Length (octets)	Type	Mnemonic	Description
00:34	3	RW	RX_FWTO	Receive frame wait timeout period

Register file: 0x00-0x1 – General configuration registers, sub-register 0x34 of register file 0x00 is the receive frame wait timeout period. It is a 20-bit wide register with a unit of 512/499.2 MHZ (~ 1.0256 µs). The receive frame wait timeout function is provided to allow the host processor to enter a low power state awaiting a valid receive frame and be woken up by the QM33100 when either a frame is received or the programmed timeout has elapsed. While manymicrocontrollers have timers that might be used for this purpose, including this RX timeout functionality in the QM33100 allows additional flexibility to the system designer in selecting the microprocessor to optimise the solution. The frame wait timeout is enabled by the RXWTOE in *Sub-register 0x00:10 – System configuration*.

When the receiver is enabled (and begins hunting for the preamble sequence) and RXWTOE is set, then the frame wait timeout counter starts counting the timeout period programmed. Thereafter, assuming no action is taken to change the operation, one of two things should happen:

a) The Receive Frame Wait Timeout period elapses. This disables the receiver and sets the RXFTO (Receiver Frame Wait Timeout) bit in the status register, (and resets the counter).
b) A valid receive frame arrives and sets the RXFR and RXFCG bits in the status register. This stops the receive frame wait timer counter so RXFTO will not be set.
c) Host can issue transceiver off command (CMD_TRXOFF) at anytime to stop the RX and disable RXFTO.

The RX frame wait timeout period should only be programmed when the IC is in **IDLE_PLL** state, before the receiver is enabled. Programming the RXFTO at other times (e.g. in **RX** state) is not prevented but may result in unpredictable behaviour.

8.2.2.12 Sub-register 0x00:3C – System event enable mask

ID	Length (octets)	Type	Mnemonic	Description
00:3C	6	RW	SYS_ENABLE	System event enable mask register

Register file: 0x00-0x1 – General configuration registers, sub-register 0x3C of register file 0x00 is the system event mask register. These are aligned with the event status bits in the SYS_STATUS register. Whenever a bit in the SYS_ENABLE is set (to 1) and the corresponding bit in the SYS_STATUS register is also set, then an interrupt will be generated asserting the hardware IRQ output line. The interrupt condition may be removed by clearing the corresponding bit in this SYS_ENABLE register (by setting it to 0) or by clearing the corresponding latched bit in the SYS_STATUS register (generally by writing a 1 to the bit – please refer to individual SYS_STATUS register bit definitions for details).

The SYS_ENABLE register contains the system event mask bits identified and described below:

REG:00:3C – SYS_ENABLE – system event enable mask (octets 0 to 3)																												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER1_EN	TIMER0_EN	ARFE_EN	CPERR_EN	HPDWARN_EN	RXSTO_EN	PLL_HILO_EN	RCINIT_EN	SPIRDY_EN	-	RXPTO_EN	RXOVRR_EN	VWARN_EN	CIAERR_EN	RXFTO_EN	RXRFSL_EN	RXFCE_EN	RXFCG_EN	RXFR_EN	RXPHE_EN	RXPHD_EN	CIADONE_EN	RXSFDD_EN	RXPRD_EN	TXFRS_EN	TXPHS_EN	TXPRS_EN	TXFRB_EN	AAT_EN	SPICRCE_EN	CPLOCK_EN	-
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REG:00:40 – SYS_ENABLE – system event enable mask (octets 4 and 5)																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RXSTS_EN	TXSTS_EN	-	COEX_CLR_EN	COEX_ERR_EN	CCA_FAIL_EN	SPIERR_EN	SPI_UNF_EN	SPI_OVF_EN	CMD_ERR_EN	AES_ERR_EN	AES_DONE_EN	GPIOIRQ_EN	VT_DET_EN	-	PGFCAL_ERR_EN	RXPREJ_EN	-
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

The system event mask bits of the SYS_ENABLE register identified above are individually described below:

Field	Description of fields within Sub-register 0x00:3C – System event enable mask
– reg:00:3C	Bits marked with '-' are reserved.
CPLOCK_EN reg:00:3C bit:1	Mask clock PLL lock event. When CPLOCK_EN is 0 the CPLOCK event status bit will not generate an interrupt. When CPLOCK_EN is 1 and the CPLOCK event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
SPICRCE_EN reg:00:3C bit:2	Mask SPI CRC Error event. When SPICRCE_EN is 0 the SPICRCE event status bit will not generate an interrupt. When SPICRCE_EN is 1 and the SPICRCE event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
AAT_EN reg:00:3C bit:3	Mask automatic acknowledge trigger event. When AAT_EN is 0 the AAT event status bit will not generate an interrupt. When AAT_EN is 1 and the AAT event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt. **AAT should be masked when the automatic acknowledge is not enabled so that spurious interrupts cannot affect system behaviour.**
TXFRB_EN reg:00:3C bit:4	Mask transmit frame begins event. When TXFRB_EN is 0 the TXFRB event status bit will not generate an interrupt. When TXFRB_EN is 1 and the TXFRB event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
TXPRS_EN reg:00:3C bit:5	Mask transmit preamble sent event. When TXPRS_EN is 0 the TXPRS event status bit will not generate an interrupt. When TXPRS_EN is 1 and the TXPRS event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.

Field	Description of fields within Sub-register 0x00:3C – System event enable mask
TXPHS_EN reg:00:3C bit:6	Mask transmit PHY Header Sent event. When TXPHS_EN is 0 the TXPHS event status bit will not generate an interrupt. When TXPHS_EN is 1 and the TXPHS event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
TXFRS_EN reg:00:3C bit:7	Mask transmit frame sent event. When TXFRS_EN is 0 the TXFRS event status bit will not generate an interrupt. When TXFRS_EN is 1 and the TXFRS event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXPRD_EN reg:00:3C bit:8	Mask receiver preamble detected event. When RXPRD_EN is 0 the RXPRD event status bit will not generate an interrupt. When RXPRD_EN is 1 and the RXPRD event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXSFDD_EN reg:00:3C bit:9	Mask receiver SFD detected event. When RXSFDD_EN is 0 the RXSFDD event status bit will not generate an interrupt. When RXSFDD_EN is 1 and the RXSFDD event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
CIADONE_EN reg:00:3C bit:10	Mask CIA processing done event. When CIADONE_EN is 0 the CIADONE event status bit will not generate an interrupt. When CIADONE_EN is 1 and the CIADONE event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXPHD_EN reg:00:3C bit:11	Mask receiver PHY header detect event. When RXPHD_EN is 0 the RXPHD event status bit will not generate an interrupt. When RXPHD_EN is 1 and the RXPHD event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXPHE_EN reg:00:3C bit:12	Mask receiver PHY header error event. When RXPHE_EN is 0 the RXPHE event status bit will not generate an interrupt. When RXPHE_EN is 1 and the RXPHE event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXFR_EN reg:00:3C bit:13	Mask receiver data frame ready event. When RXFR_EN is 0 the RXFR event status bit will not generate an interrupt. When RXFR_EN is 1 and the RXFR event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXFCG_EN reg:00:3C bit:14	Mask receiver FCS good event. When RXFCG_EN is 0 the RXFCG event status bit will not generate an interrupt. When RXFCG_EN is 1 and the RXFCG event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXFCE_EN reg:00:3C bit:15	Mask receiver FCS error event. When RXFCE_EN is 0 the RXFCE event status bit will not generate an interrupt. When RXFCE_EN is 1 and the RXFCE event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXRFSL_EN reg:00:3C bit:16	Mask receiver Reed Solomon Frame Sync Loss event. When RXRFSL_EN is 0 the RXFSL event status bit will not generate an interrupt. When RXRFSL_EN is 1 and the RXFSL event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXFTO_EN reg:00:3C bit:17	Mask Receive Frame Wait Timeout event. When RXFTO_EN is 0 the RXFTO event status bit will not generate an interrupt. When RXFTO_EN is 1 and the RXFTO event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
CIAERR_EN reg:00:3C bit:18	Mask leading edge detection processing error event. When CIAERR_EN is 0 the CIAERR event status bit will not generate an interrupt. When CIAERR_EN is 1 and the CIAERR event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
VWARN_EN reg:00:3C rxoverbit:19	Mask Voltage warning event. When VWARN_EN is 0 the VWARN event status bit will not generate an interrupt. When VWARN_EN is 1 and the VWARN event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.

Field	Description of fields within Sub-register 0x00:3C – System event enable mask
RXOVRR_EN reg:00:3C bit:20	Receiver overrun. When RXOVRR_EN is 0 the RXOVRR event status bit will not generate an interrupt. When RXOVRR_EN is 1 and the RXOVRR event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt
RXPTO_EN reg:00:3C bit:21	Mask Preamble detection timeout event. When RXPTO_EN is 0 the RXPTO event status bit will not generate an interrupt. When RXPTO_EN is 1 and the RXPTO event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt. Note: Perfroming a soft reset (SOFTRESET) may cause a glitch on the interrupt line when this event is configured to generate interrupt. The interrupt will be triggered, but when the host goes to read the status register (SYS_STATUS) it will be in the reset state as the device has reset.
reg:00:3C bit:22	reserved
SPIRDY_EN reg:00:3C bit:23	Mask SPI ready event. When SPIRDY_EN is 0 the SPIRDY event status bit will not generate an interrupt. When SPIRDY_EN is 1 and the SPIRDY event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt. Figure 8 shows the host interrupt that is generated from the SPIRDY event.
RCINIT_EN reg:00:3C bit:24	Mask IDLE RC event. When RCINIT_EN is 0 the RCINIT event status bit will not generate an interrupt. When RCINIT_EN is 1 and the RCINIT event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt. The RCINIT event will be generated when device enters **IDLE_RC** state as shown in Figure 8.
PLL_HILO_EN reg:00:3C bit:25	Mask PLL Losing Lock warning event. When PLL_HILO_EN is 0 the PLL_HILO event status bit will not generate an interrupt. When PLL_HILO_EN is 1 and the PLL_HILO event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXSTO_EN reg:00:3C bit:26	Mask Receive SFD timeout event. When RXSTO_EN is 0 the RXSTO event status bit will not generate an interrupt. When RXSTO_EN is 1 and the RXSTO event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
HPDWARN_EN reg:00:3C bit:27	Mask Half Period Delay Warning event. When HPDWARN_EN is 0 the HPDWARN event status bit will not generate an interrupt. When HPDWARN_EN is 1 and the HPDWARN event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
CPERR_EN reg:00:3C bit:28	Mask Scramble Timestamp Sequence (STS) error event. When CPERR_EN is 0 the CPERR event status bit will not generate an interrupt. When CPERR_EN is 1 and the CPERR event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
ARFE_EN reg:00:3C bit:29	Mask Automatic Frame Filtering rejection event. When ARFE_EN is 0 the ARFE event status bit will not generate an interrupt. When ARFE_EN is 1 and the ARFE event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
TIMER0_EN reg:00:3C bit:30	Timer 0 event enable/mask. When TIMER0_EN is 0 the TIMER0 event status bit will not generate an interrupt. When TIMER0_EN is 1 and the TIMER0 event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
TIMER1_EN reg:00:3C bit:31	Timer 1 event enable/mask. When TIMER1_EN is 0 the TIMER0 event status bit will not generate an interrupt. When TIMER1_EN is 1 and the TIMER0 event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
RXPREJ_EN reg:00:40 bit:1	Mask Receiver Preamble Rejection event. When RXPREJ_EN is 0 the RXPREJ event status bit will not generate an interrupt. When RXPREJ_EN is 1 and the RXPREJ event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.

Field	Description of fields within Sub-register 0x00:3C – System event enable mask
PGFCAL_ERR_EN reg:00:40 bit:2	PGF/RX calibration error event enable. This event enable is set by default. When PGFCAL_ERR_EN is 1 and the PGFCAL_ERR_EN event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
VT_DET_EN reg:00:40 bit:4	Mask Voltage/Temperature variation dtection interrupt event. When VT_DET_EN is 0 the VT_DET event status bit will not generate an interrupt. When VT_DET_EN is 1 and the VT_DET event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
GPIOIRQ_EN reg:00:40 bit:5	Mask GPIO interrupt event. When GPIOIRQ_ENGPIOIRQ_EN is 0 the GPIOIRQ event status bit will not generate an interrupt. When GPIOIRQ_EN is 1 and the GPIOIRQ event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
AES_DONE_EN reg:00:40 bit:6	Mask AES done interrupt event. When AES_DONE_EN is 0 the AES_DONE event status bit will not generate an interrupt. When AES_DONE_EN is 1 and the AES_DONE event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
AES_ERR_EN reg:00:40 bit:7	Mask AES error interrupt event. When AES_ERR is 0 the AES_ERR event status bit will not generate an interrupt. When AES_ERR is 1 and the AES_ERR event status bit is 1, the hardware IRQ interrupt line will be asserted to generate can interrupt.
CDM_ERR_EN reg:00:40 bit:8	Mask CMD error interrupt event. When CMD_ERR_EN is 0 the CMD_ERR event status bit will not generate an interrupt. When CMD_ERR_EN is 1 and the CMD_ERR event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
SPI_OVF_EN reg:00:40 bit:9	Mask SPI overflow interrupt event. When SPI_OVF_EN is 0 the SPIOVF event status bit will not generate an interrupt. When SPI_OVF_EN is 1 and the SPIOVF event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
SPI_UNF_EN reg:00:40 bit:10	Mask SPI underflow interrupt event. When SPI_UNF_EN is 0 the SPIUNF event status bit will not generate an interrupt. When SPI_UNF_EN is 1 and the SPIUNF event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
SPI_ERR_EN reg:00:40 bit:11	Mask SPI error interrupt event. When SPIERR is 0 the SPIERR event status bit will not generate an interrupt. When SPIERR is 1 and the SPIERR event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
CCA_FAIL_EN reg:00:40 bit:12	Mask CCA fail interrupt event. When CCA_FAIL_EN is 0 the CCA_FAIL event status bit will not generate an interrupt. When CCA_FAIL_EN is 1 and the CCA_FAIL event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
COEX_ERR_EN reg:00:40 bit:13	Coexistence error event interrupt enable mask. When COEX_ERR_EN is 0 the COEX_ERR event status bit will not generate an interrupt. When COEX_ERR_EN is 1 and the COEX_ERR event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
COEX_CLR_EN reg:00:40 bit:14	Coexistence clear event interrupt enable mask. When COEX_CLR_EN is 0 the COEX_CLR event status bit will not generate an interrupt. When COEX_CLR_EN is 1 and the COEX_CLR event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.
TXSTS_EN reg:00:40 bit:16	Transmit STS sent event interrupt enable mask. When TXSTS_EN is 0 the TXSTS event status bit will not generate an interrupt. When TXSTS_EN is 1 and the TXSTS event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.

Field	Description of fields within Sub-register 0x00:3C – System event enable mask
RXSTS_EN reg:00:40 bit:17	Receiver STS received event interrupt enable mask. When RXSTS_EN is 0 the RXSTS event status bit will not generate an interrupt. When RXSTS_EN is 1 and the RXSTS event status bit is 1, the hardware IRQ interrupt line will be asserted to generate an interrupt.

8.2.2.13 Sub-register 0x00:44 – System event status

ID	Length (octets)	Type	Mnemonic	Description
00:44	6	SRW	SYS_STATUS	System event status register

Register file: 0x00-0x1 – General configuration registers, sub-register 0x44 of register file 0x00 is the system event status register, SYS_STATUS. It contains status bits that indicate the occurrence of different system events or status changes. It is possible to enable particular events as interrupt sources, by employing the SYS_ENABLE, *Sub-register 0x00:3C – System event enable* mask, so that the setting of the event status bit will generate an interrupt, asserting the hardware IRQ output line. This can be used, for example, to allow the host processor to enter a low-power state during frame transmission or reception awaiting an interrupt to wake upon the completion of the TX or RX activity.

Reading the SYS_STATUS register returns the state of the status bits. Generally these event status bits are latched so that the event is captured. Such latched bits need to be explicitly cleared by writing '1' to the bit position (writing '0' has no effect).

The SYS_STATUS register contains the system event status bits identified and described below:

REG:00:44 – SYS_STATUS – System status register (octets 0 to 3)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER1	TIMER0	ARFE	CPERR	HPDWARN	RXSTO	PLLHILO	RCINIT	SPIRDY	-	RXPTO	RXOVRR	VWARN	CIAERR	RXFTO	RXFSL	RXFCE	RXFCG	RXFR	RXPHE	RXPHD	CIADONE	RXSFDD	RXPRD	TXFRS	TXPHS	TXPRS	TXFRB	AAT	SPICRCE	CPLOCK	IRQS
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REG:00:48 – SYS_STATUS – System status register (octets 4 and 5)																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RXSTS	TXSTS	-	COEX_CLR	COEX_ERR	CCA_FAIL	SPIERR	SPI_UNF	SPI_OVF	CMD_ERR	AES_ERR	AES_DONE	GPIOIRQ	VT_DET	-	PGFCAL_ERR	RXPREJ	-
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The system event status bits of the SYS_STATUS register identified above are individually described below:

Field	Description of fields within Sub-register 0x00:44 – System Event Status
– reg:00:44 bit:various	Bits marked '-' are reserved and should not be changed.
IRQS reg:00:44 bit:0	Interrupt Request Status. This is a READ ONLY status flag – it cannot be cleared or overwritten. Whenever a status bit in *Sub-register 0x00:44 – System Event Status* is activated (i.e. has a value of 1) and the corresponding bit in *Sub-register 0x00:3C – System Event Enable Mask* is enabled (i.e. has a value of 1 also) then the IRQ interrupt request line will be driven to its active ON level. If there are no active mask enabled status bits then the IRQ interrupt request line will be set to its inactive OFF level. This IRQS flag reflects the overall status of interrupts. If there are any unmasked interrupt sources active then the IRQS bit will be 1 (and IRQ interrupt request line will be at its active ON level) the otherwise IRQS will be zero (and IRQ interrupt request line at its OFF level). The polarity of the IRQ interrupt request line is controllable via the HIRQ_POL configuration bit in *Sub-register 0x0F:28 – Digital Diagnostics Test Mode Control*.
CPLOCK reg:00:44 bit:1	Clock PLL Lock. The CPLOCK event status bit indicates that the digital clock PLL has locked. This may be used as an interrupt to indicate that the QM33100 clock is operating at full speed, after which the SPI can be run at its maximum rate also. The CPLOCK bit is cleared by writing a 1 to it. The clock PLL lock status is also available via the RFREG
SPICRCE reg:00:44 bit:2	SPI CRC Error. When SPI CRC mode (see *2.3.1.3 – SPI CRC Mode*) is enabled (by the SPI_CRCEN bit in SYS_CFG register), the SPICRCE event status bit indicates that a CRC error has been detected during an SPI write. Once set the SPICRCE bit remains set until it is cleared explicitly by writing a 1 to it. When SPI CRC mode is disabled the SPICRCE bit is always set. SPI Write CRC Error events are counted in *Sub-register 0x0F:1A – SPI Write CRC Error Counter*, when counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*.
AAT reg:00:44 bit:3	Automatic Acknowledge Trigger. This status event status bit is set when frame filtering is enabled and a data frame (or MAC command frame) is received (correctly addressed and with a good CRC) with the acknowledgement request bit set in its frame control field. If the automatic acknowledgement is enabled (by the AUTO_ACK bit in *Sub-register 0x00:10 – System Configuration*) then the AAT bit can be used during receive interrupt processing to detect that acknowledgement is in progress and thus avoid taking any action until the transmission of the acknowledgement is completed – an event that might be detected by awaiting the TXFRS (Transmit Frame Sent) status interrupt. **If automatic acknowledgement is not enabled, then the AAT status bit must be ignored.** The AAT bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable.
TXFRB reg:00:44 bit:4	Transmit Frame Begins. This event status bit is set at the start of a frame transmission as the transmitter begins to send preamble. The TXFRB bit is automatically cleared at the next transmitter enable. It can also be cleared explicitly by writing a 1 to it.
TXPRS reg:00:44 bit:5	Transmit Preamble Sent. This event status bit is set at the end of preamble when SFD sending begins. The TXPRS bit is automatically cleared at the next transmitter enable. It can also be cleared explicitly by writing a 1 to it.

Field	Description of fields within Sub-register 0x00:44 – System Event Status
TXPHS reg:00:44 bit:6	Transmit PHY Header Sent. This event status bit is set when the PHR has been transmitted. This marks the start of sending the data part of the frame (assuming the frame length is non-zero) at the configured transmit data rate. The TXPHS bit is automatically cleared at the next transmitter enable. It can also be cleared explicitly by writing a 1 to it.
TXFRS reg:00:44 bit:7	Transmit Frame Sent. This event status bit is set at the end of sending the data part of the frame. It is expected that this will be used as the main "Transmit Done" (interrupt) event signalling the completion of frame transmission. (In the case where frame length is zero the TXFRS bit is set soon after the TXPHS event flag). The TXFRS bit is automatically cleared at the next transmitter enable. It can also be cleared explicitly by writing a 1 to it.
RXPRD reg:00:44 bit:8	Receiver Preamble Detected status. This event status bit is set to indicate that the receiver has detected (and confirmed) the presence of the preamble sequence. Preamble reception continues after RXPRD has been set until the SFD is detected as signalled by the RXSFDD event status bit or an SFD timeout occurs as signalled by the RXSFDTO event status bit. Section 4 *– Message reception* gives details of the frame reception process. The RXPRD bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable.
RXSFDD reg:00:44 bit:9	Receiver SFD Detected. This event status bit is set to indicate that the receiver has detected the SFD sequence and is moving on to decode the PHR. Section 4 *– Message reception* gives details of the frame reception process. The RXSFDD bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable.
CIADONE reg:00:44 bit:10	CIA processing done. This event status bit is set to indicate the completion by the CIA algorithm of the leading edge detection and its other adjustments of the receive timestamp information. The resultant adjusted message RX timestamp is then available in *Sub-register 0x00:60 – Receive Time Stamp*. To accurately determine this timestamp the QM33100 employs the CIA algorithm to adjust the RMARKER receive time. Among other functions this performs a leading edge detection search on the channel impulse response and subtracts the receive antenna delay as programmed in the RXANTD field of *Sub-register 0x0E:00 – RX antenna delay and CIA diagnostic enable*. Where the CIA is configured to operate on the STS which occurs later in the frame, the CIA_DONE event status flag bit is not set until CIA algorithm has completed its processing. For more information on the CIA and the process on message time-stamping see section *4.1.7 – RX Message timestamp*. The CIA_DONE event status flag bit is automatically cleared by the RX enable. It can also be cleared explicitly by writing a 1 to it.
RXPHD reg:00:44 bit:11	Receiver PHY Header Detect. This event status bit is set to indicate that the receiver has completed the decoding of the PHR. Section 4 *– Message reception* gives details of the frame reception process. The RXPHD bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable.
RXPHE reg:00:44 bit:12	Receiver PHY Header Error. This event status bit is set to indicate that the receiver has found a non-correctable error in the PHR. The PHR includes a SECDED error check sequence (see section 2.10) that can correct a single bit error and detect a double bit error. The double error is not correctable and its detection is the event that the RXPHE event status flag is notifying. Generally this error means that correct frame reception is not possible, and this event will abort frame reception. Section 4 *– Message reception* gives details of the frame reception process. The RXPHE bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable. PHY Header Error events are counted in *Sub-register 00F:04 – PHR Error Counter*, as long as counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*.

Field	Description of fields within Sub-register 0x00:44 – System Event Status
RXFR reg:00:44 bit:13	Receiver Data Frame Ready. This event status bit is set to indicate that the completion of the frame reception process. Section 4 *– Message reception* gives details of the frame reception process. It is expected that this will be used as the main "Receive" (interrupt) event signalling the completion of a frame reception, and, that the receive event processing routine will examine the RXFCG and RXFCE to determine whether the frame has been received without error (or not), and also to check the CIA_DONE event status flag to validate the receive timestamp information. In order to ensure that the receive timestamp information is valid before any receive interrupt processing takes place, the setting of RXDFR is delayed until the CIA adjustments of the timestamp have completed, at which time the CIA_DONE event status bit will be set (or possibly CIAERR). The RXDFR event status flag bit is automatically cleared by the RX enable. It can also be cleared explicitly by writing a 1 to it. Note: when configured to use packet mode 3 (see Figure 15) then this event should be used as the main "Receive" (interrupt). TheRXFCG and RXFCE events should not be used in packet mode 3.
RXFCG reg:00:44 bit:14	Receiver FCS Good. This event status bit reflects the result of the frame CRC checking. It is set (or not) at the end of frame reception coincidentally with the setting of the RXDFR event status flag. When RXFCG is set to 1 it indicates that the CRC check result generated on the received data matches with the 2-octet FCS sequence at the end of the frame. RXDFR with RXFCG then indicates the correct reception a valid frame. The RXFCG bit is automatically cleared by RX enable. It can also be cleared explicitly by writing a 1 to it. Note: This event should not be used in packet mode 3.
RXFCE reg:00:44 bit:15	Receiver FCS Error. This event status bit also reflects the result of the frame CRC checking. It is valid at the end of frame reception coincidentally with the setting of the RXDFR event status flag. When RXFCE is set to 1 it indicates that the CRC check result generated on the received data FAILED to match with the 2-octet FCS sequence at the end of the frame. The RXFCE bit is automatically cleared by RX enable. It can also be cleared explicitly by writing a 1 to it. RXFCE events are also counted in *Sub-register 0x0F:0A – FCS Error Counter*, as long as counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*. Note: This event should not be used in packet mode 3.
RXFSL reg:00:44 bit:16	Receiver Reed Solomon Frame Sync Loss. The RXRFSL event status bit is set to indicate that the receiver has found a non-correctable error during the Reed Solomon decoding of the data portion of the frame. Generally this means that correct frame reception is not possible, and this event will abort frame reception. Section 4 *– Message reception* gives details of the frame reception process. The RXRFSL bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable. Reed Solomon Frame Sync Loss Error events are also counted in *Sub-register 0x0F:06 – RSD Error Counter*, as long as counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*.

Field	Description of fields within Sub-register 0x00:44 – System Event Status
RXFTO reg:00:44 bit:17	Receive Frame Wait Timeout. This event status bit is set to indicate that a receive frame wait timeout has occurred. The receive frame wait timeout is enabled by the RXWTOE bit in *Sub-register 0x00:10 – System Configuration*, with the timeout being set by *Sub-register 0x00:34 – Receive Frame Wait Timeout Period*. The receive frame wait timeout starts running when the receiver is enabled and stops running either when a valid frame is received or when the timeout occurs and is signalled by this RXRFTO event status flag bit. The RXRFTO bit is automatically cleared at the next receiver enable. It can also be cleared explicitly by writing a 1 to it. Receive frame wait timeout events are also counted in *Sub-register 0x0F:14 – RX Frame Wait Timeout Event Counter*, as long as counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*.
CIAERR reg:00:44 bit:18	Channel Impulse Response Alalyser processing error. The main function of the CIA algorithm is a search in the channel impulse response to find the first arriving ray of the RMARKER. This search should be bounded and finish in a reasonably short time, CIA includes a failsafe mechanism of a watchdog timer that is initialized at the start of each CIR search when the CIA begins the processing a received CIR, for either the preamble or STS sequences. This watchdog can be disabled by clearing the CIA_WDEN bit CIA_WDEN in DIAG_TMC register which is set by default. We do not expect QM33100 users to ever see this event, however if the watchdog timer expires before the CIA has completed its RX timestamp adjustments then the error is reported through the event status flag. The bit is automatically cleared at the next receiver enable but it can also be cleared explicitly by writing a 1 to it.
VWARN reg:00:44 bit:19	Low voltage warning. This event indicates that the IC has detected that that the voltage has dropped below the warning threshold of 1.5 V. The role of the low voltage detector is to detect drops in the supply during the higher current TX or RX modes where it could cause performance issues. Low voltage events are not expected in normal operation when there is sufficient power supply current and capacitance available. The VWARN event flag stays set until it is explicitly cleared by writing a 1 to it, or the IC is reset. Low voltage events are counted in Sub-register 0x0F:22 – Low voltage warning Error Counter, assuming that counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*
RXOVRR reg:00:44 bit:20	Receiver Overrun. This event status bit only applies when double RX buffering is enabled (by clearing the DIS_DRXB bit in *Sub-register 0x00:10 – System Configuration*). The RXOVRR event flag is set to indicate that an overrun error has occurred in the receiver. See section *4.4.3 – Overrun* for more details of double buffering and the use of this RXOVRR error flag. The RXOVRR event status bit stays set until it is explicitly cleared by writing a 1 to it, or the IC is reset. Receiver Overrun events are also counted in *Sub-register 0x0F:0E –* RX Overrun Error Counter, assuming that counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*

Field	Description of fields within Sub-register 0x00:44 – System Event Status
RXPTO reg:00:44 bit:21	Preamble detection timeout. This event status bit is set when the preamble detection timeout occurs. The preamble detection timer is started when the receiver is enabled and begins preamble hunt. This may begin immediately in the case of issuing an CMD_RX command or after a delay in the case of issuing a CMD_DRX, CMD_DRX_TS or CMD_DRX_RS command. The preamble detection timeout value is programmed in *Sub-register 0x06:04 – DRX_PRETOC*. The RXPTO bit is automatically cleared at the next receiver enable. It can also be cleared explicitly by writing a 1 to it. The preamble detection timeout may be useful to save power by turning off the receiver if an expected response frame does not begin. If a response message is expected with a certain fixed timing and preamble is not detected at the appropriate time then this is likely to mean that the response will not come. Reception can thus be aborted early, saving power.
- reg:00:44 bit:22	reserved
SPIRDY reg:00:44 bit:23	SPI ready for host access. This SPIRDY event status bit is set is set to indicate that the QM33100 has completed the activities associated with power on and has transitioned from **OFF** or awaking from **SLEEP** (or **DEEPSLEEP**) and is now in the state. For more details on QM33100 states see section 2.4 where Figure 11, shows the host interrupt that is generated from the SPIRDY event. The SPIRDY bit is cleared by writing a 1 to it.
RCINIT reg:00:44 bit:24	RC INIT. This event status bit is set is set to indicate that the QM33100 has completed the activities associated with power on and has transitioned from **OFF** or awaking from **SLEEP** (or **DEEPSLEEP**) and is now in the **INIT_RC** state. For more details on QM33100 states see section 2.4 The RCINIT bit is cleared by writing a 1 to it.
PLL_HILO reg:00:44 bit:25	Clock PLL Losing Lock. This event status bit is set is set to indicate that the system's digital clock PLL is having locking issues. This should not happen in healthy devices operating in their normal range. Its occurrence may indicate a bad configuration, a faulty part or a problem in the power or clock inputs to the device. If this bit is set it may be advisable to turn off the transmitter to avoid sending spurious signals. The PLL_HILO bit is cleared by writing a 1 to it.
RXSTO reg:00:44 bit:26	Receive SFD timeout. This event status bit is set when the SFD detection timeout occurs. The SFD detection timeout starts running as soon as preamble is detected. If the SFD sequence is not detected before the timeout period expires then the timeout will act to abort the reception currently in progress. The period of the SFD detection timeout is in *Sub-register 0x06:00 – DRX_SFDTOC*. By default this has a value of 4096+64 representing the longest possible preamble and SFD. Where it is known that a shorter preamble and SFD are being employed this value can be reduced. The RXSFDTO event status bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable. SFD timeout events are also counted in *Sub-register 0x0F:10 – SFD Timeout Error Counter*, assuming that counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*.

Field	Description of fields within Sub-register 0x00:44 – System Event Status
HPDWARN reg:00:44 bit:27	Half Period Delay Warning. This event status bit relates to the use of delayed transmit and delayed receive functionality. It indicates the delay is more than half a period of the system clock. For delayed send/receive the send/receive time is programmed into *Sub-register 0x00:28 – Delayed Send or Receive time* and then the delayed sending/receiving is initiated by the following commands: CMD_DTX , CMD_DTX_TS, CMD_DTX_RS, CMD_DTX_W4R, CMD_DTX_TS_W4R, CMD_DTX_RS_W4R, CMD_DRX, CMD_DRX_TS or CMD_DRX_RS. The delayed transmit and receive functionality is described in detail in sections *3.3 – Delayed transmission* and 4.3 – Delayed receive The HPDWARN event status flag gets set if the time left to actually beginning transmission / reception is more than half a period of the system clock (SYS_TIME) away. Assuming that the intent was not to schedule transmission/reception at a time that is over 8 seconds in the future, the HPDWARN status flag can be polled after a delayed transmission or reception is commanded, to check whether the delayed send/ receive invocation was given in time (HPDWARN ==0) or not (HPDWARN == 1). Typically when the HPDWARN event is detected the host controller will abort the delayed TX/RX by issuing a transceiver off command (CMD_TRXOFF) and then take whatever remedial action is deemed appropriate for the application. The HPDWARN event status bit can be cleared explicitly by writing a 1 to it. The HPDWARN bit is automatically cleared at the next transmitter enable. HPDWARN events are counted in *Sub-register 0x0F:18 – Half Period Warning Counter*, assuming counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*.
CPERR reg:00:44 bit:28	Scramble Timestamp Sequence (STS) error. The CPERR event status flag will get set if the CP_TOAST bits are non-zero. The CPERR events are counted in EVC_CPQE field in Sub-register 0x0F:20 – STS Quality Error Counter. The CPERR status bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable.
ARFE reg:00:44 bit:29	Automatic Frame Filtering rejection. The AFFREJ event status flag bit is set to indicate when a frame has been rejected in receiver due to it not passing through the frame filtering. See section *5.4 – Frame filtering* for details of the operation of frame filtering. The AFFREJ event status bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable. Frame Filtering rejection events are also counted in *Sub-register 0x0F:0C – Frame Filter Rejection Counter*, assuming counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event Counter Control*.
TIMER0 reg:00:44 bit:30	Timer 0 event indicator. The TIMER0 event bit will be set when Timer 0 is running and its count reaches the terminal value programmed in the TIMER0_CNT_SET register, or when the timer is configured (by TIMER0_GPIO=1) to stop on the assertion of the GPIOIRQ event and that happens. The TIMER0 event status bit may be cleared by writing a 1 to it.

Field	Description of fields within Sub-register 0x00:44 – System Event Status
TIMER1 reg:00:44 bit:31	Timer 1 event indicator. The TIMER1 event bit will be set when Timer 1 is running and its count reaches the terminal value programmed in the TIMER1_CNT_SET register, or when the timer is configured (by TIMER1_GPIO=1) to stop on the assertion of the GPIOIRQ event and that happens. The TIMER1 bit is cleared by writing a 1 to it.
- reg:00:48 bit:0	reserved
RXPREJ reg:00:48 bit:1	Receiver Preamble Rejection. This is a low-level event status flag, which is probably not of interest to the host system. It was used during the IC implementation as part of tuning the preamble detection algorithm. In the QM33100, preamble detection a two stage process where preamble is initially seen and then has to be confirmed as continuing for a number of symbols before the RXSFDD event status bit actually gets set. If the preamble is not confirmed then the RXSFDD event status bit will not be set, but instead this RXPREJ status will be set. The RXPREJ event status bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable.
PGFCAL_ERR reg:00:48 bit:2	PGF/RX calibration error event. If this event gets set, this means that the receiver calibration has not run correctly. The host should re-run RX calibration, otherwise the receiver performance will be severely affected. For further details see RX_CAL register, and consult API functions [3]
VT_DET reg:00:48 bit:4	Voltage or temperature variation detected.
GPIOIRQ reg:00:48 bit:5	GPIO interrupt. The GPIOIRQ event status bit is set when an interrupt condition occurs in the GPIO block. Various configurations are possible to enable interrupts coming from GPIO input lines. The GPIO block may need to be interrogated to determine the source of the interrupt if more than one input line is configured to interrupt. The GPIOIRQ bit is cleared by writing a 1 to it. For details of GPIO programming see Register file: 0x05 – GPIO Control and status.
AES_DONE reg:00:48 bit:6	AES-DMA operation complete. The AES_DONE bit is cleared by writing a 1 to it.
AES_ERR reg:00:48 bit:7	AES-DMA error, indicates an AES authentication error or DMA transfer error or memory address conflict. The AES_ERR bit is cleared by writing a 1 to it.
CMD_ERR reg:00:48 bit:8	Command error. Indicates that a fast command was programmed, given that will be ignored. This can happen when the host issues two commands in quick succession, if the device receives the second of the two while the first has not completed, the device will ignore the second command and raise this event. Section 9 Fast Commands describes the available commands.
SPI_OVF reg:00:48 bit:9	SPI overflow error. Occurs is written into the RX FIFO at too fast a rate and the FIFO overflows possibly due to an SCLK frequency that is set much too fast.
SPI_UNF reg:00:48 bit:10	SPI underflow error. Occurs when the data to be read is not available in the TX FIFO possibly due to clocks not being turned on or a very high SCLK frequency.

Field	Description of fields within Sub-register 0x00:44 – System Event Status
SPIERR reg:00:48 bit:11	SPI collision error. If there is a case of a failed SPI transaction caused by internal contention the QM33100 will indicate this by the SPIERR event. This SPI_COLLISION status indicates which internal QM33100 block has conflicted with the host SPI access, and that last host SPI transaction has not successfully completed.
CCA_FAIL reg:00:48 bit:12	This event will be set as a result of failure of CMD_CCA_TX to transmit a frame. When CMD_CCA_TX is invoked the device initially enters preamble hunt, and if no preamble is found within the time specified byPRE_TOC, the device will proceed to transmit the frame. However, if the preamble is detected, then the device will go back to IDLE and report CCA_FAIL event. The CCA_FAIL bit is cleared by writing a 1 to it. It is also automatically cleared at the next transmitter enable.
COEX_ERR reg:00:48 bit:13	Coexistence error event status flag. The COEX_ERR event status bit is set when the GPIO acting as the coexistence input has been asserted to abort an active transmission or reception activity.
COEX_CLR reg:00:48 bit:14	Coexistence clear event status flag. The COEX_CLR event status bit is set to indicate that the GPIO acting as the coexistence input has been de-asserted.
TXSTS reg:00:48 bit:16	Transmit STS sent. This event status bit is set at the end of sending the STS sequence (if enabled). The TXSTS bit is automatically cleared at the next transmitter enable. It can also be cleared explicitly by writing a 1 to it.
RXSTS reg:00:48 bit:17	Receiver STS received. This event status bit is set to indicate that the receiver has completed the reception of the STS sequence. Section 4 – *Message reception* gives details of the frame reception process. The RXSTS bit can be cleared explicitly by writing a 1 to it. It is also automatically cleared by the next receiver enable. Note this does not indicate that the STS is correct but meerly that the receiver has completed that phase of the recrption. The correctness of the STS and the resultan RX timestamp should be assessed using the STS ACC_QUAL and STS_TOAST status reports.

8.2.2.14 Sub-register 0x00:4C – RX frame information

ID	Length (octets)	Type	Mnemonic	Description
00:4C	4	ROD	RX_FINFO	RX frame information

Register file: 0x00-0x1 – General configuration registers, sub-register 0x4C of register file 0x00 gives information on the received frame. It is updated after the reception of a good PHR, i.e. PHR where the SECDED has not flagged a non-correctable error (see section 2.10).

This RX_FINFO register contains a number of fields, separately identified and described below:

REG:00:4C – RX_FINFO – RX frame information

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXPACC												RXPSR		RXPRF		RNG	-	RXBR	RXNSPL		-	RXFLEN									
0												0		0		0	0	0	0		0	0									

The individual sub-fields are described below:

Field	Description of fields within Sub-register 0x00:4C – RX frame information
RXFLEN reg:00:4C bits:9–0	Receive Frame Length. This value is copied from the PHR of the received frame when a good PHR is detected (when the RXPHD status bit is set). The frame length from the PHR is used in the receiver to know how much data to receive and decode, and where to find the FCS (CRC) to validate the received data. The frame length also tells the host system how much data to read from the receive buffer (RX_BUFFER_0 or RX_BUFFER_1) This field is 10-bits wide to accommodate both the extended frame mode of operation (up to 1023 bytes, IEEE802.15.8 standard [4]) and the IEEE802.15.4 standard [1] frames which can be up to 127 bytes long. The PHR mode of operation is enabled via the PHR_MODE selection bits of *Sub-register 0x00:10 – System configuration*. See also section **3.4 – Extended length data frames**.
- reg:10:00 bit:10	This bit is reserved.

<table>
<tr><th>Field</th><th>Description of fields within Sub-register 0x00:4C – RX frame information</th></tr>
<tr><td>RXNSPL

reg:00:4C
bits:12,11</td><td>Receive non-standard preamble length. The QM33100 is able to send non-standard preamble lengths to allow system designers more choice in optimising performance. The RXNSPL field operate in conjunction with the RXPSR field to report the received preamble length. The RXPSR field reports the preamble length as signalled in the PHR (see 2.10 for details). The receiver determines additional information about the transmitted preamble length from the count of preamble accumulation, as reported by the RXPACC field, and uses this to set the RXNSPL value.

Table 22 below lists the preamble lengths that can be reported by considering RXNSPL and the RXPSR fields together:

Table 22: preamble length reporting

<table>
<tr><th>Bit 19</th><th>Bit 18</th><th>Bit 12</th><th>Bit 11</th><th></th></tr>
<tr><th colspan="2">RXPSR</th><th colspan="2">RXNSPL</th><th>RX Preamble Length</th></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>64</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>128</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>256</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>512</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>1024</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>1536</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>2048</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>4096</td></tr>
</table>
The bit numbers quoted above are the bit numbers in the RX_FINFO register.

Where preamble length is not predetermined and hard coded in the application, the received preamble length information may be used to select the preamble length for any response message, by copying RXNSPL and RXPSR fields into the TXPSR configuration respectively (TXPSR = RXNSPL << 2 + RXNSPL).
This value is updated when a good PHR is detected (when the RXPHD status bit is set).</td></tr>
<tr><td>RXBR

reg:00:4C
bit:13</td><td>Receive Bit Rate report. This field reports the received bit rate. This information is signalled in the received frames's PHR (see 2.10 for details): 0 = 850 kb/s, and 1 = 6.8Mb/s
This value is updated when a good PHR is detected (when the RXPHD status bit is set).</td></tr>
<tr><td>-
reg:00:4C
bit:14</td><td>This bit is reserved.</td></tr>
<tr><td>RNG

reg:00:4C
bit:15</td><td>Receiver Ranging. This reflects the ranging bit in the received PHY header identifying the frame as a ranging frame.
This value is updated when a good PHR is detected (when the RXPHD status bit is set).</td></tr>
<tr><td>RXPRF

reg:00:4C
bits:17,16</td><td>RX Pulse Repetition Rate report. This field reports the PRF being employed in the receiver. This is determined by the RX_PCODE programmed in CHAN_CTRL register. The values are: 01 = 16 MHz, 10 = 64 MHz</td></tr>
</table>

Field	Description of fields within Sub-register 0x00:4C – RX frame information
RXPSR reg:00:4C bits:19,18	RX Preamble Repetition. This field reports the received frames's preamble length as signalled in the frames's PHR (see 2.10 for details). The values of these two bits are defined by the standard as: 00 = 16 symbols, 01 = 64 symbols, 10= 1024 symbols, and, 11= 4096 symbols In addition to these standard preamble lengths, the QM33100 also supports the transmission of non-standard preamble lengths. These non-standard lengths cannot be signalled in the PHR; instead the QM33100 gives an estimate of the preamble length based on the RXPSR from the PHR and the RXPACC value. The estimate is reported using RXPSR and RXNSPL fields together as per Table 22 above. This value is updated when a good PHR is detected (when the RXPHD status bit is set).
RXPACC reg:00:4C bits:31–20	Preamble Accumulation Count. This reports the number of symbols of preamble and SFD that are accumulated. The value is updated/available after a good PHR is detected (when the RXPHD status bit is set). RXPACC will usually be slightly smaller than the transmitted preamble length because detecting the presence of the preamble consumes several symbols (a PAC size) and they do not contribute to the CIR estimate. **Note:** This value is similar but not the same as the IP_NACC from the CIA interface. IP_NACC allows for the effect of negative and zero symbols in the SFD to produce an effective accumulated symbol count and it should be used when determining RSSI.

8.2.2.15 Sub-register 0x00:60 – Receive time stamp

ID	Length (octets)	Type	Mnemonic	Description
00:60	12	ROD	RX_TIME	Receive time stamp

Register file: 0x00-0x1 – General configuration registers, sub-register 0x64 of register file 0x00 reports the receive time stamp. The IEEE802.15.4 standard [1] defines a point during a frame reception which is timestamped (shown in Figure 13), this point is called the RMARKER. QM33100 takes a coarse timestamp of the symbol in which the RMARKER event occurs and to this adds various correction factors to give a resultant time stamp value. Please refer to section *4.1.7 – RX message* timestamp for more details of the corrections applied.The RX_TIME register contains the following sub-fields:

REG:00:60 – RX_TIME – Receive time stamp (Octets 0 to 3, 32-bits)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_STAMP (low 32 bits of 40-bit value)																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REG:00:64 – RX_TIME – Receive time stamp (Octet 4)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								RX_STAMP (high 8 bits of 40)							
																								0	0	0	0	0	0	0	0

REG:00:6C – RX_TIME – Receive time stamp (Octets 8 to 11, 32-bits)																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_RAWST																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The sub-fields of RX_TIME register are described below:

Field	Description of fields within Sub-register 0x00:60 – Receive time stamp
RX_STAMP reg:00:60 bits:39–0	This 40-bit (5-octet) field reports the fully adjusted time of reception. Please refer to section *4.1.7 – RX message timestamp* for more details of the adjustments applied. The units of the low order bit are approximately 15.65 picoseconds. The actual unit may be calculated as 1/ (128*499.2×10^6) seconds. The value is available here when the leading edge determination and timestamp adjustments are completed (when the CIADONE event status flag bit is set). **Note:** The RX timestamp estimate resulting from running the CIA on the preamble CIR is written here and also to the IP_TOA field in *Sub-register 0x0C:00 – Preamble receive time stamp and status*, subsequently if STS is enabled the RX_STAMP value here may be over written by the STS based RX timestamp estimate, which is also written to the STS_TOA field in *Sub-register 0x0C:08 – STS receive time stamp and status*. Please refer to *§ 6 – Secure ranging / timestamping* for details of the operation of the STS for secure ranging operations.
- reg:00:64 bits:31-8	These bits are reserved.
RX_RAWST reg:00:6C bits:31–0	This 32-bit (4-octet) field reports the Raw Timestamp for the frame. This is the value of the system clock (125 MHz) captured at start of the PHR (the high 32-bits). The precision here is approximately 125 MHz (8 ns), i.e. the least significant bit is zero.

8.2.2.16 Sub-register 0x00:70 – Transmit time stamp

ID	Length (octets)	Type	Mnemonic	Description
00:70	5	RO	TX_TIME	Transmit time stamp

Register file: 0x00-0x1 – General configuration registers, sub-register 0x74 of register file 0x00 reports the transmit time stamp information. The IEEE802.15.4 standard [1] defines a point during a frame transmission which is timestamped (shown in Figure 13), this point is called the RMARKER. The QM33100 takes a timestamp of the symbol in which the RMARKER event occurs and to this adds the antenna delay to give a resultant time stamp value, of when the RMARKER is launched from the antenna.

This TX_TIME register contains the following sub-fields:

REG:00:70 – TX_TIME – Transmit time stamp (Octets 0 to 3, 32-bits)																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_STAMP (low 32 bits of 40-bit value)																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REG:00:74 – TX_TIME – Transmit time stamp (Octet 4, 8-bits)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-																								TX_STAMP (high 8 bits of 40)							
-																								0	0	0	0	0	0	0	0

The sub fields of TX_TIME register are laid out above. It is possible to read a variable number of bytes any byte index. The individual sub-fields are described below:

Field	Description of fields within Sub-register 0x00:70 – Transmit time stamp
TX_STAMP reg:00:70 bits:39–0	This 40-bit (5-octet) field reports the fully adjusted time of transmission. The unit of the least significant bit is approximately 15.65 picoseconds. The actual unit may be calculated as 1/ (128*499.2×10^6) seconds. The value is available here when the PHR transmission has completed.

8.2.2.17 Sub-register 0x00:78 – Transmit time stamp raw

ID	Length (octets)	Type	Mnemonic	Description
00:78	4	RO	TX_RAWST	Transmit time stamp raw

Register file: 0x00-0x1 – General configuration registers, sub-register 0x78 of register file 0x00 reports the raw transmit time stamp information. The IEEE802.15.4 standard [1] defines a point during a frame reception which is timestamped (shown in Figure 13), this point is called the RMARKER. The QM33100 takes a timestamp of the symbol in which the RMARKER event occurs. This is a 32-bt register which contains 39-8 bits of the TX_TIME register prior to addition of antenna delay.

8.2.2.18 Sub-register 0x00:7C – Transmitter antenna delay

ID	Length (octets)	Type	Mnemonic	Description
00:7C	2	RW	TX_ANTD	16-bit delay from transmiter to the antenna

Register file: 0x00-0x1 – General configuration registers, sub-register 0x7C of register file 0x00, the Transmitter Antenna Delay, is used to account for the delay between the internal digital timestamp of the RMARKER and the time the RMARKER is at the antenna. The value programmed here is automatically added to the raw timestamp TX_RAWST to get the TX_STAMP reported in *Sub-register 0x00:70 – Transmit time stamp*. Refer to section *10.3 – IC calibration – antenna delay* for details of calibration of antenna delay. The units here are the same as those used for system time and time stamps, i.e. 499.2 MHz × 128, so the least significant bit is about 15.65 picoseconds. The default antenna delay is 0x4015, which is approx. 256.74 ns.

8.2.2.19 Sub-register 0x01:00 – Acknowledgement time and response time

ID	Length (octets)	Type	Mnemonic	Description
01:00	4	RW	ACK_RESP_T	Acknowledgement delay time and response time

Register file: 0x00-0x1 – General configuration registers, sub-register 0x00 of register file 0x01 is a configuration register used for specifying turn-around times for QM33100 to use when automatically switching between TX mode and RX modes. The ACK_RESP_T register contains the following bitmapped sub-fields:

REG:01:00 – ACK_RESP – Acknowledgement time and response time																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACK_TIM								-	-	-	-	W4R_TIM																			
0								0	0	0	0	0																			

The individual sub-fields are described below:

Field	Description of fields within Sub-register 0x01:00 – Acknowledgement time and response time
W4R_TIM reg:01:00 bits:19–0	Wait-for-Response turn-around Time. This 20-bit field is used to configure the turn-around time between TX complete and RX enable when the wait for response function is being used. This function is enabled by any of the TX and wait-4-response commands (TXW4R, TXW4RCCA, TXW4RDLY, TXW4RDLYREF, TXW4RDLYTS, and TXW4RDLYRS). The time specified by this W4R_TIM parameter is in units of approximately 1 µs, or 128 system clock cycles. This configuration may be used to save power by delaying the turn-on of the receiver, to align with the response time of the remote system, rather than turning on the receiver immediately after transmission completes. For more details see section *5.6 – Transmit and automatically wait for response*.

Field	Description of fields within Sub-register 0x01:00 – Acknowledgement time and response time
ACK_TIM reg:01:00 bits:31–24	Auto-Acknowledgement turn-around Time. This 8-bit field is used to configure the turn-around time between the correct receipt of a data frame (or a MAC command frame) and the transmission by the QM33100 of the acknowledgement frame. The time here is specified in units of preamble symbols. (This resultant time is slightly different depending on whether the PRF is 16 or 64 MHz, see Table 9 for details the preamble symbol lengths).This timer only applies if auto-acknowledgement is in use and then only acts when the frame is correctly received, passing through the RX frame filtering rules, and when the ACK bit in the frame's MAC header is set to request acknowledgement. Please refer to section *5.5 –* Automatic acknowledgment Automatic for details of the Acknowledgement function. To ensure that the receiver is ready for the first preamble symbol, and assuming that the remote QM33100 has a its W4R_TIM parameter set to 0, the recommended minimum ACK_TIM settings are as follows: Data Rate / Recommend min. ACK_TIM 850 kb/s / 2 6.8 Mb/s / 3 This is most important at the 6.8 Mb/s data rate, where preamble sequences are generally short, and losing even a few preamble symbols could potentially compromise ACK reception. Where the W4R_TIM parameter is larger than zero, the ACK_TIM setting should be increased also to ensure that none of the packet is sent before the remote receiver is listening. **Note:** When the CIA algorithm is being run, the frame reception RXFR (and RXFCG) event status flags are delayed until the CIA algorithm completes. This is done to ensure that the receive timestamp information is valid when the interrupt is asserted. This delay is incurred by the Auto-Acknowledgement function, since it is initiated by the frame reception RXFR event also. The CIA processing delay is around 70 µs for a non-PDoA STS mode (35 µs each for preamble and STS CIR analysis). Any programmed ACK_TIM will act in addition to this, i.e. the delay timer will only start running when the RXFR event is asserted. Clearly, where a remote device is waiting for the ACK response it will need to take this extra delay into account in its turnaround receiver enablement and any timeouts it may be implementing. To speed up Auto-Acknowledgement transmission, FAST_AAT bit can be set or where the RX timestamp is not required, the CIA analysis may be disabled by clearing both CIA_IPATOV and CIA_STS configuration bits in *Sub-register 0x00:10 – System configuration.*
- reg:01:00	Bits marked '-' are reserved and should always be written as zero.

8.2.2.20 Sub-register 0x01:04 – Transmit power control

ID	Length (octets)	Type	Mnemonic	Description
01:04	4	RW	TX_POWER	TX power control

Register file: 0x00-0x1 – General configuration registers, sub-register 0x04 of register file 0x01 is used for configuration and control of the transmitter output power.

The transmitter output power can be adjusted using this *Sub-register 0x01:04 – Transmit power control*. Register, which contains the four octets each of which specifies a separate transmit power setting. Each power control octet, specifies the power as a combination of a coarse gain parameter and a fine gain parameter.

REG:01:04 – TW_POWER – TX power control register (octets 0 to 3)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS_PWR								SHR_PWR								PHR_PWR								DATA_PWR							
FINE						COARSE		FINE						COARSE		FINE						COARSE		FINE						COARSE	
0x82								0x82								0x82								0x82							

The individual sub-fields of TX_POWER register are described below:

Field	Description of fields within Sub-register 0x01:04 – Transmit power control
DATA_PWR reg:01:04 bits:7–0	This octet is used to control the TX power of the Data (PHY Payload) part of the TX packet (see Figure 13). The gain control range consists of 64 fine control steps (bits 7-2) and 4 coarse control steps (bits 1-0). The fine control steps provide approximately 25 dB of control, while the coarse steps provide an additional 7 dB of control. For the best current consumption performance, the coarse steps should be minimised for the desired output power.
PHR_PWR reg:01:04 bits:15–8	This octet is used to control the TX power of the PHY header (PHR) part of the TX packet (see Figure 13). To optimise the output power profile and comply with regulations, it is required to reduce the PHR_PWR setting in comparison to the DATA_PWR, SHR_PWR and STS_PWR settings. The PHR_PWR setting should reflect a Tx power reduction of 6dB relatively to the DATA_PWR, SHR_PWR and STS_PWR settings. The tables from *APPENDIX 4: Reference TX_POWER setting* tables provide reference settings for which PHR byte is reduced relatively to other bytes. It is advised to refer to these tables when selecting a reference value for tx_power register.
SHR_PWR reg:01:04 bits:23–16	This octet is used to control the TX power during the transmission of the synchronisation header (SHR), which consists of the preamble and SFD, portions of the frame that precede the PHR of the SFD part of the TX frame (see Figure 13). The gain control is as specified in DATA_PWR above.
STS_PWR reg:01:04 bits:31–24	This octet is used to control the TX power of the STS part of the TX packet (see Figure 13). The gain control is as specified in DATA_PWR above.

Note: For optimum performance, manufacturers have to calibrate the TX power of each unit/QM33100 IC board/module to account for IC to IC variations and different IC to antenna losses. Usually the TX power is set to the maximum allowed by spectral emission regulations (-41.3 dBm/MHz) and such that no other out-of-band limits are exceeded.

The register default values may be too high for the use case. In order to comply with regional spectrum regulations the resultant output power spectrum should be checked.

8.2.2.20.1 Transmit power control

QM33100 provides functionality to exercise power control over various parts of the transmit packet individually. Please refer to section 3 Message transmission, for details of packet structure. If a packet is short, it is possible to boost the power for the packet while staying within the regulatory limits. Each part of the packet may be controlled separately so that the power can be boosted without violating the regulatory limits for peak power. However, in general, the power settings for all parts of the packet should be programmed to the same value.

8.2.2.20.2 Transmit power variation across coarse and fine gain steps

Figure 27 below, shows typical TX power variation over coarse and fine gain steps, for the two channels.

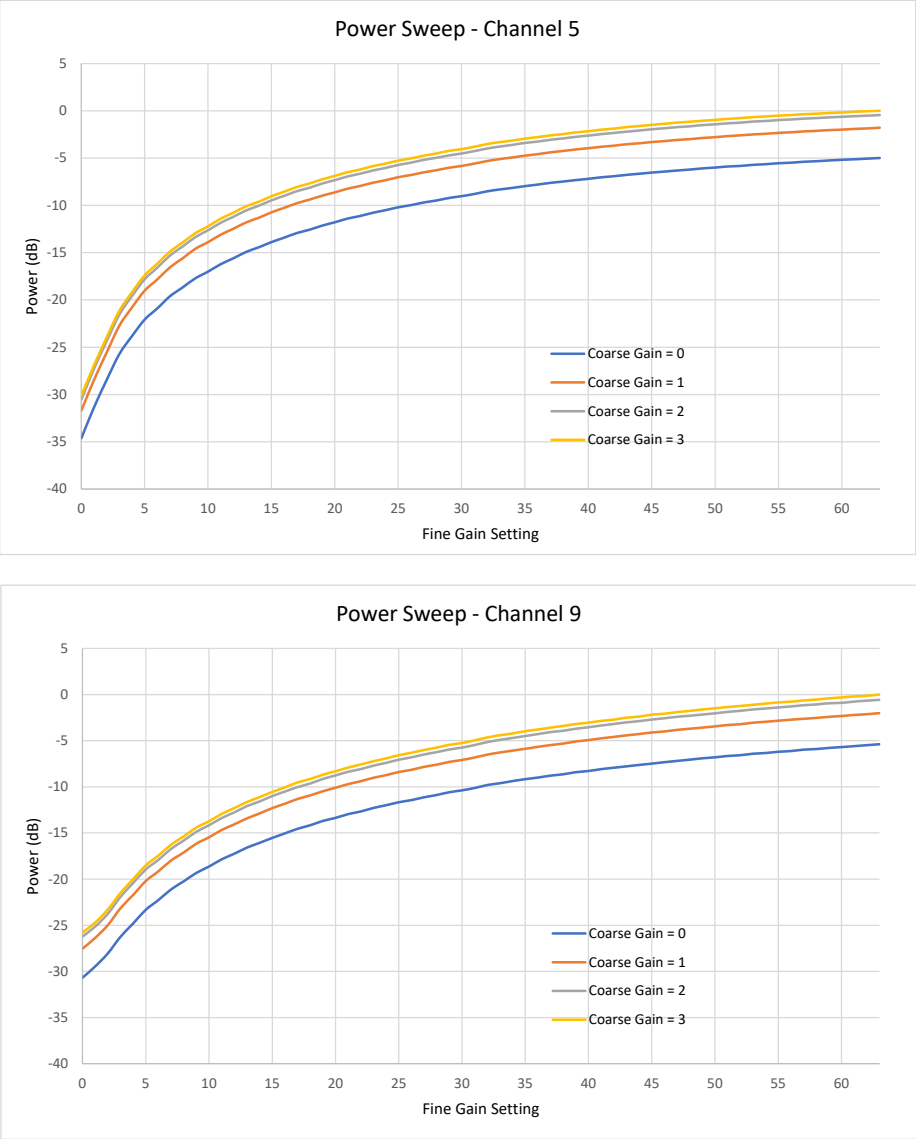

Figure 27: Typical TX power variation with coarse and fine gain

Note:

Channel 9: the maximum coarse gain setting is 2, and the setting 3 should not be used.

Channel 5: the recommended coarse gain setting is 2. There is only a marginal increase in TX power using coarse gain setting of 3, but there is a relatively large increase in current when it is applied. For further details see the Datasheet [5].

8.2.2.21 Sub-register 0x01:08 – Channel control

ID	Length (octets)	Type	Mnemonic	Description
01:14	2	RW	CHAN_CTRL	Channel control register

Register file: 0x00-0x1 – General configuration registers, sub-register 0x14 of register file 0x01 is the channel control register. This is used to select transmit and receive channels, and configure preamble codes and some related parameters.

The fields of the CHAN_CTRL register are defined as follows:

REG:01:14 – CHAN_CTRL – Channel control register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	RX_PCODE					TX_PCODE					SFD_TYPE		RF_CHAN
																0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

The individual sub-fields are described below:

Field	Description of fields within Sub-register 0x01:08 – Channel control
RF_CHAN reg:01:14 bit:0	This selects the transmit and receive channel. Supported channels are 5, and 9. Setting bit 0 to 0 selects channel 5 and to 1 selects channel 9. For the complete configuration of TX and RX block for each channel following registers also need to be configured: Full selection of TX channel requires that the following parameters are also set appropriately: • *Sub-register 0x07:1A – RF* • *Sub-register 0x07:1C – RF TX control register* 2 • *Sub-register 0x09:00 – PLL configuration* For correct operation of the QM33100 and compliance to the IEEE802.15.4 standard [1], the preamble code should be set according to the operating channel. For details of centre frequencies and preamble codes for the supported channels, please refer to section *2.7 – UWB channels and preamble codes*.
SFD_TYPE reg:01:14 bit:2-1	Thise bits select the SFD type. Four different SFD types are supported. These are listed in the Table 23. **Table 23: SFD types** SFD_TYPE (bits 2:1) / Sequence / Description 00 / 0+0-+00- / IEEE 802.15.4 short 8-symbol SFD 01 / ----+-00 / Decawave-defined 8-symbols SFD 10 / ----+-+--++--+00 / Decawave-defined 16-symbols SFD 11 / ---+--+- / IEEE 802.15.4z defined 8-symbol SFD

Field	Description of fields within Sub-register 0x01:08 – Channel control
TX_PCODE reg:01:14 bits:7–3	This field selects the preamble code used in the transmitter. The valid codes are 1 to 29. The user should select the preamble code from those recommended for the selected channel and the PRF they wish to use. The selection of the preamble code will set the PRF, selecting 1-8 will set 16 MHz PRF, selecting 9-24 will set 64 MHz PRF. Section *2.7 – UWB channels and preamble codes* details the preamble codes allowed in the supported channels.
RX_PCODE reg:01:14 bits:12–8	This field selects the preamble code used in the receiver. The valid codes are 1 to 29. The user should select the preamble code from those recommended for the selected channel and the PRF they wish to use. The selection of the preamble code will set the PRF, selecting 1-8 will set 16 MHz PRF, selecting 9-24 will set 64 MHz PRF. Section *2.7 – UWB channels and preamble codes* details the preamble codes allowed in the supported channels.
- reg:01:14 bits:15–13	Bits marked '-' are reserved.

8.2.2.22 Sub-register 0x01:0C – LE PEND address 0 and 1

ID	Length (octets)	Type	Mnemonic	Description
01:0C	4	RW	LE_PEND_01	Low Energy device address 0 and 1

Register file: 0x00-0x1 – General configuration registers, sub-register 0x0C of register file 0x01 is a register used for specifying the 16-bit addresses of LE devices for which there is data and PEND bit should be set in the ACK frame (as a response to a MAC Data Request command from that node). See also section 5.5.2 Frame pending bit. The LE_PEND_01 register contains the following bitmapped sub-fields:

REG:01:0C – LE_PEND_01 – LE PEND address																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LE_ADDR1																LE_ADDR0															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The individual sub-fields are described below:

Field	Description of fields within Sub-register 0x01:0C – LE PEND address 0 and 1
LE_ADDR0 reg:01:0C bits:15–0	Low Energy device 16-bit address. Host may program a 16-bit address of a node for which it has pending data. Then when that node sends a MAC Data Request command, the device will reply with an ACK with PEND bit set. The host can then send data to that device in the following message. LE0_PEND bit must also be set. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail.

Field	Description of fields within Sub-register 0x01:0C – LE PEND address 0 and 1
LE_ADDR1 reg:01:0C bits:31–16	Low Energy device 16-bit address. Host may program a 16-bit address of a node for which it has pending data. Then when that node sends a MAC Data Request command, the device will reply with an ACK with PEND bit set. The host can then send data to that device in the following message. LE1_PEND bit must also be set. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail..

8.2.2.23 Sub-register 0x01:10 – LE PEND address 2 and 3

ID	Length (octets)	Type	Mnemonic	Description
01:10	4	RW	LE_PEND_23	Low Energy device address 2 and 3

Register file: 0x00-0x1 – General configuration registers, sub-register 0x10 of register file 0x01 is a register used for specifying the 16-bit addresses of LE devices for which there is data and PEND bit should be set in the ACK frame (as a response to a MAC Data Request command from that node). See also section 5.5.2 Frame pending bit. The LE_PEND_23 register contains the following bitmapped sub-fields:

REG:01:10 – LE_PEND_23 – LE PEND address																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LE_ADDR3																LE_ADDR2															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The individual sub-fields are described below:

Field	Description of fields within Sub-register 0x01:10 – LE PEND address 2 and 3
LE_ADDR2 reg:01:10 bits:15–0	Low Energy device 16-bit address. Host may program a 16-bit address of a node for which it has pending data. Then when that node sends a MAC Data Request command, the device will reply with an ACK with PEND bit set. The host can then send data to that device in the following message. LE2_PEND bit must also be set. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail.
LE_ADDR3 reg:01:10 bits:31–16	Low Energy device 16-bit address. Host may program a 16-bit address of a node for which it has pending data. Then when that node sends a MAC Data Request command, the device will reply with an ACK with PEND bit set. The host can then send data to that device in the following message. LE3_PEND bit must also be set. Note AUTO_ACK bit in SYS_CFG, and frame filtering rules (e.g. to allow MAC commands) in FF_CFG must also be configured. Section 5.4 describes frame filtering in more detail.

8.2.2.24 Sub-register 0x01:14 – SPI collision status

ID	Length (octets)	Type	Mnemonic	Description
01:14	1	RW	SPI_COLLISION	SPI collision status

Register file: 0x00-0x1 – General configuration registers, sub-register 0x14 of register file 0x01 is a status register used for specifying which QM33100 internal block caused the SPI collision error event. The SPI_COLLISION register contains the following bitmapped sub-fields:

REG:01:20 – SPI_COLLISION – SPI collision status

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	SPI_COLLISION				
																								0	0	0	0	0	0	0	0

The individual sub-fields are described below:

Field	Description of fields within Sub-register 0x01:14 – SPI collision status
SPI_COLLISION reg:01:14 bits:4–0	SPI collision status. If there is a case of a failed SPI transaction caused by internal contention the QM33100 will indicate this by the SPIERR event. This SPI_COLLISION status indicates which internal QM33100 block has conflicted with the host SPI access: 0x10 = AON, 0x08 = CIA while accessing ROM, 0x04 = CIA while accessing RAM, 0x02 = digital receiver, 0x01 = digital transmitter.
- reg:01:14 bits:7–5	Bits marked '-' are reserved.

8.2.2.25 Sub-register 0x01:18 – RX double buffer status

ID	Length (octets)	Type	Mnemonic	Description
01:18	1	RW	RDB_STATUS	RX double buffer status

Register file: 0x00-0x1 – General configuration registers, sub-register 0x18 of register file 0x01 is a status register used for indicating the events following a reception of a frame when operating in double buffer mode. It contains a sub section of status bits which happen as a result of frame reception. More details on the operation of double buffering are given in section 4.4 Double receive buffer. The RDB_STATUS register contains the following bitmapped sub-fields:

REG:01:18 – RDB_STATUS – RX double buffer status																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								CP_ERR1	CIADONE1	RXFR1	RXFCG1	CP_ERR0	CIADONE0	RXFR0	RXFCG0
																								0	0	0	0	0	0	0	0

The individual RDB_STATUS register sub-fields are described below:

Field	Description of fields within Sub-register 0x01:18 – RX double buffer status
RXFCG0 reg:01:18 bit:0	Receiver FCS Good. This event status bit reflects the result of the frame CRC checking. When the frame has been received in RX_BUFFER_0 when operating in double buffer mode (see section 4.4 Double receive buffer). When RXFCG0 is set to 1 it indicates that the CRC check result generated on the received data matches with the 2-octet FCS sequence at the end of the frame. RXFR0 with RXFCG0 then indicates the correct reception a valid frame in RX_BUFFER_0. The RXFCG0 status bit can be cleared explicitly by writing a 1 to it.
RXFR0 reg:01:18 bit:1	Receiver Data Frame Ready. This event status bit is set to indicate that the completion of the frame reception process of a frame in RX_BUFFER_0 when operating in double buffer mode (see section 4.4 Double receive buffer). It is expected that this will be used as the main "Receive" (interrupt) event signalling the completion of a frame reception in double buffer mode, and, that the receive event processing routine will examine the RXFCG0 and determine whether the frame has been received correctly, and also to check the CIADONE0 event status flag to validate the receive timestamp information. The RXFR0 status bit can be cleared explicitly by writing a 1 to it.
CIADONE0 reg:01:18 bit:2	CIA processing done on the CIR relating to a message in RX_BUFFER_0 when operating in double buffer mode (see section 4.4 Double receive buffer). This event status bit is set to indicate the completion by the CIA algorithm of the leading edge detection and its other adjustments of the receive timestamp information. The resultant adjusted message RX timestamp is then available in SET_1 register. The CIADONE0 status bit can be cleared explicitly by writing a 1 to it.
CP_ERR0 reg:01:18 bit:3	Scramble Timestamp Sequence (STS) error. The CPERR event status flag will get set if the STS_TOAST bits are non-zero on the frame that was received in RX_BUFFER_0. The CP_ERR0 status bit can be cleared explicitly by writing a 1 to it.
RXFCG1 reg:01:18 bits:4	Receiver FCS Good. This event status bit reflects the result of the frame CRC checking. When the frame has been received in RX_BUFFER_1 when operating in double buffer mode (see section 4.4 Double receive buffer). When RXFCG1 is set to 1 it indicates that the CRC check result generated on the received data matches with the 2-octet FCS sequence at the end of the frame. RXFR1 with RXFCG1 then indicates the correct reception a valid frame in RX_BUFFER_1. The RXFCG1 status bit can be cleared explicitly by writing a 1 to it.

Field	Description of fields within Sub-register 0x01:18 – RX double buffer status
RXFR1 reg:01:18 bit:5	Receiver Data Frame Ready. This event status bit is set to indicate that the completion of the frame reception process of a frame in RX_BUFFER_1 when operating in double buffer mode (see section 4.4 Double receive buffer). It is expected that this will be used as the main "Receive" (interrupt) event signalling the completion of a frame reception in double buffer mode, and, that the receive event processing routine will examine the RXFCG1 and determine whether the frame has been received correctly, and also to check the CIADONE1 event status flag to validate the receive timestamp information. The RXFR1 status bit can be cleared explicitly by writing a 1 to it.
CIADONE1 reg:01:18 bit:6	CIA processing done on the CIR relating to a message in RX_BUFFER_1 when operating in double buffer mode (see section 4.4 Double receive buffer). This event status bit is set to indicate the completion by the CIA algorithm of the leading edge detection and its other adjustments of the receive timestamp information. The resultant adjusted message RX timestamp is then available in SET_2 register. The CIADONE1 status bit can be cleared explicitly by writing a 1 to it.
CP_ERR1 reg:01:18 bit:7	Scramble Timestamp Sequence (STS) error. The CPERR event status flag will get set if the STS_TOAST bits are non-zero on the frame that was received in RX_BUFFER_1. The CP_ERR1 status bit can be cleared explicitly by writing a 1 to it.

8.2.2.26 Sub-register 0x01:1C – RX double buffer event mask

ID	Length (octets)	Type	Mnemonic	Description
01:1C	1	RW	RDB_MASK	RX double buffer event mask

Register files: 0x00 and 0x1 – General configuration registers sub-register 0x1C of register file 0x01 is a mask register used in double buffer mode for enabling interrupts associated with the events reported by the RDB_STATUS register. It is assumed that in double buffer mode the receiver auto RX re-enable is enabled, see RXAUTR. This means that the receiver status events are cleared in the main status register (SYS_STATUS), after reception of one frame as the receiver is re-enabled and proceeds to receive the next frame, and thus they cannot be used for interrupt generation. Instead these RDB_MASK enables are used.

More details on the operation of double buffering are given in section 4.4 Double Receive Buffer. The RDB_MASK register contains the following bitmapped sub-fields:

REG:01:1C – RDB_MASK – RX double buffer event mask

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								CPERR1_EN	CIADONE1_EN	RXFR1_EN	RXFCG1_EN	CPERR0_EN	CIADONE0_EN	RXFR0_EN	RXFCG0_EN
																								0	0	0	0	0	0	0	0

The individual RDB_MASK register sub-fields are described below:

Field	Description of fields within Sub-register 0x01:1C – RX Double Buffer Event Mask
RXFCG0_EN reg:01:1C bit:0	Receiver FCS Good (RX_BUFFER_0) Event Mask. When enabled (set to 1). It will allow RXFCG0 status bit to generate an interrupt.
RXFR0_EN reg:01:1C bit:1	Receiver Data Frame Ready (RX_BUFFER_0) Event Mask. When enabled (set to 1). It will allow RXFR0 status bit to generate an interrupt.
CIADONE0_EN reg:01:1C bit:2	CIA processing done (RX_BUFFER_0) Event Mask. When enabled (set to 1). It will allow CIADONE0 status bit to generate an interrupt.
CPERR0_EN reg:01:1C bit:3	Scramble Timestamp Sequence (STS) error (RX_BUFFER_0) Event Mask. When enabled (set to 1). It will allow CP_ERR0 status bit to generate an interrupt.
RXFCG1_EN reg:01:1C bits:4	Receiver FCS Good (RX_BUFFER_1) Event Mask. When enabled (set to 1). It will allow RXFCG1 status bit to generate an interrupt.
RXFR1_EN reg:01:1C bit:5	Receiver Data Frame Ready (RX_BUFFER_1) Event Mask. When enabled (set to 1). It will allow RXFR1 status bit to generate an interrupt.
CIADONE1_EN reg:01:1C bit:6	CIA processing done (RX_BUFFER_1) Event Mask. When enabled (set to 1). It will allow CIADONE1 status bit to generate an interrupt.
CPERR1_EN reg:01:1C bit:7	Scramble Timestamp Sequence (STS) error (RX_BUFFER_1) Event Mask. When enabled (set to 1). It will allow CP_ERR1 status bit to generate an interrupt.

8.2.2.27 Sub-register 0x01:20 – RX double buffer diagnostic configuration

ID	Length (octets)	Type	Mnemonic	Description
01:20	1	RW	RDB_DIAG	RX double buffer diagnostic configuration

Register file: 0x00-0x1 – General configuration registers, sub-register 0x20 of register file 0x01 is a configuration register used for specifying the level of diagnostic data available after each receive event when operating in double buffer mode. More details on the operation of double buffering are given in section 4.4 Double receive buffer. The RDB_DIAG register contains the following bitmapped sub-fields:

REG:01:20 – RDB_DIAG – RX double buffer diagnostic configuration																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	-	-	RDB_DMODE		
																								0	0	0	0	0	0	0	0

The individual RDB_DIAG register sub-fields are described below:

Field	Description of fields within Sub-register 0x01:20 – RX double buffer diagnostic configuration
RDB_DMODE reg:01:20 bits:2-0	RX double buffer diagnostic mode. The are 3 levels of diagnostic data that can be logged while RX double buffer mode is enabled: 0x1 – Minimal set (logs 7 registers as shown in Table 47) 0x2 – Medium set (logs 13 registers as shown in Table 47) and 0x4 – Full set. (logs all registers as shown in Table 47) More details on the operation of double buffering are given in section 4.4 Double receive buffer. **Note:** The registers saved into the double buffer sets (SET_1 and SET_2) will depend on the CIA diagnostic level as set by MINDIAG.
- reg:01:20 bits:7–3	Bits marked '-' are reserved.

8.2.2.28 Sub-register 0x01:30 – AES configuration

ID	Length (octets)	Type	Mnemonic	Description
01:30	2	RW	AES_CFG	AES configuration

Register file: 0x00-0x1 – General configuration registers, sub-register 0x30 of register file 0x01 is a configuration register used for the configuration of the QM33100 AES block. The AES_CFG register contains the following bitmapped sub-fields:

REG:01:30 – AES_CFG – AES configuration																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	KEY_OTP_SEL	KEY_OTP	CORE_SEL	TAG_SIZE			KEY_SRC	KEY_LOAD	KEY_ADDR			KEY_SIZE		MODE
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The individual AES_CFG register sub-fields are described below:

Field	Description of fields within Sub-register 0x01:30 – AES configuration
MODE reg:01:30 bit:0	Mode of operation of AES core 0: Encrypt – instructs the AES core to perform encryption on the input data 1: Decrypt– instructs the AES core to perform decryption on the input data
KEY_SIZE reg:01:30 bits:2-1	AES Key Size, configures the size of the KEY to be used by the AES engine. 0x0: 128 bits 0x1: 192 bits 0x2: 256 bits 0x3 Unused and should not be configured
KEY_ADDR reg:01:30 bits:5-3	Address offset of AES KEY when using one of the AES KEY from the AES RAM (8 x 128 bits)
KEY_LOAD reg:01:30 bit:6	Load the AES KEY from AES KEY source (e.g. AES_KEY) into the AES core engine. This is done at the start of a session and all packets in the same session use the same pre-loaded key. When using CCM mode, AES key should be loaded each time prior to starting of AES engine. This key update is needed even if the key remains the same. **NOTE:** **When using KEY stored in AES_KEY_RAM or OTP, the KEY must be stored in big endian format. When using KEY in AES_KEY register the KEY is stored in little endian format.**
KEY_SRC reg:01:30 bit:7	AES key source: This bit selects if the AES KEY to be used for AES operation is the one stored in AES_KEY register (when this bit is set to 0) or (when this bit is set to 1) one of the KEYs stored in OTP memory or in AES RAM, as specified by the KEYOTP bit.
TAG_SIZE reg:01:30 bits:10-8	Size of AES tag field. This is also known as the message integrity code size field. The AES tag field can be configured to be one of the following: 0x0: 0 bytes 0x1: 4 bytes 0x2: 6 bytes 0x3: 8 bytes 0x4: 10 bytes 0x5: 12 bytes 0x6: 14 bytes 0x7: 16 bytes
CORE_SEL reg:01:30 bit:11	AES Core select. QM33100 supports two AES engine cores: GCM and CCM*. This bit selects which AES core is used, when set to 0, the GCM is used; when set to 1 CCM* is used.
KEY_OTP reg:01:30 bit:12	AES key Memory source. When this bit is set to 0, the AES KEY used for the AES operation is taken from RAM (can be one of the 8 AES KEYs as specified by the KEYADDR) or when set to 1 the AES KEY will be the one stored in the OTP (see 7.3.1 memory map). Note KEYSRC needs to also be set to 1.
KEY_OTP_SEL reg:01:30 bit:13	OTP key selecton bit. When this is set to 0 and KEY_OTP is set to 1, then the KEY from OTP location 0x78 is used, otherwise when this is set to 1 and KEY_OPT is set to 1, the KEY from OTP location 0x7C is used.
- reg:01:30 bits:15–14	Bits marked '-' are reserved.

8.2.2.29 Sub-register 0x01:34 – AES IV0

ID	Length (octets)	Type	Mnemonic	Description
01:34	4	RW	AES_IV0	The 1st IV 32-bit word for AES GCM core mode, or when AES-CCM* core is used, the 4-bytes of the 13 byte nonce (e.g. iv[12]) should be written into this register: i.e. iv[10], iv[9], iv[8], iv[7], where iv[10] is the least significant byte.

Register file: 0x00-0x1 – General configuration registers, sub-register 0x34 of register file 0x01, this is the 1st IV 32-bit word for AES GCM core mode, or when AES-CCM* core is used, the 4-bytes of the 13 byte nonce (e.g. iv[12]) should be written into this register: i.e. iv[10], iv[9], iv[8], iv[7], where iv[10] is the least significant byte.

8.2.2.30 Sub-register 0x01:38 – AES IV1

ID	Length (octets)	Type	Mnemonic	Description
01:38	4	RW	AES_IV1	The 2nd IV 32-bit word for AES GCM core mode, or when AES-CCM* core is used, the 4-bytes of the 13 byte nonce (e.g. iv[12]) should be written into this register: i.e. iv[6], iv[5], iv[4], iv[3], where iv[6] is the least significant byte

Register file: 0x00-0x1 – General configuration registers, sub-register 0x38 of register file 0x01, this is the 2nd IV 32-bit word for AES GCM core mode, or when AES-CCM* core is used, the 4-bytes of the 13 byte nonce (e.g. iv[12]) should be written into this register: i.e. iv[6], iv[5], iv[4], iv[3], where iv[6] is the least significant byte.

8.2.2.31 Sub-register 0x01:3C – AES IV2

ID	Length (octets)	Type	Mnemonic	Description
01:3C	4	RW	AES_IV2	The 3rd IV 32-bit word for AES GCM core mode, or when AES-CCM* core is used, the 4-bytes of the 13 byte nonce (e.g. iv[12]) should be written into this register: i.e. iv[2], iv[1], iv[0], don't care, where iv[2] is the least significant byte

Register file: 0x00-0x1 – General configuration registers, sub-register 0x3C of register file 0x01 this is the 3rd IV 32-bit word for AES GCM core mode, or when AES-CCM* core is used, the 4-bytes of the 13 byte nonce (e.g. iv[12]) should be written into this register: i.e. iv[2], iv[1], iv[0], don't care, where iv[2] is the least significant byte.

8.2.2.32 Sub-register 0x01:40 – AES IV3

ID	Length (octets)	Type	Mnemonic	Description
01:40	2	RW	AES_IV3	When AES-CCM* core is used, the payload length needs to be programmed here. This is not used in the AES GCM core mode.

Register file: 0x00-0x1 – General configuration registers, sub-register 0x40 of register file 0x01, is a 16-bit register which needs to contain the length of the payload which the AES engine is going to decrypt/encrypt. This register is not used in the AES GCM core mode.

8.2.2.33 Sub-register 0x01:42 – AES IV4

ID	Length (octets)	Type	Mnemonic	Description
01:42	2	RW	AES_IV4	The two bytes of the 13 byte nonce (e.g. iv[12]) should be written into this register: i.e. iv[12], iv[11], where iv[12] is the least significant byte. This is not used in the AES GCM core mode.

Register file: 0x00-0x1 – General configuration registers, sub-register 0x42 of register file 0x01, is a 16-bit register which contains two bytes of the 13 byte nonce (e.g. iv[12]): i.e. iv[12], iv[11], where iv[12] is the least significant byte. This register is not used in the AES GCM core mode.

8.2.2.34 Sub-register 0x01:44 – DMA configuration

ID	Length (octets)	Type	Mnemonic	Description
01:44	8	RW	DMA_CFG	The DMA configuration register

Register file: 0x00-0x1 – General configuration registers, sub-register 0x44 of register file 0x01, this is the configuration register for the AES-DMA engine. The AES-DMA Engine performs encryption/authentication of packets and transfers data from/to the RAMs/buffers.

The DMA_CFG register contains the following bitmapped sub-fields:

REG:01:44 – DMA_CFG – DMA configuration (octets 0 to 3)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	SCR_FIFO_MODE	RAM_WIPE	-	CP_END_SEL	DST_ADDR										DST_PORT			SRC_ADDR										SRC_PORT		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REG:01:48 – DMA_CFG – DMA configuration (octets 4 to 7)																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAM_SIZE	RAM_SEL				-	-	-	-	-	-	-	-	-	-	PYLD_SIZE										HDR_SIZE						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The individual DMA_CFG register sub-fields are described below:

Field	Description of fields within Sub-register 0x01:44 – DMA configuration
SRC_PORT reg:01:44 bits:2-0	Source memory port for DMA transfer 0x0: AES scratch RAM 0x1: RX buffer 0 0x2: RX buffer 1 0x3: TX buffer 0x4-0x7: Unused
SRC_ADDR reg:01:44 bits:12-3	Address offset within source memory for DMA transfer (0 – 1024 bytes)
DST_PORT reg:01:44 bits:15-13	Destination memory port for DMA transfer 0x0: AES scratch RAM. 0x1: RX buffer 0 0x2: RX buffer 1 0x3: TX buffer 0x4: STS key register 0x5-0x7: Unused
DST_ADDR reg:01:44 bits:25-16	Address offset within destination memory for DMA transfer (0 – 1024 bytes)
CP_END_SEL reg:01:44 bit:26	Select the endianess of the CP seed port. 0 = BigEndian. 1= Little Endian
RAM_WIPE reg:01:44 bit:28	RAM Wipe Enable. Used to set zero's as the input to a DMA transfer specified by the DMA_DST settings.

Field	Description of fields within Sub-register 0x01:44 – DMA configuration
SCR_FIFO_MODE reg:01:44 bit:29	Experimental mode to allow the SCR RAM to act as a FIFO.
- reg:01:44 bits:31–30	Bits marked '-' are reserved.
HDR_SIZE reg:01:48 bits:6-0	Size of header field in the packet to be transferred via the DMA (0 to 127 bytes) Note, the payload length does not include MIC (TAG_SIZE) and FCS lengths. Thus if RX or TX frame length is 100 bytes and header length is 10 bytes, MIC is 16 bytes and FCS is 2 bytes then payload length is 100-10-16-2 = 72 bytes.
PYLD_SIZE reg:01:48 bits:16:7	Size of payload field in the packet to be transferred via the DMA (0 to 1023 bytes) Note, the payload length does not include MIC (TAG_SIZE) and FCS lengths. Thus if RX or TX frame length is 100 bytes and header length is 10 bytes, MIC is 16 bytes and FCS is 2 bytes then payload length is 100-10-16-2 = 72 bytes.
RAM_SEL reg:01:48 bits:30:27	RAM selection for wipe operation: Bit 27 : RX1 Bit 28 : RX2 Bit 29 : TX Bit 30 : Scratch RAM If all zero, RAM wipe will use the DMA destination and other DMA settings.
RAM_SIZE reg:01:48 bits:31	RAM size selection for wipe operation: 0 = Use DMA_PYLD_SIZE 1 = Full wipe of all RAMs

8.2.2.35 Sub-register 0x01:4C – AES start

ID	Length (octets)	Type	Mnemonic	Description
01:4C	1	RW	AES_START	Start AES operation

Register file: 0x00-0x1 – General configuration registers, sub-register 0x4C of register file 0x01, this is the AES core operation register. The AES_START register contains the following bitmapped sub-fields:

REG:01:4C – AES_START – AES start

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	-	-	-	-	AES_START
																								0	0	0	0	0	0	0	0

The individual AES_START register sub-fields are described below:

Field	Description of fields within Sub-register 0x01:4C – AES start
AES_START reg:01:4C bit:0	Start AES operation. Set this bit to start AES operation, the bit will be cleared by AES core once operation starts.
- reg:01:4C bits:7–1	Bits marked '-' are reserved.

8.2.2.36 Sub-register 0x01:50 – AES status

ID	Length (octets)	Type	Mnemonic	Description
01:50	4	RW	AES_STS	The AES Status

Register file: 0x00-0x1 – General configuration registers, sub-register 0x50 of register file 0x01, this is the AES core status register. The AES_STS register contains the following bitmapped sub-fields:

REG:01:50 – AES_STS – AES status

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	RAM_FULL	RAM_EMPTY	MEM_CONF	TRANS_ERR	AUTH_ERR	AES_DONE
																								0	0	0	1	0	0	0	0

The individual AES_STS register sub-fields are described below:

Field	Description of fields within Sub-register 0x01:50 – AES status
AES_DONE reg:01:50 bit:0	AES operation complete. Write 1 to clear.
AUTH_ERR reg:01:50 bit:1	AES authentication error. Write 1 to clear. 0: The tag/MIC that was supplied with the encrypted data has been validated correctly. 1: The tag/MIC that was supplied with the encrypted data has not been validated correctly, therefore the protected (frame) content should not be trusted.
TRANS_ERR reg:01:50 bit:2	Indicates error with DMA transfer to memory. Write 1 to clear. This will occur when writing more data to a buffer than the buffer can hold. Size of data is too big for the destination buffer.
MEM_CONF reg:01:50 bit:3	Indicates access conflict between multiple masters (SPI host, CIA engine and AES-DMA engine) trying to access same memory. Write 1 to clear.

Field	Description of fields within Sub-register 0x01:50 – AES status
RAM_EMPTY reg:01:50 bit:4	Indicates AES scratch RAM is empty. Write 1 to clear.
RAM_FULL reg:01:50 bit:5	Indicates AES scratch RAM is full. Write 1 to clear.
- reg:01:50 bits:7–6	Bits marked '-' are reserved.

8.2.2.37 Sub-register 0x01:54 – AES KEY

ID	Length (octets)	Type	Mnemonic	Description
01:54	16	RW	AES_KEY	The 128-bit KEY for the AES GCM/CCM* core

Register file: 0x00-0x1 – General configuration registers, sub-register 0x54 of register file 0x01, this is the 128-bit key (four 32-bit words) for the AES GCM/CCM* core.

8.2.3 Register file: 0x02 – STS configuration and status registers

ID	Length (octets)	Type	Mnemonic	Description
0x02	55	-	STS_CONFIG	Scrambled Timestamp Sequence configuration and status registers

Register map register file 0x02 is concerned with the use of the QM33100 STS block. It contains a number of sub-registers. An overview of these is given by Table 24. Each of these sub-registers is separately described in the sub-sections below.

Table 24: Register file: 0x02 – STS configuration and status overview

OFFSET in Register 0x02	Mnemonic	Description
0x00	STS_CFG	STS configuration
0x04	STS_CTRL	STS control
0x08	STS_STS	STS status
0x0C	STS_KEY	STS 128-bit KEY
0x1C	STS_IV	STS 128-bit IV

8.2.3.1 Sub-register 0x02:00 – STS configuration

ID	Length (octets)	Type	Mnemonic	Description
0x02:00	2	RW	STS_CFG	The STS configuration

Register file: 0x02 – STS configuration and status , sub-register 0x00, this contains the configuration settings for STS generation. The STS_CFG register contains the following bitmapped sub-fields:

REG:02:00 – STS_CFG – STS configuration

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	-	CPS_LEN							
																0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	1

The individual STS_CFG register sub-fields are described below:

Field	Description of fields within Sub-register 0x02:00 – STS configuration
CPS_LEN reg:02:00 bit:7-0	STS length. This 8-bit field specifies the length of the STS in blocks of 8 symbols. A value of 0 selects one block of 8 symbols. The default CPS_LEN value of 7 selects a STS length of ~64 symbols (i.e. 64 x 512 chips). A value of 3 gives the minimum supported STS length of 32. The maximum CPS_LEN value 255 results in an STS length of ~2048 symbols (i.e. 2048 x 512 chips).
- reg:02:00 bits:15–8	Bits marked '-' are reserved. **N.B.: Any change in these bits field may cause the QM33100 to malfunction.**

8.2.3.2 Sub-register 0x02:04 – STS control

ID	Length (octets)	Type	Mnemonic	Description
02:04	1	RW	STS_CTRL	The STS control

Register file: 0x02 – STS configuration and status , sub-register 0x04, this contains the control bits of STS generation. The STS_CTRL register contains the following bitmapped sub-fields:

REG:02:04 – STS_CTRL – STS control																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	-	-	-	RST_LAST	LOAD_IV
																								0	0	0	0	0	0	0	0

The individual STS_CTRL register sub-fields are described below:

Field	Description of fields within Sub-register 0x02:04 – STS control
LOAD_IV reg:02:04 bit:0	Load STS_IV bit into the AES-128 block for the generation of STS. Writing 1 loads the bits. Note: CP_SPC configuration must be set before asserting this bit. This bit has priority over RST_LAST below. This is a self clearing bit. When a 1 is written to this bit it is cleared to 0 one clock cycle later.
RST_LAST reg:02:04 bit:1	Start from last, when it is set to 1 the STS generation starts from the last count that was used by the AES-128 block for the generation of the previous STS. This is a self clearing bit. When a 1 is written to this bit it is cleared to 0 one clock cycle later. **NOTE:** **If both RST_LAST and LOAD_IV are asserted simultaneously, LOAD_IV will take priority and RST_LAST will be ignored.**
- reg:02:04 bits:7–2	Bits marked '-' are reserved.

8.2.3.3 Sub-register 0x02:08 – STS status

ID	Length (octets)	Type	Mnemonic	Description
02:08	2	RW	STS_STS	The STS status

Register file: 0x02 – STS configuration and status , sub-register 0x08, this contains the status of STS reception. The STS_STS register contains the following bitmapped sub-fields:

REG:02:08 – STS_STS – STS status																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	ACC_QUAL											
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The individual STS_STS register sub-fields are described below:

Field	Description of fields within Sub-register 0x02:08 – STS status
ACC_QUAL reg:02:08 bits:11-0	STS accumulation quality. This 12-bit field reports a quality measure of the accumulation of the STS. It is very important to assess ACC_QUAL before accepting that a receive timestamp STS_TOA value has not been corrupted and is sufficiently good to be trusted in a two-way ranging exchange. Please refer to in *§ 6 – Secure ranging / timestamping* for more details on using this value.
- reg:02:08 bits:15–12	Bits marked '-' are reserved.

8.2.3.4 Sub-register 0x02:0C– STS KEY

ID	Length (octets)	Type	Mnemonic	Description
02:0C	16	RW	STS_KEY	The 128-bit KEY for the STS

Register file: 0x02 – STS configuration and status , sub-register 0x0C, this is the 128-bit key (four 32-bit words) used by the AES-128 block for the generation of the scrambled timestamp sequence (STS). (The lower order octets are at lower offset addresses). Please refer to in *§ 6 – Secure ranging / timestamping* for more details on this process. The default value is 0x14148674 0xD1D336AA 0xF86050A8 0x14EB220F.

8.2.3.5 Sub-register 0x02:1C– STS IV

ID	Length (octets)	Type	Mnemonic	Description
02:1C	16	RW	STS_IV	The 128-bit IV for the STS

Register file: 0x02 – STS configuration and status , sub-register 0x1C, this is the 128-bit IV (four 32-bit words) used by the AES-128 block for the generation of the scrambled timestamp sequence (STS). (The lower order octets are at lower offset addresses). Please refer to in *§ 6 – Secure ranging / timestamping* for more details on this process.The default value is 0x362EEB34 0xC44FA8FB 0xD37EC3CA 0x1F9A3DE4

Note: Sub-register 0x0F:4C – Counter debug contains the current value of the low 32-bits of the STS_IV.

8.2.4 Register file: 0x03 – Receiver tuning parameters

ID	Length (octets)	Type	Mnemonic	Description
0x03	107	RW	RX_TUNE	Receiver tuning parameters

Register map register file 0x03 is for control and configuration of the QM33100 receiver. The following tuning parameters should be configured depending on the channel used. They optimise the receiver

performance, when 64 MHz PRF has been configured. Other sub-registers in this Register file are reserved and should not be modified.

Note: These values are automatically loaded from OTP by the driver.

Table 25: Receiver tuning parameters

OFFSET in Register 0x03	Mnemonic	Channel 5 value	Channel 9 value
0x1C	DGC_CFG0	0x10000240	0x10000240
0x20	DGC_CFG1	0x1b6da489	0x1b6da489
0x2C	DGC_LUT_0	0x0003803e	0x0005407e
0x30	DGC_LUT_1	0x0003876e	0x000547be
0x34	DGC_LUT_2	0x000397fe	0x00054d36
0x38	DGC_LUT_3	0x00038efe	0x00055e36
0x3C	DGC_LUT_4	0x00039c7e	0x00055f36
0x40	DGC_LUT_5	0x00039dfe	0x00055df6
0x44	DGC_LUT_6	0x00039ff6	0x00055ffe

Note: When 16 MHz PRF is used, then the RX_TUNE_EN bit (bit 0 in register DGC_CFG) needs to be cleared.

8.2.4.1 Sub-register 0x03:18 – RX tune configuration

ID	Length (octets)	Type	Mnemonic	Description
03:18	2	RW	DGC_CFG	The RX tuning configuration register

Register file: 0x03 – Receiver tuning parameters, sub-register 0x18, this is a RX tune configuration register, it contains the following bitmapped sub-fields:

REG:03:18 – DGC_CFG – RX tune configuration

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	THR_64						-	-	-	-	-	-	-	-	RX_TUNE_EN
																1	1	1	0	0	1	0	1	1	1	1	0	0	1	0	1

The individual DGC_CFG register sub-fields are described below:

Field	Description of fields within Sub-register 0x03:18 – RX tune configuration
RX_TUNE_EN reg:03:18 bit:0	RX tuning enable bit. This bit should be set to 1 when 64 MHz PRF is used, and set to 0 otherwise.

Field	Description of fields within Sub-register 0x03:18 – RX tune configuration
THR_64 reg:03:18 bits:14-9	RX tuning threshold configuration for 64 MHz PRF. This should be changed from the default value of 0x38 to 0x32, to give a more optimised receiver performance.
- reg:03:18 bits: various	Bits marked '-' are reserved.

8.2.4.2 Sub-register 0x03:54 – DGC report

ID	Length (octets)	Type	Mnemonic	Description
03:54	4	RW	DGC_DBG	Reports DGC information

Register file: 0x03 – Receiver tuning parameters, sub-register 0x54, this is a DGC reporting register, it contains the following bitmapped sub-fields:

REG:03:54 – DGC_DBG – DGC_report																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	DGC_DECISION			-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The individual DGC_DBG register sub-fields are described below:

Field	Description of fields within Sub-register 0x03:54 – DGC report
DGC_DECISION reg:03:54 bits:30-28	DGC decision index.
- reg:03:54 bits: various	Bits marked '-' are reserved.

8.2.5 Register file: 0x04 – External sync control and RX calibration

ID	Length (octets)	Type	Mnemonic	Description
0x04	39	RW	EXT_SYNC	External synchronisation control and RX calibration

Register map register file 0x04 is for control of the QM33100 synchronisation hardware, and RX calibration function.

There is a separate application note giving details of the external synchronisation. Please consult with Qorvo applications support team for details. The capabilities of the QM33100 with respect to external synchronisation are described briefly in section *7.1- External Synchronisation*.

OFFSET in Register 0x04	Mnemonic	Description
0x00	EC_CTRL	External clock synchronisation counter configuration
0x0C	RX_CAL	RX calibration block configuration
0x14	RX_CAL_RESI	RX calibration block result
0x1C	RX_CAL_RESQ	RX calibration block result
0x20	RX_CAL_STS	RX calibration block status

8.2.5.1 Sub-register 0x04:00 External clock sync control

ID	Length (octets)	Type	Mnemonic	Description
04:00	4	RW	EC_CTRL	External clock synchronisation counter configuration

Register file: 0x04 – External sync control and RX calibration, sub-register 0x00 is the External clock synchronisation counter configuration register, EC_CTRL. The EC_CTRL register is used to configure the external synchronisation mode. The EC_CTRL register contains the following sub-fields:

REG:04:00 –EC_CTRL– External clock synchronisation counter configuration

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OSTR_MODE	OSTS_WAIT								-	-	-
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0

The fields of the EC_CTRL register identified above are individually described below:

Field	Description of fields within Sub-register 0x04:00 External clock sync control
- reg:04:00 bits:various	Bits marked '-' are reserved.
OSTS_WAIT reg:04:00 bits:10:3	Wait counter used for external timebase reset. See *7.1.1 – One Shot Timebase Reset (OSTR) Mode.*
OSTR_MODE reg:04:00 bit:11	External timebase reset mode enable bit. For details of use, please refer to section *7.1.1 – One Shot Timebase Reset (OSTR) Mode.*

8.2.5.2 Sub-register 0x04:0C RX calibration configuration

ID	Length (octets)	Type	Mnemonic	Description
04:0C	4	RW	RX_CAL	RX calibration block configuration

Register file: 0x04 – External sync control and RX calibration, sub-register 0x0C is RX calibration block configuration register, RX_CAL. The RX_CAL register is used to configure the RX calibration block prior to enabling the RX operation. The RX_CAL register contains the following sub-fields:

REG:04:0C –RX_CAL– RX calibration block configuration

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	COMP_DLY				-	-	-	-	-	-	-	-	-	-	-	CAL_EN	-	-	CAL_MODE	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the RX_CAL register identified above are individually described below:

Field	Description of fields within Sub-register 0x04:0C RX calibration configurartion
- reg:04:0C bits:various	Bits marked '-' are reserved.
CAL_MODE reg:04:0C bits:1:0	RX calibration mode: 0 = normal mode, should be set to this for normal receiver operation 1 = calibration mode, needs to be set when RX calibration is perfomed 2, 3 = reserved vaues
CAL_EN reg:04:0C bit:4	RX calibration enable. Set this to 1 to start RX calibration. The bit will self-clear when calibration is finished. The RX_CAL_STS will be set when calibration is complete.
COMP_DLY reg:04:0C bits:19:16	RX calibration tuning value. The host should set this to 0x2, for optimal performance. Other values should not be used.

8.2.5.3 Sub-register 0x04:14 RX calibration result I

ID	Length (octets)	Type	Mnemonic	Description
04:14	4	RW	RX_CAL_RESI	RX calibration block result I

Register file: 0x04 – External sync control and RX calibration, sub-register 0x14 is RX calibration block result register. This register reports the result once the RX calibration is complete. It is a 29-bit register. If a value of 0x1fffffff is read back then the calibration has failed and the host should not use receiver function. The

device RX calibration should be repeated. If either RX_CAL_RESI or RX_CAL_RESQ report failure the RX cal should be repeated.

8.2.5.4 Sub-register 0x04:1C RX calibration result Q

ID	Length (octets)	Type	Mnemonic	Description
04:1C	4	RW	RX_CAL_RESQ	RX calibration block result Q

Register file: 0x04 – External sync control and RX calibration, sub-register 0x1C is RX calibration block result register. This register reports the result once the RX calibration is complete. It is a 29-bit register. If a value of 0x1fffffff is read back then the calibration has failed and the host should not use receiver function. The device RX calibration should be repeated. If either RX_CAL_RESI or RX_CAL_RESQ report failure the RX cal should be repeated.

8.2.5.5 Sub-register 0x04:20 RX calibration status

ID	Length (octets)	Type	Mnemonic	Description
04:20	1	RW	RX_CAL_STS	RX calibration block status

Register file: 0x04 – External sync control and RX calibration, sub-register 0x20 is RX calibration block status register. This register reports the status once the RX calibration is complete. It is a single bit register. If bit 0 is set, the RX calibration is complete, host can write 1 to clear.

8.2.6 Register file: 0x05 – GPIO control and status

ID	Length (octets)	Type	Mnemonic	Description
0x05	47	-	GPIO_CTRL	General Purpose Input-Output control registers

Register map register file 0x05 is concerned with the use of the GPIO. It contains a number of sub-registers. An overview of these is given by Table 26. Each of these sub-registers is separately described in the sub-sections below.

Note: the GPIO clocks need to be turned on, before enabling or disabling the GPIO mode or value. The GPIO clocks are enabled by setting GPIO_CLK_EN, GPIO_DCLK_EN and GPIO_DRST_N in Sub-register 0x11:04 – Clock control

Table 26: Register file: 0x05 – GPIO control and status overview

OFFSET in Register 0x05	Mnemonic	Description
0x00	GPIO_MODE	GPIO Mode Control Register
0x08	GPIO_DIR	GPIO Direction Control Register

OFFSET in Register 0x05	Mnemonic	Description
0x0C	GPIO_OUT	GPIO Data Output Register
0x10	GPIO_IRQE	GPIO Interrupt Enable
0x14	GPIO_ISTS	GPIO Interrupt Status
0x18	GPIO_ISEN	GPIO Interrupt Sense Selection
0x1C	GPIO_IMODE	GPIO Interrupt Mode (Level / Edge)
0x20	GPIO_IBES	GPIO Interrupt “Both Edge” Select
0x24	GPIO_ICLR	GPIO Interrupt Latch Clear
0x28	GPIO_IDBE	GPIO Interrupt De-bounce Enable
0x2C	GPIO_RAW	GPIO Raw State

8.2.6.1 Sub-register 0x05:00 – GPIO mode control

ID	Length (octets)	Type	Mnemonic	Description
05:00	4	RW	GPIO_MODE	GPIO mode control register

Register file: 0x05 – GPIO control and status, sub-register 0x00 is the GPIO Mode Control Register, GPIO_MODE. The GPIO_MODE register is used to select whether the GPIO is operating as a GPIO or has another special function. The GPIO_MODE register contains the following sub-fields:

REG:05:00 – GPIO_MODE – GPIO mode control register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	COEX_IO_SWAP	MSGP8			MSGP7			MSGP6			MSGP5			MSGP4			MSGP3			MSGP2			MSGP1			MSGP0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the GPIO_MODE register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:00 – GPIO mode control
MSGP0 reg:05:00 bits:2-0	Mode Selection for GPIO0/RXOKLED. Allowed values are: 000: Reserved – This is the default (reset) state. 001: The pin operates as the RXOKLED output. 010 : The pin operates as GPIO0. 011 : The pin is configured for use as dbg_mux_0. 100 – 111: Reserved. Do not select this value. When operating as the RXOKLED driver, the output is asserted briefly when the receiver completes the reception of a frame with a good FCS/CRC. The on time for the RXOKLED depends on the blink time set in *Sub-register 0x11:18 – LED control*.Note: Lighting LEDs will drain power in battery-powered applications.

Field	Description of fields within Sub-register 0x05:00 – GPIO mode control
MSGP1 reg:05:00 bits:5-3	Mode Selection for GPIO1/SFDLED. Allowed values are: 000: Reserved– This is the default (reset) state. 001: The pin operates as the SFDLED output. 010: The pin operates as GPIO1. 011 : The pin is configured for use as dbg_mux_1. 100 – 111: Reserved. Do not select this value. When operating as the driver, the output is asserted briefly when the receiver detects the SFD sequence in the RX packet. The on time is determined by the blink time configuration set in *Sub-register 0x11:18 – LED control*Note: Lighting LEDs will drain power in battery-powered applications.
MSGP2 reg:05:00 bits:8-6	Mode Selection for GPIO2/RXLED. Allowed values are: 000: Reserved – This is the default (reset) state. 001: The pin operates as the RXLED output. 010 : The pin operates as GPIO2. 011 : The pin is configured for use as dbg_mux_2. 100 – 111: Reserved. Do not select this value. When operating as the RXLED driver, the output is asserted when the receiver is on, and stays on for a brief period after the receiver is turned off. The minimum on time is determined by the blink time configurable in *Sub-register 0x11:18 – LED control*.Note: Lighting LEDs will drain power in battery-powered applications.
MSGP3 reg:05:00 bits:11-9	Mode Selection for GPIO3/TXLED. Allowed values are: 000: Reserved– This is the default (reset) state. 001: The pin operates as the TXLED output. 010 : The pin operates as GPIO3. 011 : The pin is configured for use as dbg_mux_3. 100 – 111: Reserved. Do not select this value.When operating as the TXLED driver, the output is asserted briefly when the transmitter completes sending a packet. The blink time is configurable via *Sub-register 0x11:18 – LED control*. Note: Lighting LEDs will drain power in battery-powered applications.
MSGP4 reg:05:00 bits:14-12	Mode Selection for GPIO4/COEXIN. Allowed values are: 000: The pin operates as GPIO4– This is the default (reset) state. 001 : The pin is configured for use as COEX_IN. 010 : The pin is configured for use as AOA_SW_0. 011 : The pin is configured for use as dbg_mux_4. 100 – 111: Reserved. Do not select this value. **NOTE:** **COEX_IN is an input used to abort any ongoing TX or RX RF operations. It can be SW configured for "TX, RX or Both". This will trigger an coex_err interrupt flag to current granted host.** **When Configured as AOA_SW_0, it will stay on during TX frame, turns on approx.. 10us before first TX pulse.**

Field	Description of fields within Sub-register 0x05:00 – GPIO mode control
MSGP5 reg:05:00 bits:17-15	Mode Selection for GPIO5/COEXOUT. Allowed values are: 000: The pin operates as GPIO5 – This is the default (reset) state. 001 : The pin is configured for use as COEX_OUT. 010 : The pin is configured for use as AOA_SW_1. 011 : The pin is configured for use as dbg_mux_5. 100 – 111: Reserved. Do not select this value. **NOTE:** **COEX_OUT is an output flag that can be configured via SW to indicate "RX, TX or Both" operations are under way. Turn on time before active window is 10us.** **AOA_SW_1 : Similar to ext_sw_rx; but turns on 100ns before the start of RX.**
MSGP6 reg:05:00 bits:20-18	Mode Selection forGPIO6/EXTSWRX. Allowed values of MSGP6 are: 000: The pin operates as GPIO6– This is the default (reset) state. 001 : The pin is configured for use as ext_sw_rx. 010 : The pin is configured for use as AOA_SW_2 011 : The pin is configured for use as dbg_mux_6 100 – 111: Reserved. Do not select this value. **NOTE:** **When it's configured as AOA_SW_2, the pin will go high when device is receiving or transmitting from RF port 1, used for AoA switching (within a frame), switches within 1us STS gap.** .
MSGP7 reg:05:00 bits:23-21	Mode Selection for GPIO7. Values are: 000: Default (reset) state: The pin operates as "SYNC" input, which is a reserved test function. 001: The pin operates as GPIO7. 010 : The pin is configured for use as AOA_SW_3 011 : The pin is configured for use as dbg_mux_7 100 – 111: Reserved. Do not select this value. **NOTE:** **When it's configured as AOA_SW_3, the pin will go high when device is receiving or transmitting from RF port 2, used for AoA switching (within a frame), switches within 1us STS gap.**
MSGP8 reg:05:00 bits:26-24	Mode Selection for IRQ/GPIO8. Allowed values are: 000: The pin operates as the IRQ output – This is the default (reset) state. 001: The pin operates as GPIO8. 010 : The pin is configured for use as ext_ref 011: The pin is configured for use as dbg_mux_8 100 – 111: Reserved. Do not select this value.
COEX_IO_SWAP reg:05:00 bit:27	Default is for COEX_OUT, see mode 001 of MSGP5 above, is on GPIO5 pin and COEX_IN, see mode 001 of MSGP4 above, is on GPIO4 pin. If this bit is set to 1 then the COEX modes will be swapped, i.e. COEX_OUT will be on GPIO4 and COEX_IN on GPIO5. See also TIMER_CTRL register and COEX_OUT_MODE (in the SYS_CFG register) for further details on use of these modes.
- reg:05:00 bits:31–28	Bits marked '-' are reserved.

8.2.6.2 Sub-register 0x05:08 – GPIO direction control

ID	Length (octets)	Type	Mnemonic	Description
05:08	2	RW	GPIO_DIR	GPIO direction control register

Register file: 0x05 – GPIO control and status, sub-register 0x08 is the GPIO Direction Control Register, GPIO_DIR. The GPIO_DIR register applies to the GPIO pins when they are selected to operate as GPIOs via the GPIO_MODE register. It contains a bit for each GPIO to individually configure that GPIO as an input or an output. A GDP direction bit value of 1 means the pin is an input. A GDP bit value of 0 means the pin is an output.

The GPIO_DIR register contains the following sub-fields:

REG:05:08 – GPIO_DIR – GPIO direction control register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	GDP8	GDP7	GDP6	GDP5	GDP4	GDP3	GDP2	GDP1	GDP0
																0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

The fields of the GPIO_DIR register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:08 – GPIO direction
GDP0 reg:05:08 bit:0	Direction Selection for GPIO0. Reading this bit shows the direction setting for GPIO0.Values: 1 = input, 0 = output.
GDP1 reg:05:08 bit:1	Direction Selection for GPIO1. (See GDP0).
GDP2 reg:05:08 bit:2	Direction Selection for GPIO2. (See GDP0).
GDP3 reg:05:08 bit:3	Direction Selection for GPIO3. (See GDP0).
GDP4 reg:05:08 bit:4	Direction Selection for GPIO4. (See GDP0).
GDP5 reg:05:08 bit:5	Direction Selection for GPIO5. (See GDP0).
GDP6 reg:05:08 bit:6	Direction Selection for GPIO6. (See GDP0).
GDP7 reg:05:08 bit:7	Direction Selection for GPIO7. (See GDP0).

Field	Description of fields within Sub-register 0x05:08 – GPIO direction
GDP8 reg:05:08 bit:8	Direction Selection for GPIO8. (See GDP0).
- reg:05:08 bits:15–9	Bits marked '-' are reserved.

8.2.6.3 Sub-register 0x05:0C – GPIO data output configuration

ID	Length (octets)	Type	Mnemonic	Description
05:0C	2	RW	GPIO_OUT	GPIO data output configuration register

Register file: 0x05 – GPIO control and status, sub-register 0x0C is the GPIO data output register. The GPIO_OUT register applies to the GPIO pins when they are selected to operate as GPIO outputs via the GPIO_MODE and GPIO_DIR registers. It contains a bit for each GPIO pin to individually select the data to output on the GPIO output pin. When reading from the GPIO_OUT register output value bits will show the current output setting for the GPIO pins. Note, this does not mean this is being output since that depends also on the appropriate selection of the GPIO_MODE and GPIO_DIR registers.

The GPIO_OUT register contains the following sub-fields:

REG:05:0C – GPIO_OUT – GPIO data output configuration

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	GOP8	GOP7	GOP6	GOP5	GOP4	GOP3	GOP2	GOP1	GOP0
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the GPIO_OUT register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:0C – GPIO data output configuration
GOP0 reg:05:0C bit:0	Output state setting for the GPIO0 output. Reading this bit shows the current setting for GPIO0. Value: 1 = logic 1 voltage high output and, value 0 = logic 1 voltage low output.
GOP1 reg:05:0C bit:1	Output state setting for GPIO1. (See GOP0).
GOP2 reg:05:0C bit:2	Output state setting for GPIO2. (See GOP0).
GOP3 reg:05:0C bit:3	Output state setting for GPIO3. (See GOP0).
GOP4 reg:05:0C bit:4	Output state setting for GPIO4. (See GOP0).
GOP5 reg:05:0C bit:5	Output state setting for GPIO5. (See GOP0).

Field	Description of fields within Sub-register 0x05:0C – GPIO data output configuration
GOP6 reg:05:0C bit:6	Output state setting for GPIO6. (See GOP0).
GOP7 reg:05:0C bit:7	Output state setting for GPIO7. (See GOP0).
GOP8 reg:05:0C bit:8	Output state setting for GPIO8. (See GOP0).
- reg:05:08 bits:15–9	Bits marked '-' are reserved.

8.2.6.4 Sub-register 0x05:10 – GPIO interrupt enable

ID	Length (octets)	Type	Mnemonic	Description
05:10	2	RW	GPIO_IRQE	GPIO interrupt enable

Register file: 0x05 – GPIO control and status, sub-register 0x10 is the GPIO interrupt enable register. The GPIO_IRQE register allows a GPIO input pin to be selected as an interrupt source into the QM33100. Additional configuration registers GPIO_IMODE, GPIO_ISEN, GPIO_IBES and GPIO_IDBE allow the interrupt to be set as level sensitive with control of whether it is the low or high state that generates the interrupt, or as edge sensitive with control of the edge(s) that generates the interrupt, and includes a configurable de-bounce circuit that can be used to ignore transients on the input. The GPIO_IRQE register contains a bit for each GPIO pin to allow each to be individually selected as interrupt source. Setting the appropriate bit to 1 enables the corresponding GPIO input as an interrupt source, a value of 0 disables that interrupt. When a GPIO interrupt is triggered it is signalled to the host via the GPIOIRQ event status bit in *Sub-register 0x00:44 – System event status*. The bits of the GPIO_IRQE register are as following:

REG:05:10 – GPIO_IRQE – GPIO interrupt enable register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	GIRQE8	GIRQE7	GIRQE6	GIRQE5	GIRQ E4	GIRQ E3	GIRQE2	GIRQE1	GIRQE0
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The bits identified above are individually described below:

Field	Description of fields within Sub-register 0x05:10 – GPIO interrupt enable
GIRQE0 reg:05:10 bit:0	GPIO IRQ Enable for GPIO0 input. Value 1 = enable GPIO input GPIO0 as an interrupt source. Value 0 = GPIO0 is not an interrupt source.
GIRQE1 reg:05:10 bit:1	GPIO IRQ Enable for GPIO1 input. Value 1 = enable, 0 = disable.

Field	Description of fields within Sub-register 0x05:10 – GPIO interrupt enable
GIRQE2 reg:05:10 bit:2	GPIO IRQ Enable for GPIO2 input. Value 1 = enable, 0 = disable.
GIRQE3 reg:05:10 bit:3	GPIO IRQ Enable for GPIO3 input. Value 1 = enable, 0 = disable.
GIRQE4 reg:05:10 bit:4	GPIO IRQ Enable for GPIO4 input. Value 1 = enable, 0 = disable.
GIRQE5 reg:05:10 bit:5	GPIO IRQ Enable for GPIO5 input. Value 1 = enable, 0 = disable.
GIRQE6 reg:05:10 bit:6	GPIO IRQ Enable for GPIO6 input. Value 1 = enable, 0 = disable.
GIRQE7 reg:05:10 bit:7	GPIO IRQ Enable for GPIO7 input. Value 1 = enable, 0 = disable.
GIRQE8 reg:05:10 bit:8	GPIO IRQ Enable for GPIO8 input. Value 1 = enable, 0 = disable.
- reg:05:10 Bits:15–9	Bits marked '-' are reserved and should be written as zero.

8.2.6.5 Sub-register 0x05:14 – GPIO interrupt status

ID	Length (octets)	Type	Mnemonic	Description
05:14	2	RW	GPIO_ISTS	GPIO interrupt status

Register file: 0x05 – GPIO control and status, sub-register 0x14 is the GPIO interrupt status register. The GPIO_ISTS register acts to set the state/event that gives rise to a GPI interrupt. Assuming that the GPIO is an input and that it is enabled as an interrupt via the GPIO_IRQE register, then the GPIO_IMODE register together with GPIO_ISEN selects whether the interrupt is level or edge sensitive, and this GPIO_ISTS register selects which level or edge is the state/event that causes the interrupt. The GPIO_ISTS bits are as follows:

REG:05:14 – GPIO_ISTS– GPIO interrupt status

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	GIST8	GIST7	GIST6	GIST5	GIST4	GIST3	GIST2	GIST1	GIST0
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The bits of the GPIO_ISTS register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:14 – GPIO interrupt status
GISTS0 reg:05:14 bit:0	Value 1 means GPIO0 gave rise to the GPIOIRQ SYS_STATUS event.
GISTS1 reg:05:14 bit:1	Value 1 means GPIO1 gave rise to the GPIOIRQ SYS_STATUS event.
GISTS2 reg:05:14 bit:2	Value 1 means GPIO2 gave rise to the GPIOIRQ SYS_STATUS event.
GISTS3 reg:05:14 bit:3	Value 1 means GPIO3 gave rise to the GPIOIRQ SYS_STATUS event.
GISTS4 reg:05:14 bit:4	Value 1 means GPIO4 gave rise to the GPIOIRQ SYS_STATUS event..
GISTS5 reg:05:14 bit:5	Value 1 means GPIO5 gave rise to the GPIOIRQ SYS_STATUS event.
GISTS6 reg:05:14 bit:6	Value 1 means GPIO6 gave rise to the GPIOIRQ SYS_STATUS event.
GISTS7 reg:05:14 bit:7	Value 1 means GPIO7 gave rise to the GPIOIRQ SYS_STATUS event.
GISTS8 reg:05:14 bit:8	Value 1 means GPIO8 gave rise to the GPIOIRQ SYS_STATUS event.
- reg:05:14 bits:15–9	Bits marked '-' are reserved and should be written as zero.

8.2.6.6 Sub-register 0x05:18 – GPIO interrupt sense selection

ID	Length (octets)	Type	Mnemonic	Description
05:18	2	RW	GPIO_ISEN	GPIO interrupt sense selection

Register file: 0x05 – GPIO control and status, sub-register 0x18 is the GPIO interrupt sense selection register. The GPIO_ISEN register acts to set the state/event that gives rise to a GPI interrupt. Assuming that the GPIO is an input and that it is enabled as an interrupt via the GPIO_IRQE register, then the GPIO_IMODE register selects whether the interrupt is level or edge sensitive, and this register GPIO_ISENGPIO_ISEN selects which level or edge is the state/event that causes the interrupt. The GPIO_ISEN register contains a bit for each GPIO pin to allow each to be individually configured. The bits are as follows:

REG:05:18 – GPIO_ISEN – GPIO interrupt sense selection register																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	GISEN8	GISEN7	GISEN6	GISEN5	GISEN4	GISEN3	GISEN2	GISEN1	GISEN0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The bits of the GPIO_ISEN register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:18 – GPIO interrupt sense selection
GISEN0 reg:05:18 bit:0	GPIO IRQ Sense selection GPIO0 input. Value 0 = Active high level sensitive interrupt or rising-edge triggered interrupt. Value 1 = Active low level sensitive interrupt or falling-edge triggered interrupt.
GISEN1 reg:05:18 bit:1	GPIO IRQ sense for GPIO1 input. Value 0 = High or Rising-Edge, 1 = Low or falling-edge.
GISEN2 reg:05:18 bit:2	GPIO IRQ sense for GPIO2 input. Value 0 = High or Rising-Edge, 1 = Low or falling-edge.
GISEN3 reg:05:18 bit:3	GPIO IRQ sense for GPIO3 input. Value 0 = High or Rising-Edge, 1 = Low or falling-edge.
GISEN4 reg:05:18 bit:4	GPIO IRQ sense for GPIO4 input. Value 0 = High or Rising-Edge, 1 = Low or falling-edge.
GISEN5 reg:05:18 bit:5	GPIO IRQ sense for GPIO5 input. Value 0 = High or Rising-Edge, 1 = Low or falling-edge.
GISEN6 reg:05:18 bit:6	GPIO IRQ sense for GPIO6 input. Value 0 = High or Rising-Edge, 1 = Low or falling-edge.
GISEN7 reg:05:18 bit:7	GPIO IRQ sense for GPIO7 input. Value 0 = High or Rising-Edge, 1 = Low or falling-edge.
GISEN8 reg:05:18 bit:8	GPIO IRQ sense for GPIO8 input. Value 0 = High or Rising-Edge, 1 = Low or falling-edge.
- reg:05:18 bits:31–9	Bits marked '-' are reserved and should be written as zero.

8.2.6.7 Sub-register 0x05:1C– GPIO interrupt mode selection

ID	Length (octets)	Type	Mnemonic	Description
05:1C	2	RW	GPIO_IMODE	GPIO interrupt mode (level/edge)

Register file: 0x05 – GPIO control and status, sub-register 0x1C is the GPIO interrupt mode selection register. Assuming that the GPIO is an input and enabled as an interrupt via the GPIO_IRQE register, then this

GPIO_IMODE register acts to select whether the interrupt is level sensitive or edge triggered. The GPIO_IMODE register contains a bit for each GPIO pin to allow each to be individually configured. The bits are as follows:

REG:05:1C – GPIO_IMODE – GPIO interrupt mode selection register																												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	GIMOD8	GIMOD7	GIMOD6	GIMOD5	GIMOD4	GIMOD3	GIMOD2	GIMOD1	GIMOD0
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The bits of the GPIO_IMODE register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:1C– GPIO interrupt mode
GIMOD0 reg:05:1C bit:0	GPIO IRQ Mode selection for GPIO0 input. Value 0 = Level sensitive interrupt. Value 1 = Edge triggered interrupt.
GIMOD1 reg:05:1C bit:1	GPIO IRQ Mode selection for GPIO1 input. Value 0 = Level, 1 = Edge.
GIMOD2 reg:05:1C bit:2	GPIO IRQ Mode selection for GPIO2 input. Value 0 = Level, 1 = Edge.
GIMOD3 reg:05:1C bit:3	GPIO IRQ Mode selection for GPIO3 input. Value 0 = Level, 1 = Edge.
GIMOD4 reg:05:1C bit:4	GPIO IRQ Mode selection for GPIO4 input. Value 0 = Level, 1 = Edge.
GIMOD5 reg:05:1C bit:5	GPIO IRQ Mode selection for GPIO5 input. Value 0 = Level, 1 = Edge.
GIMOD6 reg:05:1C bit:6	GPIO IRQ Mode selection for GPIO6 input. Value 0 = Level, 1 = Edge.
GIMOD7 reg:05:1C bit:7	GPIO IRQ Mode selection for GPIO7 input. Value 0 = Level, 1 = Edge.
GIMOD8 reg:05:1C bit:8	GPIO IRQ Mode selection for GPIO8 input. Value 0 = Level, 1 = Edge.
- reg:05:1C bits:15–9	Bits marked '-' are reserved and should be written as zero.

8.2.6.8 Sub-register 0x05:20 – GPIO interrupt “Both-Edge” selection

ID	Length (octets)	Type	Mnemonic	Description
05:20	2	RW	GPIO_IBES	GPIO interrupt “Both Edge” selection

Register file: 0x05 – GPIO control and status, sub-register 0x20 is the GPIO interrupt “Both Edge” selection register. This only applies when edge sensitive interrupts are enabled in the GPIO_IMODE register. In this case the GPIO_ISEN register normally acts to select which edge triggers the interrupt. This GPIO_IBES register overrides the GPIO_ISEN register to select both edges as which edge triggers the interrupt. The GPIO_IBES register contains a bit for each GPIO pin to allow each to be individually configured. The bits are as follows:

REG:05:20 – GPIO_IBES – GPIO interrupt “Both Edge” selection register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	GIBES8	GIBES7	GIBES6	GIBES5	GIBES4	GIBES3	GIBES2	GIBES1	GIBES0
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The bits of the GPIO_IBES register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:20 – GPIO interrupt “Both-Edge” selection
GIBES0 reg:05:20 bit:0	GPIO IRQ “Both Edge” selection for GPIO0 input. Value 0 = GPIO_IMODE register selects the edge. Value 1 = Both edges trigger the interrupt.
GIBES1 reg:05:20 bit:1	GPIO IRQ “Both Edge” selection for GPIO1 input. Value 0 = use GPIO_IMODE, 1 = Both Edges.
GIBES2 reg:05:20 bit:2	GPIO IRQ “Both Edge” selection for GPIO2 input. Value 0 = use GPIO_IMODE, 1 = Both Edges.
GIBES3 reg:05:20 bit:3	GPIO IRQ “Both Edge” selection for GPIO3 input. Value 0 = use GPIO_IMODE, 1 = Both Edges.
GIBES4 reg:05:20 bit:4	GPIO IRQ “Both Edge” selection for GPIO4 input. Value 0 = use GPIO_IMODE, 1 = Both Edges.
GIBES5 reg:05:20 bit:5	GPIO IRQ “Both Edge” selection for GPIO5 input. Value 0 = use GPIO_IMODE, 1 = Both Edges.
GIBES6 reg:05:20 bit:6	GPIO IRQ “Both Edge” selection for GPIO6 input. Value 0 = use GPIO_IMODE, 1 = Both Edges.
GIBES7 reg:05:20 bit:7	GPIO IRQ “Both Edge” selection for GPIO7 input. Value 0 = use GPIO_IMODE, 1 = Both Edges.
GIBES8 reg:05:20 bit:8	GPIO IRQ “Both Edge” selection for GPIO8 input. Value 0 = use GPIO_IMODE, 1 = Both Edges.

Field	Description of fields within Sub-register 0x05:20 – GPIO interrupt “Both-Edge” selection
- reg:05:20 bits:15–9	Bits marked ‘-’ are reserved and should be written as zero.

8.2.6.9 Sub-register 0x05:24 – GPIO interrupt latch clear

ID	Length (octets)	Type	Mnemonic	Description
05:24	4	RW	GPIO_ICLR	GPIO interrupt latch clear

Register file: 0x05 – GPIO control and status, sub-register 0x24 is the GPIO interrupt clear register. When a GPIO interrupt occurs that meets the configured criteria (edge/level etc.) that event is latched in an internal interrupt latch. To clear the interrupt the host needs to write a 1 to the appropriate bit of this GPIO_ICLR register. There is no way to read the interrupt latch, which means that only one GPIO can be enabled to interrupt at a time, unless the host has some other external way to distinguish events. Although level sensitive interrupts are latched, if the active level persists, then clearing the latch will be ineffective, since the interrupt will occur again immediately. The GPIO_ICLR register contains a bit for each GPIO pin as follows:

REG:05:24 – GPIO_ICLR – GPIO interrupt latch clear

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	GICLR8	GICLR7	GICLR6	GICLR5	GICLR4	GICLR3	GICLR2	GICLR1	GICLR0
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The bits of the GPIO_ICLR register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:24 – GPIO interrupt latch clear
GICLR0 reg:05:24 bit:0	GPIO IRQ latch clear for GPIO0 input. Write 1 to clear the GPIO0 interrupt latch. Writing 0 has no effect. Reading returns zero.
GICLR1 reg:05:24 bit:1	GPIO IRQ latch clear for GPIO1 input. Write 1 to clear the interrupt latch.
GICLR2 reg:05:24 bit:2	GPIO IRQ latch clear for GPIO2 input. Write 1 to clear the interrupt latch.
GICLR3 reg:05:24 bit:3	GPIO IRQ latch clear for GPIO3 input. Write 1 to clear the interrupt latch.
GICLR4 reg:05:24 bit:4	GPIO IRQ latch clear for GPIO4 input. Write 1 to clear the interrupt latch.
GICLR5 reg:05:24 bit:5	GPIO IRQ latch clear for GPIO5 input. Write 1 to clear the interrupt latch.

Field	Description of fields within Sub-register 0x05:24 – GPIO interrupt latch clear
GICLR6 reg:05:24 bit:6	GPIO IRQ latch clear for GPIO6 input. Write 1 to clear the interrupt latch.
GICLR7 reg:05:24 bit:7	GPIO IRQ latch clear for GPIO7 input. Write 1 to clear the interrupt latch.
GICLR8 reg:05:24 bit:8	GPIO IRQ latch clear for GPIO8 input. Write 1 to clear the interrupt latch.
- reg:05:24 bits:15–9	Bits marked '-' are reserved and should be written as zero.

8.2.6.10 Sub-register 0x05:28 – GPIO interrupt de-bounce enable

ID	Length (octets)	Type	Mnemonic	Description
05:28	4	RW	GPIO_IDBE	GPIO interrupt de-bounce enable

Register file: 0x05 – GPIO control and status, sub-register 0x28 is the GPIO interrupt de-bounce enable register. The GPIO_IDBE controls a filtering function that operates on the GPIO inputs prior to their presentation into the GPIO interrupt logic. This de-bounce filter circuit removes short transients by using the kilohertz clock (as enabled by the LP_CLK_EN bit in *Sub-register 0x11:04 – Clock control*) to sample the input signal. See LP_CLK_DIV in *Sub-register 0x11:08 – Sequencing control* for a description of the kilohertz clock. The de-bounce filter is active when a state change of the GPIO input needs to persist for two cycles of this clock before it will be seen by the interrupt handling logic. The GPIO_IDBE register contains a bit for each GPIO pin as follows:

REG:05:28 – GPIO_IDBE – GPIO interrupt de-bounce enable

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	GIDBE8	GIDBE7	GIDBE6	GIDBE5	GIDBE4	GIDBE3	GIDBE2	GIDBE1	GIDBE0
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The bits of the GPIO_IDBE register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:28 – GPIO interrupt de-bounce enable
GIDBE0 reg:05:28 bit:0	GPIO IRQ de-bounce enable for GPIO0. Value 1 = de-bounce enabled. Value 0 = de-bounce disabled.
GIDBE1 reg:05:28 bit:1	GPIO1 IRQ de-bounce configuration. Value 1 = de-bounce enabled, 0 = de-bounce disabled.
GIDBE2 reg:05:28 bit:2	GPIO2 IRQ de-bounce configuration. Value 1 = de-bounce enabled, 0 = de-bounce disabled.

Field	Description of fields within Sub-register 0x05:28 – GPIO interrupt de-bounce enable
GIDBE3 reg:05:28 bit:3	GPIO3 IRQ de-bounce configuration. Value 1 = de-bounce enabled, 0 = de-bounce disabled.
GIDBE4 reg:05:28 bit:4	GPIO4 IRQ de-bounce configuration. Value 1 = de-bounce enabled, 0 = de-bounce disabled.
GIDBE5 reg:05:28 bit:5	GPIO5 IRQ de-bounce configuration. Value 1 = de-bounce enabled, 0 = de-bounce disabled.
GIDBE6 reg:05:28 bit:6	GPIO6 IRQ de-bounce configuration. Value 1 = de-bounce enabled, 0 = de-bounce disabled.
GIDBE7 reg:05:28 bit:7	GPIO7 IRQ de-bounce configuration. Value 1 = de-bounce enabled, 0 = de-bounce disabled.
GIDBE8 reg:05:28 bit:8	GPIO8 IRQ de-bounce configuration. Value 1 = de-bounce enabled, 0 = de-bounce disabled.
- reg:05:28 bits:15–9	Bits marked '-' are reserved and should be written as zero.

8.2.6.11 Sub-register 0x05:2C – GPIO raw state

ID	Length (octets)	Type	Mnemonic	Description
05:2C	2	RO	GPIO_RAW	GPIO raw state

Register file: 0x05 – GPIO control and status, sub-register 0x2C allows the raw state of the GPIO pin to be read. The GPIO_RAW register contains a bit for each GPIO pin as follows:

REG:05:2C – GPIO_RAW – GPIO raw state

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	GRAWP8	GRAWP7	GRAWP6	GRAWP5	GRAWP4	GRAWP3	GRAWP2	GRAWP1	GRAWP0
																-	-	-	-	-	-	-	0	0	0	0	0	0	0	0	0

The bits of the GPIO_RAW register identified above are individually described below:

Field	Description of fields within Sub-register 0x05:2C – GPIO raw state
GRAWP0 reg:05:2C bit:0	This bit reflects the raw state of GPIO0. It samples the input present on the GPIO, i.e. 0 or 1.
GRAWP1 reg:05:2C bit:1	GPIO1 port raw state.

Field	Description of fields within Sub-register 0x05:2C – GPIO raw state
GRAWP2 reg:05:2C bit:2	GPIO2 port raw state.
GRAWP3 reg:05:2C bit:3	GPIO3 port raw state.
GRAWP4 reg:05:2C bit:4	GPIO4 port raw state.
GRAWP5 reg:05:2C bit:5	GPIO5 port raw state.
GRAWP6 reg:05:2C bit:6	GPIO6 port raw state.
GRAWP7 reg:05:2C bit:7	GPIO7 port raw state.
GRAWP8 reg:05:2C bit:8	GPIO8 port raw state.
- reg:05:2C bits:15–9	Bits marked '-' are reserved and should be written as zero.

8.2.7 Register file: 0x06 – Digital receiver configuration

ID	Length (octets)	Type	Mnemonic	Description
0x06	60	-	DRX_CONF	Digital receiver configuration

Register map register file 0x06 is concerned with the low-level digital receiver configuration. It contains a number of sub-registers. An overview of these is given by Table 27. Each of these sub-registers is separately described in the sub-sections below.

Table 27: Register file: 0x06 – Digital receiver configuration overview

OFFSET in Register 0x06	Mnemonic	Description
0x00	DTUNE0	PAC configuration
0x02	RX_SFD_TOC	SFD timeout
0x04	PRE_TOC	Preamble detection timeout
0x0C	DTUNE3	Receiver tuning register
0x10	DTUNE4	Digital Tuning Reserved register
0x14	DTUNE_5	Digital Tuning Reserved register
0x29	DRX_CAR_INT	Carrier recovery integrator register

8.2.7.1 Sub-register 0x06:00 – Digital RX tuning register 0

ID	Length (octets)	Type	Mnemonic	Description
06:00	2	RW	DTUNE0	Digital tuning register 0

Register file: 0x06 – Digital receiver configuration, sub-register 0x00 is a tuning register. The value here needs to change depending on a number of parameters. Please take care not to write other values to this register as doing so may cause the QM33100 to malfunction.

REG:06:00 – DTUNE0 – Digital tuning register 0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	-	-	-	-	DT0B4	-	-	PAC	
																0	0	0	1	0	0	0	0	0	0	0	1	1	1	0	0

The bits of the DTUNE0 register identified above are individually described below:

Field	Description of fields within Sub-register 0x06:00 – Digital RX tuning register
PAC reg:06:00 bits:1-0	Preamble Acquisition Chunk size.The PAC size should be selected depending on the expected preamble length in the receiver. For details of PAC size and its role please refer to section 4 – *To* enable the receiver, the host issues an RX start command (see section on Fast Commands). The receiver will start by searching for preamble continually until preamble has been detected or acquired, then a demodulation will be attempted. A preamble detection timeout may be set to allow the receiver to stop searching for preamble after a desired period (which is programmable in PRE_TOC). A receive sequence is shown in Figure 16. Preamble Table 12 gives the recommended PAC size configuration for each preamble length. 0x00 – is used for a PAC of 8 symbols 0x01 – is used for a PAC of 16 symbols 0x02 – is used for a PAC of 32 symbols 0x03 – is used for a PAC of 4 symbols

Field	Description of fields within Sub-register 0x06:00 – Digital RX tuning register
DT0B4 reg:06:00 bit:4	Tuning bit 4 of digital tuning reg0. This bit should be cleared to zero for best performance.
- reg:06:00 bits:7-5	Bits marked '-' are reserved and should not be modified.

8.2.7.2 Sub-register 0x06:02 – SFD detection timout count

ID	Length (octets)	Type	Mnemonic	Description
06:02	2	RW	RX_SFD_TOC	SFD detection timeout count

Register file: 0x06 – Digital receiver configuration, sub-register 0x02 is used to set the 16-bit SFD detection timeout counter period, in units of preamble symbols. The SFD detection timeout starts running as soon as preamble is detected. If the SFD sequence is not detected before the timeout period expires then the timeout will act to abort the reception currently in progress, and set RXSTO event status bit. SFD timeout events are also counted in *Sub-register 0x0F:10 – timeout error counter*, assuming that counting is enabled by the EVC_EN bit in *Sub-register 0x0F:00 – Event counter control*.

The purpose of the SFD detection timeout is to recover from the occasional false preamble detection events that occur. By default this value is 65 symbols (64+8-8+1), which is set to match the default preamble, SFD length and PAC size. When it is known that a shorter or longer preamble is being used then the RX_SFD_TOC value can be reduced or increased appropriately. It is recommended to set it according to formula: preamble length + 1 – PAC size + SFD length. (The reduction of the RX_SFD_TOC value by the PAC size is done because one PAC size of the preamble length will be lost as part of the preamble detection).

WARNING: Please do NOT set RX_SFD_TOC to zero (disabling SFD detection timeout), because in the event of false preamble detection, the IC will remain in receive mode until commanded to do otherwise by the external microcontroller, which can lead to significant reduction in battery life.

8.2.7.3 Sub-register 0x06:04 – Preamble detection timeout count

ID	Length (octets)	Type	Mnemonic	Description
06:04	2	RW	PRE_TOC	Preamble detection timeout count

Register file: 0x06 – Digital receiver configuration, sub-register 0x04 is used to set the 16-bit preamble detection timeout period, in units of PAC size symbols. The default/reset value is zero which disables the preamble detection timeout. The preamble detection timeout starts running as soon as the receiver is enabled to hunt for preamble. In the case of delayed receive (as commanded using the CMD_DRX) the preamble detection timeout starts after the delay when the receiver actually turns on to hunt for preamble. If a preamble sequence in not detected before the timeout period expires then the timeout will act to abort

the reception currently in progress, and set the RXPTO event status bit in *Sub-register 0x00:44 – System event status*.

In cases where a response is expected at a particular time, this timeout can be used to flag that the expected response is not starting on time and hence to turn off the receiver earlier than would otherwise be the case, (i.e. if just employing the frame wait timeout). This can give a good power saving, in situations of sending a message and awaiting a response that often does not come.

The PRE_TOC is programmed in units of PAC size, which can be 4, 8, 16 or 32 symbols. Table 9 gives the preamble symbol lengths. The PAC size is set in DTUNE0. As this is a 16-bit counter the maximum preamble detection timeout possible is 65535 × (*PAC size)*, a period of over 250 ms for the smallest PAC size. A value of zero disables the preamble detection timeout.

Example: Supposing our preamble length is 1024 symbols and the PAC size is set to 32 (in line with Table 12) and, we send a message and know that the response (if present) will come after exactly 30 ms (because the responder is using delayed send to begin the response exactly 30 ms after receiving our message). We can command a 30 ms delayed receive (timed from our message transmission time) and have PRE_TOC programmed to a value of 32, which is the preamble length (1024) divided by the PAC size (32).

8.2.7.4 Sub-register 0x06:0C – Digital RX tuning register 3

ID	Length (octets)	Type	Mnemonic	Description
06:0C	4	RW	DTUNE3	Digital receiver tuning register 3

Register file: 0x06 – Digital receiver configuration, sub-register 0x0C is a 32-bit tuning register. The value here needs to change depending packet configuration used (as shown in Figure 13) . The values needed are given in Table 28 below. Please take care not to write other values to this register as doing so may cause the QM33100 to malfunction.

Table 28 : Sub-register 0x06:0C – DTUNE3 values

packet configurations	Value to program to Sub-register 0x06:0C – DRX_TUNE3
0, 1 or 2	0xAF5F584C
3 (STS with no data)	0xAF5F35CC

8.2.7.5 Sub-register 0x06:10 – Digital RX tuning register 4

ID	Length (octets)	Type	Mnemonic	Description
06:10	4	RO	DTUNE_4	Digital Tuning Reserved register

Register file: 0x06 – Digital receiver configuration, sub-register 0x10 is a reserved register.

8.2.7.6 Sub-register 0x06:14 – Digital RX tuning register 5

ID	Length (octets)	Type	Mnemonic	Description
06:14	4	RO	DTUNE_5	Digital Tuning Reserved register

Register file: 0x06 – Digital receiver configuration, sub-register 0x14 is a reserved register.

8.2.7.7 Sub-register 0x06:29 – Carrier recovery integrator register

ID	Length (octets)	Type	Mnemonic	Description
06:29	3	RO	DRX_CAR_INT	Carrier recovery integrator register

Register file: 0x06 – Digital receiver configuration, sub-register 0x29 is a read-only register estimate of the remote transmitter's frequency offset. This is generated during the reception of each packet as the receiver locks on and compensates for the frequency offset of the transmitting device to successfully receive a packet. This is a 21-bit signed number with the lower 17 bits (fractional part), and the upper 4 bits as the integer portion of the number. When a packet is successfully received, this register can be read and converted to the frequency error (in Hz) using the formula:

$$F_{offset} = \frac{C_{int} \times 2^{-17}}{2\left(\frac{N_{samples}}{F_s}\right)}$$

$$where$$

$$C_{int} = carrier\ integrator\ value$$

$$N_{samples} = \begin{cases} 8192\ for\ 110kb/s \\ 1024\ otherwise \end{cases}$$

$$F_s = 998.4 \times 10^6$$

F_{offset} is the absolute frequency error in Hz. It can be converted to a clock offset (in ppm) by scaling by the carrier frequency as follows

$$Offset_{ppm} = -10^6 \times \frac{F_{offset}}{F_c}$$

$$where$$
$$F_{offset} = frequency\ offset\ in\ Hz$$
$$F_c = 6489.6MHz\ for\ channel\ 5$$

The minus sign is produced by the process of measuring the clock offset.

For a particular channel, the formulas reduce to multiplying the content of the carrier integrator register with the appropriate constant from the table below:

Table 29: Constants for frequency offset calculation

Data Rate	Channel 5	Channel 9
850 kb/s, 6.81 Mb/s	-0.5731e-3	-0.1252e-3

NOTE: The carrier recovery algorithm continues to run during the reception of the whole packet so that, in processing a receive interrupt say, the DRX_CAR_INT register value reflects the value at the end of reception. An alternative value is given in CIA_DIAG_0 register.

8.2.8 Register file: 0x07 – Analog RF configuration block

ID	Length (octets)	Type	Mnemonic	Description
0x07	91	-	RF_CONF	Analog RF configuration

Register map register file 0x07 is concerned with the low-level configuration of the IC analog blocks. It contains a number of sub-registers. An overview of these is given by Table 30. Each of these sub-registers is separately described in the sub-sections below.

Table 30: Register file: 0x07 – Analog RF configuration block overview

OFFSET in Register 0x07	Mnemonic	Description
0x00	RF_ENABLE	RF enable
0x04	RF_CTRL_MASK	RF enable mask
0x14	RF_SWITCH	RF switch configuration
0x1A	RF_TX_CTRL_1	RF transmitter configuration
0x1C	RF_TX_CTRL_2	RF transmitter configuration
0x28	TX_TEST	Transmitter test configuration
0x34	SAR_TEST	Transmitter Calibration – SAR temperature sensor read enable

8.2.8.1 Sub-register 0x07:00 – RF control enable

ID	Length (octets)	Type	Mnemonic	Description
07:00	4	RW	RF_ENABLE	RF control enable

Register file: 0x07 – Analog RF configuration block, sub-register 0x00 is a 32-bit configuration register for the transceiver. This register can be used to enable TX RF blocks when exercising certain test modes (e.g. Continuous Wave mode). Table 31: RF_ENABLE and RF_CTRL_MASK values shows the value to program to this register when we want to force the transmitter on even when frame are not being transmitted. Please take care not to write other values to the reserved area of this register as doing so may cause the QM33100 to malfunction. **Note:** the same value should be written into RF_CTRL_MASK register.

Table 31: RF_ENABLE and RF_CTRL_MASK values

TX Channel	32-bit value to program to RF_ENABLE and RF_CTRL_MASK
5	0x02003C00
9	0x02001C00

8.2.8.2 Sub-register 0x07:04 – RF control mask

ID	Length (octets)	Type	Mnemonic	Description
07:04	4	RW	RF_CTRL_MASK	RF control mask

Register file: 0x07 – Analog RF configuration block, sub-register 0x04 is a 32-bit configuration register for the transceiver. This register can be used to enable TX RF blocks when exercising certain test modes (e.g.

Continuous Wave mode). Table 31: RF_ENABLE and RF_CTRL_MASK values shows the value to program to this register when we want to force the transmitter on even when frames are not being transmitted. Please take care not to write other values to the reserved area of this register as doing so may cause the QM33100 to malfunction. **Note:** this register should match the value written into RF_ENABLE register.

8.2.8.3 Sub-register 0x07:14 – RF switch control

ID	Length (octets)	Type	Mnemonic	Description
07:14	4	RW	RF_SWITCH	TX RX switch control

Register file: 0x07 – Analog RF configuration block, sub-register 0x14 is a 32-bit control register for the TXRX switch. The RF_SWITCH register contains the following bitmapped sub-fields:

REG:07:14 – RF_SWITCH – TX RX switch control																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	TRXSWCTRL						-	-	-	-	-	-	-	TRXSWEN	-	ANTSWCTRL			-	-	-	ANTSWEN	-	ANT_TXRX_TXPORT	ANT_TXRX_RXPORT	ANT_TXRX_MODE_	-	-	ANTSWPDOAPORT	ANTSWNOTOGGLE
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Definition of the bit fields within RF_SWITCH:

Field	Description of fields within Sub-register 0x07:14 – RF switch control
ANTSWNOTOGGLE reg:07:14 bit:0	When set to 1, the automatic toggling of the antenna switch is disabled when the device is operating in PDoA modes. Setting this bit to 0 will allow device to automatically toggle the switch, this is the default operation.
ANTSWPDOAPORT reg:07:14 bit:1	Specifies the starting port for reception when the device is operating in PDoA modes. If set to 0, which is the default operation, RF port 1 will be used as the starting port, when set to 1, RF port 2 will be used as the starting port.
ANT_TXRX_MODE_OVR reg:07:14 bit:4	This enables the TXRX override mode to control the RF switch.
ANT_TXRX_RXPORT reg:07:14 bit:5	This configures antenna port for the reception when the TXRX override mode is enabled Antenna port for RX when in TXRX mode override. 0x0 - RF port 2 will be used for reception 0x1 - RF port 1 will be used for reception This only has effect when ANT_TXRX_MODE_OVR

Field	Description of fields within Sub-register 0x07:14 – RF switch control
ANT_TXRX_TXPORT reg:07:14 bit:6	This configures antenna port for the transmission when the TXRX override mode is enabled Antenna port for TX when in TXRX mode override. 0x0 - RF port 1 will be used for transmission 0x1 – RF port 2 will be used for transmission This only has effect when ANT_TXRX_MODE_OVRANT_TXRX_MODE_OVRANT_TXRX_MODE_OVRANT_TXRX_MODE_OVRANT_TXRX_MODE_OVRANT_TXRX_MODE_OVR is enabled.
ANTSWEN reg:07:14 bit:8	Setting this to 1 will enable manual control of the antenna switch (various options are detailed in ANTSWCTRL below),. This means that the switch needs to be toggled by the host software. When it is set to 0, which is the default operation, the switch will be automatically controlled.
ANTSWCTRL reg:07:14 bits:14:12	Manual control of antenna switch when ANTSWEN is set. This is the bitmask with the following bit definitions: 0x0 – disconnect from RF ports 0x1 – configure switch for RF port 1 0x2 – configure switch for RF port 2 All others – Reserved
TRXSWEN reg:07:14 bit:16	Setting this to 1 will enable manual control of the TX RX switch. When set to 0, the switch is automatically controlled by the device.
TRXSWCTRL reg:07:14 bits:29-24	TX/RX switch control when TRXSWEN bit is set. This is bitmask with following bit definitions: 0x01 – configure switch for TX 0x2a – configure switch for RX channel 9 0x1c – configure switch for RX channel 5 All others – Reserved
- reg:07:14 bits:various	These fields are reserved.

8.2.8.4 Sub-register 0x07:1A – RF TX control register 1

ID	Length (octets)	Type	Mnemonic	Description
07:1A	1	RW	RF_TX_CTRL_1	Analog TX control register

Register file: 0x07 – Analog RF configuration block, sub-register 0x1A is an 8-bit control register for the transmitter. The value here needs to be set to 0x0E for the optimal performance.

8.2.8.5 Sub-register 0x07:1C – RF TX control register 2

ID	Length (octets)	Type	Mnemonic	Description
07:1C	4	RW	RF_TX_CTRL_2	Analog TX control register

Register file: 0x07 – Analog RF configuration block, sub-register 0x1C is a 32-bit control register for the transmitter. The value here needs to be set depending on the TX channel selected by the RF_CHAN

configuration in *Sub-register 0x01:08 – Channel control*. The values required are given in Table 32. Please take care not to write other values to this register as doing so may cause the QM33100 to malfunction.

Table 32: RF_TX_CTRL_2 values

TX Channel	32-bit value to program to RF_TX_CTRL_2
5	0x1C071134
9	0x9C010027

REG:07:1C –RF_TX_CTRL_2– Transmitter analog settings

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PULSE_SHAPE	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PG_DELAY					
0	0	0	1	1	1	0	0	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	1	0	0	1	1	0	1	1	0

Definition of the bit fields within RF_TX_CTRL_2:

Field	Description
PG_DELAY reg:07:1C bits:5:0	the Pulse Generator Delay value. This effectively sets the width of the transmitted pulses there by setting the output bandwidth . The value used here depends on the TX channel selected by the RF_CHAN configuration in Sub-register 0x01:08 – Channel control. Recommended values are given in Table 33 below; note however that these values may need to be tuned for spectral regulation compliance depending on external circuitry.
PULSE_SHAPE reg:07:1C bits:31	Selects the TX pulse shape. 0x0 – Default TX pulse shape 0x1 – Alternate pulse shape Note: It is required to set this bit to 1 to comply with Japanese and Korean regulations.
Reserved reg:07:1C bits:various	These fields are reserved. Program only as directed in Table 32.

Table 33: PG_DELAY recommended values

Pulse Shape	6-bit value to program into PG_DELAY
0	0x34
1	0x27

Note: The same PG_DELAY value is recommended for Channel 5 and 9.

8.2.8.6 Sub-register 0x07:28 – TX test

ID	Length (octets)	Type	Mnemonic	Description
07:28	1	RW	TX_TEST	Transmitter test register

Register file: 0x07 – Analog RF configuration block, sub-register 0x28, is used for configuring transmitter test modes. It contains the following bitmapped sub-fields:

REG:07:28 – TX_TEST – Transmitter test register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	-	TX_ENTEST			
																								0	0	0	0	0	0	0	0

Definition of the bit fields within TX_TEST:

Field	Description of fields within Sub-register 0x07:28 – TX test
TX_ENTEST reg:07:28 bits:0-3	Transmitter test enable. This is bitmask with following bit definitions: 0x01 – configure transmitter for continuous wave transmission mode 4 0x02 – configure transmitter for continuous wave transmission mode 3 0x04 – configure transmitter for continuous wave transmission mode 2 0x08 – configure transmitter for continuous wave transmission mode 1 All others – Reserved More information can be found in PG_TEST
- reg:07:28 bits:various	Bits marked '-' in this register are reserved and should always be written as zero to avoid any malfunction of the QM33100.

8.2.8.7 Sub-register 0x07:34 – SAR temperature sensor read enable

ID	Length (octets)	Type	Mnemonic	Description
07:34	1	RW	SAR_TEST	SAR temperature sensor read enable

Register file: 0x07 – Analog RF configuration block, sub-register 0x34, contains the following bitmapped sub-fields:

REG:07:34 – SAR_TEST – SAR TS read enable																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	-	-	SAR_RDEN	-	-
																								0	0	0	0	0	0	0	0

Definition of the bit fields within SAR_TEST:

Field	Description of fields within Sub-register 0x07:34 – SAR temperature sensor read enable
SAR_RDEN reg:07:34 bit:2	Writing 1 enables the SAR temperature sensor reading
- reg:07:34 bits:various	Bits marked '-' in this register are reserved and should always be written as zero to avoid any malfunction of the QM33100.

8.2.8.8 Sub-register 0x07:40 – LDO voltage tune

ID	Length (octets)	Type	Mnemonic	Description
07:40	8	RW	LDO_TUNE	Internal LDO voltage tuning parameter

Register file: 0x07 – Analog RF configuration block, sub-register 0x40 is the LDO voltage tuning register. Please take care not to write to this register unless you are loading the calibrated value from OTP.

REG:07:40 – LDO_TUNE – LDO voltage tuning																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LDO_TUNE (low 40 bits)																													
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0

REG:07:44 – LDO_TUNE – LDO voltage tuning																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	LDO_TUNE (high 28 bits)																											
0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0

Definition of the bit fields within LDO_TUNE:

Field	Description of fields within Sub-register 0x07:40 – LDO voltage tune
LDO_TUNE reg:07:40 bits:60:0	This register is used to control the output voltage levels of the on chip LDOs. For production devices, on power-up this register is loaded with a trim value stored in OTP memory – see LDOTUNE_CAL in Table 17. This value can be automatically copied from OTP into the register using LDO_KICK. Ensure that the LDOTUNE_CAL OTP value is programmed before attempting to copy it over to this address – i.e. if OTP reads 0x0 then default should be used.

Note: When programming OTP memory the LDO_TUNE bits [15-12] need to be set to 0xf.

8.2.8.9 Sub-register 0x07:48 – LDO control

ID	Length (octets)	Type	Mnemonic	Description
07:48	4	RW	LDO_CTRL	LDO control register

Register file: 0x07 – Analog RF configuration block, sub-register 0x48 is a 32-bit configuration register for the transceiver. This register can be used to enable LDO blocks when excercising certain test modes (e.g. Continuous Wave mode), please see QM33100 APIs for details [3]

8.2.8.10 Sub-register 0x07:51 – LDO load

ID	Length (octets)	Type	Mnemonic	Description
07:51	1	RW	LDO_RLOAD	LDO tuning register

Register file: 0x07 – Analog RF configuration block, sub-register 0x51 is a 8-bit tuning register for the transceiver. This register should be set to 0x14 for optimal operation.

8.2.9 Register file: 0x08 – Transmitter calibration block

ID	Length (octets)	Type	Mnemonic	Description
0x08	47	-	TX_CAL	Transmitter calibration block

Register file: 0x08 is the transmit calibration block concerned with ensuring the optimum configuration of the transmit signal. It contains a number of sub-registers. An overview of these is given by Table 34. Each of these sub-registers is separately described in the sub-sections below.

Table 34: Register file: 0x08 – Transmitter calibration block overview

OFFSET in Register 0x08	Mnemonic	Description
0x00	SAR_CTRL	Transmitter Calibration – SAR control
0x04	SAR_STATUS	Transmitter Calibration – SAR status
0x08	SAR_READING	Transmitter Calibration – Latest SAR readings
0x0C	SAR_WAKE_RD	Transmitter Calibration – SAR readings at last wake-up
0x10	PGC_CTRL	Transmitter Calibration – Pulse Generator control
0x14	PGC_STATUS	Transmitter Calibration – Pulse Generator status
0x18	PG_TEST	Transmitter Calibration – Pulse Generator test
0x1C	PG_CAL_TARGET	Transmitter Calibration – Pulse Generator count target value

8.2.9.1 Sub-register 0x08:00 – SAR control

ID	Length (octets)	Type	Mnemonic	Description
08:00	1	RW	SAR_CTRL	Transmitter Calibration – SAR control

Register file: 0x08 – Transmitter calibration block, sub-register 0x00, contains the following bitmapped sub-fields:

REG:08:00 – SAR_CTRL – Transmitter Calibration SAR control																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	-	-	-	-	SAR_START
																								0	0	0	0	0	0	0	0

Definition of the bit fields within SAR_CTRL:

Field	Description of fields within Sub-register 0x08:00 – SAR control
SAR_START reg:08:00 bit:0	Writing 1 sets SAR enable and writing 0 clears the enable. Once the enable is set, the host should poll the SAR status (SAR_STATUS) which will be set to 1 once the SAR sampling has completed. The temperature and voltage values can be read then.
- reg:08:00 bits:15–1	Bits marked '-' in register SAR_CTRL are reserved and should always be written as zero to avoid any malfunction of the QM33100.

8.2.9.2 Sub-register 0x08:04 – SAR status

ID	Length (octets)	Type	Mnemonic	Description
08:04	1	RW	SAR_STATUS	Transmitter Calibration – SAR status

Register file: 0x08 – Transmitter calibration block, sub-register 0x04, contains the following bitmapped sub-fields:

REG:08:04 – TC_SARC – Transmitter Calibration SAR status																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SAR_DONE
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Definition of the bit fields within SAR_STATUS:

Field	Description of fields within Sub-register 0x08:04 – SAR status
SAR_DONE reg:08:04 bit:0	The SAR status bit is zero when SAR is gathering data from the sensors, and will be set to 1 when the data is ready to be read.
- reg:08:04 bits:15–1	Bits marked '-' in register SAR_STATUS are reserved and should always be written as zero to avoid any malfunction of the QM33100.

8.2.9.3 Sub-register 0x08:08 – Latest SAR readings

ID	Length (octets)	Type	Mnemonic	Description
08:08	3	RO	SAR_READING	Transmitter Calibration –Latest SAR readings

Register file: 0x08 – Transmitter calibration block, sub-register 0x08, contains the following bitmapped sub-fields:

REG:08:08 – SAR_READING – Transmitter Calibration latest SAR readings																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								-	-	-	-	-	-	-	-	SAR_LTEMP								SAR_LVBAT							
								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Definition of the bit fields within SAR_READING:

Field	Description of fields within Sub-register 0x08:08 – Latest SAR readings
SAR_LVBAT reg:08:08 bits:7–0	Latest SAR reading for Voltage level. The 8-bit value reported here is the voltage reading, from the last time the SAR A/D was used to sample the battery voltage monitor output. The LSB is approximately 6 mV. The value can be converted to an actual voltage by employing the formula: Voltage (volts) = ((SAR_LVBAT – *OTP_READ(V_{meas} @ 3.0 V)*) * 0.0251) + 3.0 This uses the stored 3.0 V reading in the OTP that was recorded during production test. The effective range of measurement is 2.25 V to 3.76 V. For more details please refer to For more details please refer to section *7.4 – Measuring IC temperature and voltage*.
SAR_LTEMP reg:08:08 bits:15–8	Latest SAR reading for Temperature level. The 8-bit value reported here is the temperature reading from the SAR A/D sampling of the QM33100 internal temperature sensor. The LSB is approximately 0.8 °C. The value can be converted to an actual voltage by employing the formula: Temperature (°C)= ((SAR_LTEMP – *OTP_READ(V_{temp} @ 22°C)*) x 1.05) + 22 This uses the stored 22°C reading in the OTP that was recorded during production test. For additional details please refer to section *7.4 – Measuring IC temperature and voltage*. In order to read temperature sensor, the SAR_TEST needs to be set.
- reg:08:08 bits:23–16	Bits marked '-' in register TC_SARL are reserved

8.2.9.4 Sub-register 0x08:0C – Wake up SAR readings

ID	Length (octets)	Type	Mnemonic	Description
08:0C	2	RO	SAR_WAKE_RD	Transmitter Calibration – SAR readings at last wake-up

Register file: 0x08 – Transmitter calibration block, sub-register 0x06, is a 16-bit status register that contains the following bitmapped sub-fields:

REG:08:0C – TC_SARW– Transmitter Calibration SAR readings at last wake-up																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																SAR_WTEMP								SAR_WVBAT							
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Definition of the bit fields within SAR_WAKE_RD:

Field	Description of fields within Sub-register 0x08:0C – Wake up SAR readings
SAR_WBAT reg:08:0C bits:7–0	SAR reading of Voltage level taken at last wake up event. The 8-bit value reported here is the voltage reading from the SAR A/D sampling of the battery voltage monitor output during wake up. For this to be valid the QM33100 has to have been reset or woken from sleeping with the ONW_RUN_SAR bit enabled in the (saved) *Sub-register 0x0A:00 – AON on* wake configuration.

Field	Description of fields within Sub-register 0x08:0C – Wake up SAR readings
SAR_WTEMP reg:08:0C bits:15– 8	SAR reading of temperature level taken at last wake up event. The 8-bit value reported here is the temperature reading from the SAR A/D sampling of the QM33100 internal temperature sensor during wake up. For this to be valid the QM33100 has to have been reset or woken from sleep with the ONW_RUN_SAR bit enabled in the (saved) *Sub-register 0x0A:00 – AON on* wake configuration. **NOTE: . To read device temperature SAR_READING register needs to be used.**

8.2.9.5 Sub-register 0x08:10 – PG calibration control

ID	Length (octets)	Type	Mnemonic	Description
08:10	2	RW	PGC_CTRL	Transmitter Calibration – Pulse Generator control

Register file: 0x08 – Transmitter calibration block, sub-register 0x10, is a 16-bit control register that controls the pulse generator calibration. There are two modes of operation:

1. PGC_CNT: calculation of PG_COUNT value from a given PG_DELAY
2. PGC_DLY: auto-calibration which updates the PG_DELAY given a reference PG_COUNT value.

When operating in PGC_CNT mode when the calibration is compete it will report new pulse generator count value based on the current PG_DELAY value. When operating in PGC_DLY mode and the calibration is complete, a new pulse generator delay will be generated based on the current PG_TARGET value.

The count value is then stored automatically into the PGC_STATUS register. This delay count gives a consistent reflection of the bandwidth regardless of temperature.

PGC_CTRL register contains the following bitmapped sub-fields:

REG:08:10 – TC_PG_CTRL – Transmitter Calibration – Pulse Generator control settings

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	-	-	-	PGC_TMEAS				PGC_AUTO_CAL	PG_START
																0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Definition of the bit fields within PGC_CTRL:

Field	Description of fields within Sub-register 0x08:10 – PG calibration control
PGC_START reg:08:10 bit:0	Start the pulse generator calibration. Note: This bit is self-clearing when calibration is complete.

Field	Description of fields within Sub-register 0x08:10 – PG calibration control
PGC_AUTO_CAL reg:08:10 bit:1	Start the pulse generator auto-calibration. Note: This bit is self-clearing when calibration is complete. The desired PG_TARGET value should be programmed before running auto-cal. Once auto-cal has compelted the new value of PG_DELAY will be written to TX_CTRL register.
PGC_TMEAS reg:08:10 bit5:2	Number of clock cycles over which to run the pulse generator calibration counter. These are the 4 most significant bits of a 10 bit counter clocked by the system clock.
- reg:08:10	Bits marked '-' in register PGC_CTRL are reserved and should not be changed to avoid any malfunction of the QM33100.

8.2.9.6 Sub-register 0x08:14 – PG calibration status

ID	Length (octets)	Type	Mnemonic	Description
08:14	2	RO	PGC_STATUS	Transmitter Calibration – PG calibration status

Register file: 0x08 – Transmitter calibration block, sub-register 0x09, is a 16-bit status register.

The reference value that is required for temperature bandwidth compensation is the contents of the PGC_STATUS register, PG_DELAY_CNT, which represents a counter incremented with every pulse generated by the QM33100 IC's internal pulse generator. Intuitively, this count value (referred to as PG_COUNT) will vary inversely with the PG_DELAY value – if the delay between pulses increases, the number of pulses within a given timeframe will decrease, and vice versa. PG_DELAY_CNT represents a counter that increments with every pulse generated by the QM33100 IC's internal pulse generator.

The PG_DELAY value will not give the same bandwidth for varying temperatures. The PG_COUNT value, however, will give a stable bandwidth across all temperatures. It is taken as a reference as the QM33100 has a pulse generator auto-calibration procedure; the procedure takes a PG_COUNT value and calculates the PG_DELAY value from this. This PG_DELAY value can then be programmed in to give the desired bandwidth.

That contains the following bitmapped sub-fields:

REG:08:14 – PGC_STATUS – Transmitter Calibration – Pulse Generator status																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	AUTOCAL_DONE	PG_DELAY_CNT											
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Definition of the bit fields within PGC_STATUS:

Field	Description of fields within Sub-register 0x08:14 – PG calibration status
PG_DELAY_CNT reg:08:14 bits:11-0	Pulse generator count value (PG_COUNT) calculated by the calibration routine which gives the PG_DELAY.
AUTOCAL_DONE reg:08:14 bit:12	Auto-calibration of the PG_DELAY has completed. The new value of PG_DELAY has been written to TX_CTRL register.
- reg:08:14	Bits marked '-' in register PGC_STATUS are reserved and should not be changed to avoid any malfunction of the QM33100.

8.2.9.7 Sub-register 0x08:18 – PG test

ID	Length (octets)	Type	Mnemonic	Description
08:18	2	RW	PG_TEST	Transmitter Calibration – Pulse Generator test

Register file: 0x08 – Transmitter calibration block, sub-register 0x18 is an 16-bit configuration register for use in setting the transmitter into continuous wave (CW) mode. This CW mode is employed during the crystal trimming operation which may be done at module manufacturing stage as part of calibrating the crystal oscillator's operating frequency. At all other times, for normal operation, the value in this register should be left in its default power on value of 0x0000. Table 35 shows the values to write to this register in either mode.

Table 35: Sub-register 0x08:18 – PG test values

MODE	16-bit value to program into PG_TEST
Normal operation	0x0000
Continuous Wave (CW) Test Mode	0x000F

For more details of crystal trimming please refer to section *10.1 – IC calibration – crystal oscillator trim*.

8.2.9.8 Sub-register 0x08:1C – PG count target value

ID	Length (octets)	Type	Mnemonic	Description
08:1C	2	RO	PG_CAL_TARGET	Transmitter Calibration – PG count target value

Register file: 0x08 – Transmitter calibration block, sub-register 0x1C, is a 16-bit register that contains PG_TARGET value. Target value of PG_COUNT at which point PG auto cal will complete. The device will return a count value as close as possible to the target. The register contains the following bitmapped sub-fields:

REG:08:1C – PG_CAL_TARGET – Transmitter Calibration PG count target value																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	PG_TARGET											
																0	0	0	0	0	0	0	0	1	0	1	0	0	1	1	0

Definition of the bit fields within PG_CAL_TARGET:

Field	Description of fields within Sub-register 0x08:1C – PG count target value
PG_TARGET reg:08:1C bits:11–0	Pulse generator target value of PG_COUNT at which point PG auto cal will complete. The device will return a count value as close as possible to the target
- reg:08:1C	Bits marked '-' in register PG_CAL_TARGET are reserved and should not be changed to avoid any malfunction of the QM33100.

8.2.10 Register file: 0x09 – Frequency synthesiser control block

ID	Length (octets)	Type	Mnemonic	Description
0x09	27	-	FS_CTRL	Frequency synthesiser control block

Register map register file 0x09 is the frequency synthesiser control block. Its main functionality is the generation of the carrier frequency necessary for the operating channel. It contains a number of sub-registers. An overview of these is given by Table 36. Each of these sub-registers is separately described in the sub-sections below.

Table 36: Register file: 0x09 – Frequency synthesiser control block overview

OFFSET in Register 0x09	Mnemonic	Description
0x00	PLL_CFG	PLL configuration
0x04	PLL_CC	PLL coarse code
0x08	PLL_CAL	PLL calibration configuration
0x14	XTAL	Crystal trim

8.2.10.1 Sub-register 0x09:00 – PLL configuration

ID	Length (octets)	Type	Mnemonic	Description
09:00	2	RW	PLL_CFG	PLL configuration

Register file: 0x09 – Frequency synthesiser control block, sub-register 0x00 is the PLL configuration register. This allows per channel configuration of the PLL.

Table 37: Reference values Sub-register 0x09:00 – PLL configuration

TX Channel	Sub-register 0x09:00 – PLL configuration values
5	0x1F3C
9	0x0F3C

8.2.10.2 Sub-register 0x09:04 – PLL coarse code

ID	Length (octets)	Type	Mnemonic	Description
09:04	1	RW	PLL_CC	PLL coarse code – starting code for calibration procedure

Register file: 0x09 – Frequency synthesiser control block, sub-register 0x04 is the PLL configuration register. It sets the starting code for PLL calibration. If OTP address 0x35 (PLL_LOCK_CODE) has a non-zero value, it should be copied to this register, prior to PLL calibration, the USE_OLD bit should also be set.

REG:09:04 – PLL_CC – PLL starting code

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	CH9_PREBUF	CH5_PREBUF	CH9_ICAS	CH9_RCAS	-	-	CH5_CODE														-	CH9_CODE							
				1	0	1	1			0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1	0	1	0

The bits of the PLL_CC register identified above are individually described below:

Field	Description of fields within Sub-register 0x09:04 – PLL coarse code
- reg:09:04 bits: 31:22	These bits are reserved.
CH9_PREBUF reg:09:04 bit:27	This bit controls enabling of pre-buffer for Channel 9 PLL calibration. This bit is set to 1 by default and should be cleared to 0 during normal operation.
CH5_PREBUF reg:09:04 bit:26	This bit controls enabling of pre-buffer for Channel 5 PLL calibration. This bit is set to 0 by default and should be left to 0 during normal operation
CH9_ICAS reg:09:04 bit:25	0 – replica bias scheme as per D0 1 – current diversion (increases headroom across cascode device in replica bias) is disabled

Field	Description of fields within Sub-register 0x09:04 – PLL coarse code
CH9_RCAS reg:09:04 bit:24	Added for debug purposes 0 (default) - Rcascode in replica bias is 1.33K 1 - Rcascode in replica bias is 1.5K
CH5_CODE reg:09:04 bit:21:8	PLL calibration coarse code for channel 5. The default value is 0xF. Following PLL calibration, this value may be updated with a new code value. This value may change per device.
CH9_CODE reg:09:4 bits:6–0	PLL calibration coarse code for channel 9. The default code is 0xB. Following PLL calibration, this value may be updated with a new code value. This value may change per device and/or over temperature.

8.2.10.3 Sub-register 0x09:08 – PLL calibration

ID	Length (octets)	Type	Mnemonic	Description
09:08	2	RW	PLL_CAL	PLL calibration configuration

Register file: 0x09 – Frequency synthesiser control block, sub-register 0x08 is the PLL calibration register.

REG:09:08 – PLL_CAL – PLL calibration configuration

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	CAL_EN	PLL_CFG_LD						USE_OLD	
							0	0	1	0	0	0	0	0	1

The bits of the PLL_CAL register identified above are individually described below:

Field	Description of fields within Sub-register 0x09:08 – PLL calibration
- reg:09:08 bits:various	These bits are reserved.
CAL_EN reg:09:08 bit:8	PLL calibration enable bit. This bit needs to be set if PLL needs to be recalibrated, e.g. on channel 9 when temperature changes by 20° or if device is configured for one channel (e.g 5) and then needs to be reconfigured to the other e.g. channel 9, but is not reset prior to re-configuration.This bit will self clear.
PLL_CFG_LD reg:09:8 bits:7–2	PLL calibration configuration value. The default is 0x8.
USE_OLD reg:09:8 bit:1	When this is set to 1 the device will use the coarse code value as set in PLL_CC register as starting point for PLL calibration.

Note: When channel 9 configuration is used, the PLL needs to be re-calibrated if temperature changes by 20°C. For more info see section 0.

8.2.10.4 Sub-register 0x09:14 – Crystal trim register

ID	Length (octets)	Type	Mnemonic	Description
09:14	1	RW	XTAL	Frequency synthesiser – Crystal trim

Register file: 0x09 – Frequency synthesiser control block, sub-register 0x14 is the crystal trim register. This allows a fine control over the crystal oscillator to tune the QM33100 operating frequencies quite precisely. For details of the use of this register please refer to section 10.1 *– IC calibration – crystal oscillator trim*.

REG:09:14 – XTAL – Crystal trim setting

7	6	5	4	3	2	1	0
-	XTAL_TRIM						
0	0	0	0	0	0	0	0

The bits of the XTAL register identified above are individually described below:

Field	Description of fields within Sub-register 0x09:14 – Crystal trim register
- reg:09:14 bits:7, 6	These bits are reserved.
XTAL_TRIM reg:09:14 bits:6–0	Crystal Trim. Crystals may be trimmed using this register setting to tune out errors, see *10.1 – IC calibration – crystal oscillator trim*.

8.2.11 Register file: 0x0A – Always-on system control interface

ID	Length (octets)	Type	Mnemonic	Description
0x0A	23	-	AON	Always on system control interface block

Register map register file 0x0A is the Always-On system control block, (AON).

The AON block contains a low-power configuration array that remains powered-up as long as power (from the battery, for example) is supplied to the QM33100 via the VDD1 pin. User configurations, from SPI accessible host interface registers, can be automatically saved in the AON memory when the QM33100 enters **SLEEP** or **DEEPSLEEP** states and automatically restored from the AON memory when the QM33100 wakes from sleeping. Additional discussion of these modes may be found in section *2.5.1 – SLEEP and DEEPSLEEP*.

This Register file: 0x0A – Always-on system control interface, controls the functions that remain on when the IC enters its low-power **SLEEP** or **DEEPSLEEP** states, and configures the activities the QM33100 should take as the IC wakes from these sleep states. It contains a number of Sub-registers. An overview of these is given by Table 38. Each of these Sub-registers is separately described in the sub-sections below.

Table 38: Register file: 0x0A – Always-on system control overview

OFFSET in Register 0x0A	Mnemonic	Description
0x00	AON_DIG_CFG	AON wake up configuration register
0x04	AON_CTRL	AON control register
0x08	AON_RDATA	AON direct access read data result
0x0C	AON_ADDR	AON direct access address
0x10	AON_WDATA	AON direct access write data
0x14	AON_CFG	AON configuration register

8.2.11.1 Sub-register 0x0A:00 – AON on wake configuration

ID	Length (octets)	Type	Mnemonic	Description
0A:00	3	RW	AON_DIG_CFG	AON wake up configuration register

Register file: 0x0A – Always-on system control interface, sub-register 0x00 is a 24-bit configuration register that is used to control what the QM33100 IC does as it wakes up from low-power **SLEEP** or **DEEPSLEEP** states. The AON_DIG_CFG register contains the following bitmapped sub-fields:

REG:0A:00 – AON_DIG_CFG – AON wake-up configuration register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								-	-	-	-	-	-	-	-				-	ONW_PGFCAL	-	ONW_GO2RX	ONW_GO2IDLE	-		-	-	-	-	ONW_RUN_SAR	ONW_AON_DLD
								0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Definition of the bit fields within AON_DIG_CFG:

Field	Description of fields within Sub-register 0x0A:00 – AON on wake configuration
- reg:0A:00 bits:[various]	Bits marked '-' in register 0x0A:00 are reserved and should be left unchanged to avoid any malfunction of the QM33100.
ONW_AON_DLD reg:0A:00 bit:0	On Wake-up download the AON array. On Wake-download configurations from the AON memory into the host interface register set. When the ONW_AON_DLD bit is set to 1 the values of user configuration registers are restored to their pre-sleep configuration values. When the ONW_AON_DLD bit is 0 the values of user configuration registers revert to their power-on-reset value when the QM33100 wakes-up from **SLEEP** or **DEEPSLEEP**.

Field	Description of fields within Sub-register 0x0A:00 – AON on wake configuration
ONW_RUN_SAR reg:0A:00 bit:1	On Wake-up Run the (temperature and voltage) Analog-to-Digital Convertors. The QM33100 is equipped with 8-bit A/D convertors to sample the IC temperature and its input battery voltage. Setting this bit will cause the automatic initiation of temperature and input battery voltage measurements when the QM33100 wakes from **SLEEP** or **DEEPSLEEP** states. As a result the temperature is measured before the IC heats up (and so may be a good measure of the ambient temperature around the IC), and, the battery voltage is measured before any significant current drain occurs (which may be useful in checking battery health). The resultant temperature and voltage values are available in *Sub-register 0x08:0C – Wake up SAR* readings. For more details of this functionality, please refer to section *7.4 – Measuring IC temperature and voltage*. NOTE: For QM33100 only voltage can be read after wake up. The temperature needs to be read via SAR_READING register.
ONW_GO2IDLE reg:0A:00 bit:8	On Wake-up go to **IDLE_PLL** state. Following completion of wake up, the device will automatically proceed to **IDLE_PLL** state from **IDLE_RC**.
ONW_GO2RX reg:0A:00 bit:9	On Wake-up go to **RX**. Following the completion of wake up, the device will automatically proceed from **IDLE_RC** to **IDLE_PLL** state and then to **RX**.
ONW_PGFCAL reg:0A:00 bit:11	On Wake-up **do not** perform RX calibration, for more details see RX_CAL register. If this is set, then the host will need to manually perform RX calibration prior to enabling the receiver. It is recommended to have this clear all the time.

8.2.11.2 Sub-register 0x0A:04 – AON control register

ID	Length (octets)	Type	Mnemonic	Description
0A:04	1	RW	AON_CTRL	AON control register

Register file: 0x0A – Always-on system control interface, sub-register 0x04 is an 8-bit control register. The bits in this register in general cause direct activity within the AON block with respect to the stored AON memory. The bits then act like commands that are processed by the QM33100 and the bits are automatically cleared as the activity is taken.

The AON_CTRL register contains the following control bits:

REG:0A:04 – AON_CTRL – AON Control Register																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								DCA_ENAB	-	DCA_WRITE_HI	DCA_WRITE	DCA_READ	CFG_UPLOAD	SAVE	RESTORE
																								0	0	0	0	0	0	0	0

Definition of the bit fields within AON_CTRL:

Field	Description of fields within Sub-register 0x0A:04 – AON control register
RESTORE reg:0A:04 bit:0	When this bit is set the QM33100 will copy the user configurations from the AON memory to the host interface register set. The RESTORE bit will auto clear when this command is executed.
SAVE reg:0A:04 bit:1	When this bit is set the QM33100 will copy the user configurations from the host interface register set into the AON memory. It will then proceed to upload the AON block configurations. The SAVE bit will auto clear when this command is executed.
CFG_UPLOAD reg:0A:04 bit:2	Upload the AON block configurations to the AON. This control will copy the AON configurations of *Sub-register 0x0A:14 – AON* configuration register into the AON configuration registers. This may be done for instance to enter **SLEEP** state after correctly configuring it in those two registers, although **SLEEP** state may also be automatically entered under certain conditions by appropriate configurations within *Sub-register 0x11:08 – Sequencing control*. If the CFG_UPLOAD is being set for a purpose other than going to sleep then it needs to be explicitly cleared immediately after use as it is not self-clearing. Going to sleep will automatically clear this bit.
DCA_READ reg:0A:04 bit:3	Direct AON memory access read. When this bit is set, (and direct access is enabled via the DCA_ENAB bit below), it commands a direct read of the low-power configuration array store memory. The address to read from is specified in *Sub-register 0x0A:0C – AON memory address* and the resultant read data is presented in *Sub-register 0x0A:08 – AON read data*. This access is needed to retrieve the result of a calibration measurement on the low-power oscillator, See QM33100 APIs for details [3].
DCA_WRITE reg:0A:04 bit:4	Direct AON memory write access. When this bit is set, (and direct access is enabled via the DCA_ENAB bit below), it commands a direct write of the low-power configuration array store memory. The address to write to is specified in *Sub-register 0x0A:0C – AON memory* address and the data to write is presented in *Sub-register 0x0A:10 – AON data* to write. This access is needed to retrieve the result of a calibration measurement on the low-power oscillator, See QM33100 APIs for details [3].
DCA_WRITE_HI reg:0A:04 bit:5	Direct AON memory write access. This bit together with DCA_WRITE bit needs to be set when using address > 0xFF, (and direct access is enabled via the DCA_ENAB bit below), it commands a direct write of the low-power configuration array store memory. The address to write to is specified in *Sub-register 0x0A:0C – AON memory* address and the data to write is presented in *Sub-register 0x0A:10 – AON data* to write. This access is needed to retrieve the result of a calibration measurement on the low-power oscillator, See QM33100 APIs for details [3].
– reg:0A:04 bit:6	Bits marked '-' in this register are reserved and should always be written as zero to avoid any malfunction of the QM33100.
DCA_ENAB reg:0A:04 bit:7	Direct AON memory access enable bit. This bit needs to be set to 1 to enable the DCA_READ above to operate. **Note:** DCA_ENAB must to be reset to 0 to allow the automatic saving/restoring of user configurations to/from the AON memory, as needed for correct operation during entry and exit from **SLEEP** and **DEEPSLEEP** states.

8.2.11.3 Sub-register 0x0A:08 – AON read data

ID	Length (octets)	Type	Mnemonic	Description
0A:08	1	RW	AON_RDATA	AON direct access read data result

Register file: 0x0A – Always-on system control interface, sub-register 0x08 is an 8-bit register used to return the result of a direct access read of a location in the AON memory array. The location to read from is specified by *Sub-register 0x0A:0C – AON memory address* and the read is initiated using the DCA_READ control bit in *Sub-register 0x0A:04 – AON control* register.

8.2.11.3.1 Reading from a specified address within AON memory

Figure 28 shows the procedural flow for reading from specified address in AON memory:

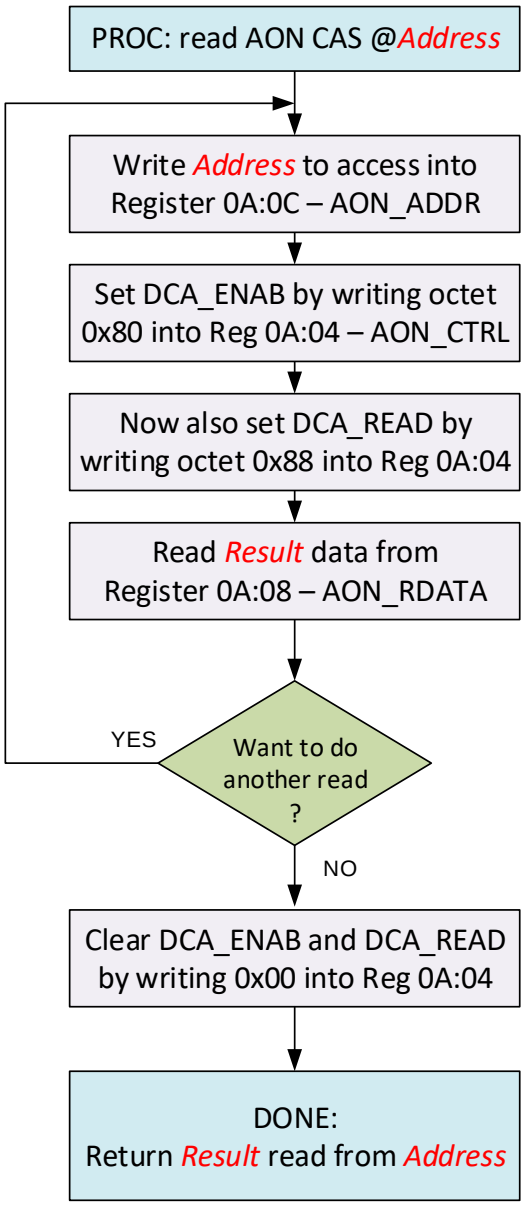

Figure 28: Flow chart for direct read of AON address

8.2.11.4 Sub-register 0x0A:0C – AON memory address

ID	Length (octets)	Type	Mnemonic	Description
0A:0C	2	RW	AON_ADDR	AON direct access address

Register file: 0x0A – Always-on system control interface, sub-register 0x0C is an 16-bit register used to specify the address (9-bits) for a direct access read of the AON memory array. The read is initiated using the DCA_READ control bit in *Sub-register 0x0A:04 – AON control* register and the read result is returned in *Sub-register 0x0A:08 – AON read* data.

8.2.11.5 Sub-register 0x0A:10 – AON data to write

ID	Length (octets)	Type	Mnemonic	Description
0A:10	1	RW	AON_WDATA	AON direct access write data

Register file: 0x0A – Always-on system control interface, sub-register 0x10 is is an 8-bit register used to provide the data for the direct write access of the AON memory array, to the address specified by *Sub-register 0x0A:0C – AON memory* address. The write is initiated using the DCA_WRITE control bit in *Sub-register 0x0A:04 – AON control* register.

8.2.11.5.1 Writing to a specified address within AON memory

Figure 29 shows the procedural flow for writing to specified address in AON memory:

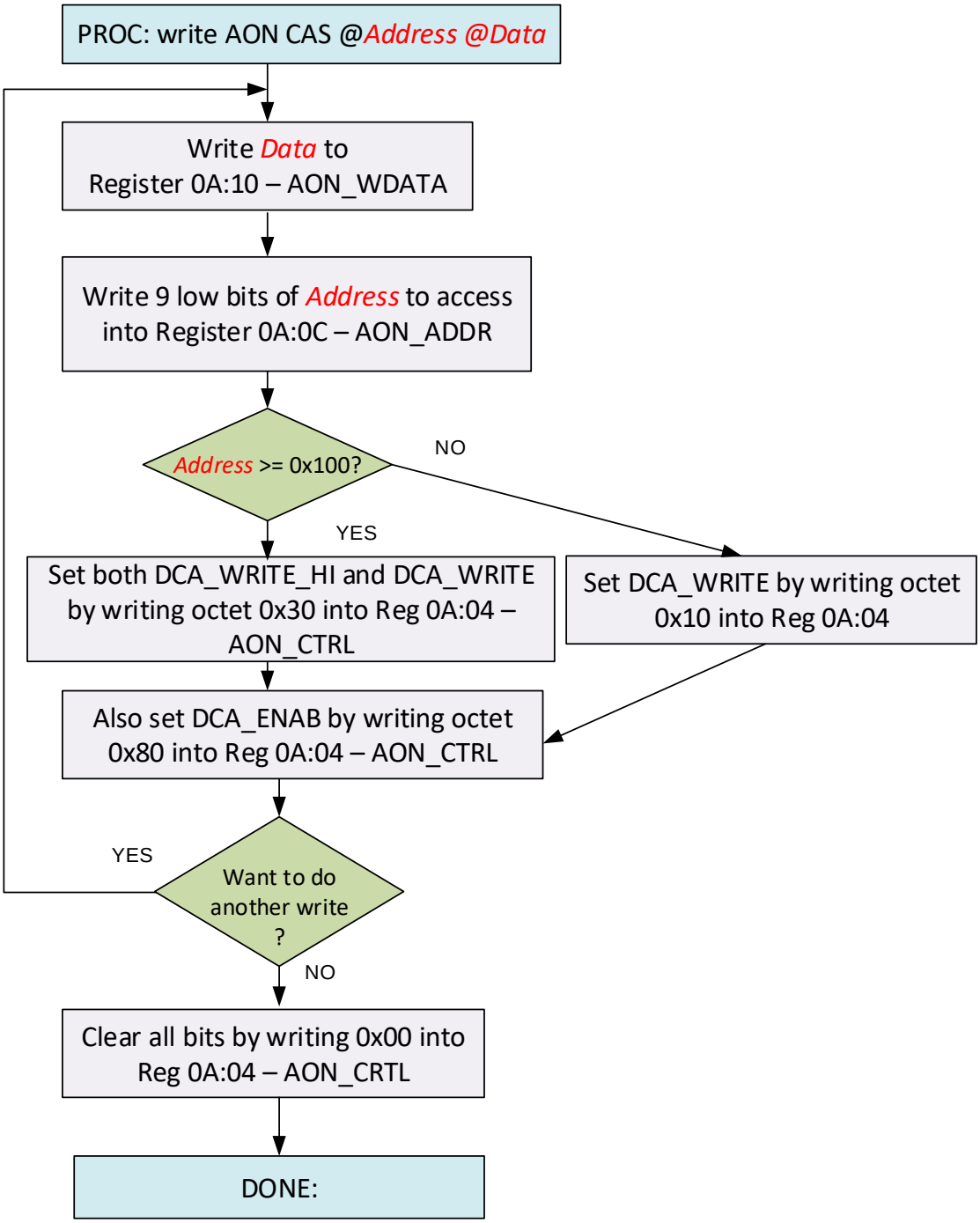

Figure 29: Flow chart for direct write of AON address

8.2.11.6 Sub-register 0x0A:14 – AON configuration

ID	Length (octets)	Type	Mnemonic	Description
0A:14	1	RW	AON_CFG	AON configuration register

Register file: 0x0A – Always-on system control interface, sub-register 0x14 is an 8-bit configuration register for the always on block. The fields of this register are interpreted inside the AON block, which can only happen after these are loaded into the AON block via the CFG_UPLOAD command in *Sub-register 0x0A:04 – AON control* register. The AON_CFG register contains the following fields:

REG:0A:14 – AON_CFG – AON configuration register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								WAKE_POL	FAST_WAKE_EN	PRES_SLEEP	WAKE_WUP	WAKE_CSN	BROUT_EN	WAKE_CNT	SLEEP_EN
																								0	0	0	0	1	1	0	0

The fields of the AON_CFG register identified above are individually described below:

Field	Description of fields within Sub-register 0x0A:14 – AON configuration
SLEEP_EN reg:0A:14 bit:0	This is the sleep enable configuration bit. In order to put the QM33100 into the **SLEEP** or **DEEPSLEEP** state this bit needs to be set and then the configuration needs to be uploaded to the AON using the CFG_UPLOAD bit in *Sub-register 0x0A:04 – AON control* register. The **SLEEP** or **DEEPSLEEP** state can also be entered via the ATX2SLP or ARX2SLP controls in *Sub-register 0x11:08 – Sequencing control*, which will automatically do the configuration upload to the AON and set this SLEEP_EN control. If WAKE_CNT bit below is disabled then the device will enter **DEEPSLEEP** state otherwise it will enter **SLEEP** state.
WAKE_CNT reg:0A:14 bit:1	Wake when sleep counter elapses. This configuration bit enables the sleep counter to bring the QM33100 out of **SLEEP** into operational mode. By default the WAKE_CNT configuration is 0 disabling the sleep counter as a wake-up signal. Setting the WAKE_CNT configuration bit to 1 will mean that the sleep counter will wake the QM33100 form **SLEEP**. See Note below on wake up events. To configure the duration of the sleep counter see 8.2.11.6.1 below.
BROUT_EN reg:0A:14 bit:2	Set to 1 to enable the BROWNOUT detector during **SLEEP** or **DEEPSLEEP**. This block will consume ~6uA. See also VWARN event in the Sub-register 0x00:44 – System event status register.
WAKE_CSN reg:0A:14 bit:3	Wake using SPI access. This configuration bit enables SPICSn to bring the QM33100 out of **SLEEP** or **DEEPSLEEP** into operational mode. By default the WAKE_CSN configuration is 1 enabling the SPICSn input as a wake-up signal. Setting the WAKE_CSN configuration bit to 0 will mean that SPICSn cannot wake the QM33100 form **SLEEP** or **DEEPSLEEP**. See Note below on wake up events.

Field	Description of fields within Sub-register 0x0A:14 – AON configuration
WAKE_WUP reg:0A:14 bit:4	Wake using **WAKEUP** pin. This configuration bit enables the **WAKEUP** line to bring the QM33100 out of **SLEEP** or **DEEPSLEEP** states into operational mode. By default the WAKE_WUP configuration is 1 enabling the **WAKEUP** line as a wake-up signal. Setting the WAKE_WUP configuration bit to 0 will mean that the **WAKEUP** line cannot wake the QM33100 from **SLEEP** or **DEEPSLEEP**. See Note below on wake up events.
PRES_SLEEP reg:0A:14 bit:5	Preserve Sleep. This bit determines what the QM33100 does after a wake-up event, with respect to the ARX2SLP and ATX2SLP sleep controls in *Sub-register 0x11:08 – Sequencing control*, and the SLEEP_EN control in Sub-register 0x0A:14 – AON configuration. When the PRES_SLEEP bit is set to 1 these sleep controls are not cleared upon wake up, allowing the device to be more easily (or automatically) returned to sleep.
FAST_WAKE_EN reg:0A:14 bit:6	This is used to shorten the required pulse duration needed on the WAKEUP SPICSn pin. This will reduce the required wakeup pulse duration from ~3 LP Oscillator cycles down to 10's of ns.
WAKE_POL reg:0A:14 bit:7	This bit allows to control the polarity of **WAKEUP** pin of the QM33100. By default this is set to 0, WAKEUP pin is active high. When WAKE_POL is 0, the WAKEUP pin is active low.

Note:
There are three mechanisms to wake the QM33100:

- using the **WAKEUP** line when the WAKE_WUP configuration is set to 1
- using **SPICSn** when the WAKE_CSN configuration is set to 1,
- using the sleep timer when the WAKE_CNT configuration is 1 and the sleep counter is enabled via the WAKE_CNT bit in AON_CFG.
- Full chip reset

If none of these wake up mechanisms are enabled and the QM33100 is put into **DEEPSLEEP** state then to take QM33100 out of sleep a reset needs to be performed with RSTn pin, or by removing power to the device.

8.2.11.6.1 Configuration of sleep counter

The sleep counter time (SLEEP_TIM) is 16-bits wide, but represents the upper 16-bits of a 28-bit counter, i.e. the low order bit is equal to 4096 counts. For example, if the low power oscillator frequency is 20000 Hz then programming the **SLEEP_TIM** with a value of 24 would yield a sleep time of 24 × 4096 ÷ 20000, which is approximately 4.92 seconds.

The **SLEEP_TIM** is located in AON memory space at the address 0x102 (low 8-bits) and 0x103 (high 8-bits). To write data to this space see 8.2.11.5.1 above.

Note: When programming sleep counter time, the host must delay enabling the sleep counter by at least 32 µs, following the programming of the timer duration for the newly programmed value to apply. Otherwise the counter will use the previously programmed value.

8.2.12 Register file: 0x0B – OTP memory interface

ID	Length (octets)	Type	Mnemonic	Description
0x0B	23	-	OTP_IF	One Time Programmable memory interface

Register map register file 0x0B is the OTP memory interface. This allows read access to parameters stored in the OTP memory, and it is also the interface via which parameters are programmed into the OTP memory. The OTP memory interface contains a number of sub-registers. An overview of these sub-registers is given by Table 39, and each is then separately described in the sub-sections below.

Note: Programming OTP memory is a one-time only activity, any values programmed in error cannot be corrected. Also, please take care when programming OTP memory to only write to the designated areas – programming elsewhere may permanently damage the QM33100's ability to function normally. The OTP memory is programmed 32-bits at a time, e.g. programming of a single 8-bit, 16-bit or 24-bit word is not supported.

For more details of the OTP memory please refer to section *7.3 – Using the on-chip memory*.

Table 39: Register file: 0x0B – OTP memory interface overview

OFFSET in Register 0x0B	Mnemonic	Description
0x00	OTP_WDATA	OTP write data
0x04	OTP_ADDR	OTP address
0x08	OTP_CFG	OTP configuration
0x0C	OTP_STAT	OTP status
0x10	OTP_RDATA	OTP read data
0x14	OTP_SRDATA	OTP Special Register (SR) read data

8.2.12.1 Sub-register 0x0B:00 – OTP data to program

ID	Length (octets)	Type	Mnemonic	Description
0B:00	4	RW	OTP_WDATA	OTP data to program to a particular address

Register file: 0x0B – OTP memory interface, sub-register 0x00 is a 32-bit register. This register is used to configure the OTP memory block for memory writing operations and also to store the data value to be programmed into an OTP location. Writing to OTP memory is an involved procedure. For details of this please refer to section *7.3.2 – Programming a value into memory.*

8.2.12.2 Sub-register 0x0B:04 – OTP programming address

ID	Length (octets)	Type	Mnemonic	Description
0B:04	2	RW	OTP_ADDR	OTP address to which to program the data

Register file: 0x0B – OTP memory interface, sub-register 0x04 is a 16-bit register used to select the address within the OTP memory block that is being accessed (for read or write) this OTP memory interface. The OTP_ADDR register contains the following fields:

REG:2D:04 – OTP_ADDR – OTP address																												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	OTP_ADDR										
																-	-	-	-	-	0	0	0	0	0	0	0	0	0	0	0

The fields of the OTP_ADDR register are described below:

Field	Description of fields within Sub-register 0x0B:04 – OTP programming address
OTP_ADDR reg:0B:04 bits:10 –0	This 11-bit field specifies the address within OTP memory that will be accessed read or written. For details of the OTP memory map and the procedures to read and write OTP memory, please refer to section *7.3 – Using the on-chip memory*.
- reg:0B:04 bits:15 –11	Reserved. The remainder of this register is reserved.

8.2.12.3 Sub-register 0x0B:08 – OTP configuration

ID	Length (octets)	Type	Mnemonic	Description
0B:08	2	RW	OTP_CFG	OTP configuration register

Register file: 0x0B – OTP memory interface, sub-register 0x08 is a 16-bit register used to select and load special receiver operational parameter sets. See *8.2.12.7 – Receiver operating parameter sets* for details. The OTP_CFG register contains the following fields:

REG:0B:08 – OTP_SF – OTP configuration

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	OPS_SEL		OPS_KICK	BIAS_KICK	LDO_KICK	DGC_KICK	DGC_SEL	-	-	-	OTP_WRITE_MR	OTP_WRITE	OTP_READ	OTP_MAN
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the OTP_CFG register are described below:

Field	Description of fields within Sub-register 0x0B:08 – OTP configuration
OTP_MAN reg:0B:08 bit:0	Enable manual control over OTP interface. See QM33100 APIs for details [3]. See section *7.3 – Using the on-chip memory*.
OTP_READ reg:0B:08 bit:1	OTP read enable. See QM33100 APIs for details [3]. See section *7.3 – Using the on-chip memory*.
OTP_WRITE reg:0B:08 bit:2	OTP write enable. See QM33100 APIs for details [3]. See section *7.3 – Using the on-chip memory*.
OTP_WRITE_MR reg:0B:08 bit:3	OTP write mode. See QM33100 APIs for details [3]. See section *7.3 – Using the on-chip memory*.
DGC_SEL reg:0B:08 bit:7	Select the channel 5 (0) or channel 9 (1) DGC table to load. Main use is for on wake loading of the DGC table. Note that CH9 can also be selected by setting the RF_CHAN = 1.
DGC_KICK reg:0B:08 bit:8	This bit when set initiates the loading of the RX_TUNE_CAL parameter from OTP address 0x20 – 0x34 into the register RX_TUNE. Either channel 5 or channel 9 RX tune calibration data is loaded, depending on the configuration of bit.
LDO_KICK reg:0B:08 bit:9	This bit when set initiates the loading of the LDOTUNE_CAL parameter from OTP address 0x4 into the register Sub-register 0x07:40 – LDO voltage tune. See the section Waking from sleep for more details.
BIAS_KICK reg:0B:08 bit:10	This bit when set initiates the loading of the BIASTUNE_CAL parameter from OTP address 0xA into the BIAS_CTRL register. See the section Waking from sleep for more details.
OPS_KICK reg:0B:08 bit:11	This bit when set initiates a load of the operating parameter set selected by the OPS_SEL configuration below.

Field	Description of fields within Sub-register 0x0B:08 – OTP configuration
OPS_SEL reg:0B:08 bits:13,12	Operating parameter set selection. This selects the operating parameter set to load when the OPS_KICK is invoked. Allowed values are: 00: Select parameter set 0 – *Long*. 01: Select parameter set 1 – *SCP*. 10: Select Default parameter set (2 – *Short*). 11: Reserved. Do not select this value. See section *8.2.12.7 – Receiver operating parameter sets* below for details of use of these operating parameter sets.
- reg:0B:08 bits:various	Reserved. Bits marked '-' are reserved. **N.B.: Any change in these bits may cause the QM33100 to malfunction.**

8.2.12.4 Sub-register 0x0B:0C – OTP programming status

ID	Length (octets)	Type	Mnemonic	Description
0B:0C	2	RW	OTP_STAT	OTP memory programming status register

Register file: 0x0B – OTP memory interface, sub-register 0x0C is a 16-bit register used to give status information about the progress of the OTP programming activity. The OTP_STAT register contains the following fields:

REG:0B:0C – OTP_STAT – OTP status

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	-	-	-	-	-	-	-	-	-	-	OTP_VPP_OK	OTP_PROG_DONE
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the OTP_STAT register are described below:

Field	Description of fields within Sub-register 0x0B:0C – OTP programming status
OTP_PROG_DONE eg:0B:0C bit:0	OTP Programming Done. This status bit indicates that the programming of the 32-bit word from OTP_WDATA to the address specified by OTP_ADDR has completed. Writing to OTP memory is an involved procedure. For details of this please refer to section *7.3.2 – Programming a value into memory.*
OTP_VPP_OK reg:0B:0C bit:1	OTP Programming Voltage OK. This status bit indicates that the VPP level is sufficient for programming the OTP memory. For details of OTP programming please refer to section *7.3 – Using the on-chip memory.*

Field	Description of fields within Sub-register 0x0B:0C – OTP programming status
- reg:0B:0C bits:various	Reserved. Bits marked '-' are reserved.

8.2.12.5 Sub-register 0x0B:10 – OTP data read from given address

ID	Length (octets)	Type	Mnemonic	Description
0B:10	4	R	OTP_RDATA	OTP data read from given address

Register file: 0x0B – OTP memory interface, sub-register 0x10 is a 32-bit register. The data value read from an OTP location will appear here after invoking the OTP read function. For details of the OTP memory map and the procedures to read OTP memory, please refer to section *7.3 – Using the on-chip memory*.

8.2.12.6 Sub-register 0x0B:14 – OTP special register

ID	Length (octets)	Type	Mnemonic	Description
0B:14	4	RW	OTP_SRDATA	OTP Special Register (SR) read data

Register file: 0x0B – OTP memory interface, sub-register 0x14 is a 32-bit register. The data value stored in the OTP SR (0x60) location will appear here after power up. For details of the OTP memory map and the procedures to read OTP memory, please refer to section *7.3 – Using the on-chip memory*.

8.2.12.7 Receiver operating parameter sets

The QM33100 receiver has the capability of operating with specific parameter sets that relate to how it acquires the preamble signal and decodes the data. Three distinct operating parameter sets are defined within the IC for selection by the host system designer depending on system characteristics. Table 40 below lists and defines these operating parameter sets indicating their recommended usages.

Table 40: Receiver operating parameter sets

Set	Description
10 – Default / Short	This is the default operating parameter set. This parameter set is designed to give good performance for very short preambles, i.e. the length 64 preamble.. It is however not optimum for long preambles.
00 – Long	This operating parameter set is designed to give good performance for long (>= 256) preambles (size including the SFD and STS length).
01 – reserved	reserved

For most applications the default operating parameter set is the best choice.

8.2.13 Register files: 0x0C, 0x0D, 0x0E – CIA Interface

ID	Length (octets)	Type	Mnemonic	Description
0x0C – 0x0E	269	-	CIA_IF	Channel Impulse Response Analyser (CIA) Interface

Register map register file 0x0C is the CIA control and status interface. The Channel Impulse response Analyser (CIA) function is responsible for analysing the accumulator data, (available through Register file: 0x15 – Accumulator CIR memory). Depending on the exact configuration, there are up to 3 channel impulse response (CIR) estimates in the accumulator memory. One based on the preamble code and two based on the STS sequence.

The CIA identifies the first path in any given CIR estimate. It can then use this to calculate the RX timestamp written to *Sub-register 0x00:60 – Receive time stamp*. The CIA interface contains a number of sub-registers. An overview of these sub-registers in this *Register files: 0x0C, 0x0D, 0x0E – CIA Interface* is given by Table 41 and each is then separately described in the sub-sections below (0x0C and 0x0D are CIA outputs, while 0x0E is the CIA configuration control). The control and configuration of the STS is achieved via the sub-registers in CP_CFG and SYS_CFG. Please also refer to *§ 6 – Secure ranging / timestamping* for details of the operation of the STS for secure ranging operations.

Table 41: Register files: 0x0C, 0x0D, 0x0E – CIA Interface overview

Register file	OFFSET in Register	Mnemonic	Description
0x0C	0x00	IP_TS	Preamble sequence receive time stamp and status
0x0C	0x08	STS_TS	STS receive time stamp and status
0x0C	0x10	STS1_TS	2nd STS receive time stamp and status
0x0C	0x18	TDOA	The TDoA between the two CIRs
0x0C	0x1E	PDOA	The PDoA between the two CIRs
0x0C	0x20	CIA_DIAG_0	CIA Diagnostic 0
0x0C	0x28	IP_DIAG_0	Preamble Diagnostic 0 – peak
0x0C	0x2C	IP_DIAG_1	Preamble Diagnostic 1 – power indication
0x0C	0x30	IP_DIAG_2	Preamble Diagnostic 2 – magnitude @ FP + 1
0x0C	0x34	IP_DIAG_3	Preamble Diagnostic 3 – magnitude @ FP + 2
0x0C	0x38	IP_DIAG_4	Preamble Diagnostic 4 – magnitude @ FP + 3
0x0C	0x48	IP_DIAG_8	Preamble Diagnostic 8 – first path
0x0C	0x58	IP_DIAG_12	Preamble Diagnostic 12 – symbols accumulated
0x0C	0x5C	STS_DIAG_0	STS Diagnostic 0 – STS CIR peak
0x0C	0x60	STS_DIAG_1	STS 0 Diagnostic 1 – STS CIR power indication
0x0C	0x64	STS_DIAG_2	STS 0 Diagnostic 2 – STS CIR magnitude @ FP + 1
0x0C	0x68	STS_DIAG_3	STS 0 Diagnostic 3 – STS CIR magnitude @ FP + 2
0x0D	0x00	STS_DIAG_4	STS 0 Diagnostic 4 – STS CIR magnitude @ FP + 3
0x0D	0x10	STS_DIAG_8	STS 0 Diagnostic 8 – STS CIR first path

Register file	OFFSET in Register	Mnemonic	Description
0x0D	0x20	STS_DIAG_12	STS 0 Diagnostic 12 – STS CIR accumulated STS length
0x0D	0x38	STS1_DIAG_0	STS 1 Diagnostic 0 – STS CIR peak
0x0D	0x3C	STS1_DIAG_1	STS 1 Diagnostic 1 – STS CIR power indication
0x0D	0x40	STS1_DIAG_2	STS 1 Diagnostic 2 – STS CIR magnitude @ FP + 1
0x0D	0x44	STS1_DIAG_3	STS 1 Diagnostic 3 – STS CIR magnitude @ FP + 2
0x0D	0x48	STS1_DIAG_4	STS 1 Diagnostic 4 – STS CIR magnitude @ FP + 3
0x0D	0x58	STS1_DIAG_8	STS 1 Diagnostic 8 – STS CIR first path
0x0D	0x68	STS1_DIAG_12	STS 1 Diagnostic 11 – STS CIR accumulated STS length
0x0E	0x00	CIA_CONF	CIA general configuration
0x0E	0x04	FP_CONF	First path temp adjustment and thresholds
0x0E	0x0C	IP_CONF	Preamble Config 0 – CIA preamble configuration
0x0E	0x14	STS_CONF_0	STS Config 0 – CIA STS configuration
0x0E	0x18	STS_CONF_1	STS Config 1 – CIA STS configuration
0x0E	0x1C	CIA_ADJUST	User adjustment to the PDoA

8.2.13.1 Sub-register 0x0C:00 – Preamble receive time stamp and status

ID	Length (octets)	Type	Mnemonic	Description
0C:00	8	RO	IP_TS	Preamble receive time stamp and status

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x00 of register file 0x0C is an 8-octet register reporting the 40-bit preamble Time of Arrival estimate along with status diagnostic octet relating to it.

REG:0C:00 – IP_TS – Preamble receive time stamp and status (Octets 0 to 3, 32-bits)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IP_TOA (low 32 bits of 40-bit value)																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REG:0C:04 – IP_TS – Preamble receive time stamp and status (Octets 4 to 7, 32-bits)																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IP_TOAST								-	-	IP_POA														IP_TOA (high 8 bits of 40)							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the IP_TS register are described below:

Field	Description of fields within Sub-register 0x0C:00 – Preamble receive time stamp and status
IP_TOA reg:0C:00 bits:39-0	Preamble sequence Time of Arrival estimate. This 40-bit (5-octet) field reports the. The fully adjusted time of reception as estimated by the CIA algorithm operating on the CIR accumulated using the preamble sequence. Please refer to section *4.1.7 – RX message timestamp* for more details of the adjustments applied. The units of the low order bit are approximately 15.65 picoseconds. The actual unit may be calculated as 1/ (128*499.2×10^6) seconds. The IP_TOA value is available here when the leading edge determination and timestamp adjustments are completed (when the CIADONE status bit is set). The IP_TOA value is also written to the RX_STAMP field in *Sub-register 0x00:60 – Receive time stamp*, but that may be overwritten by the STS_TOA value if this is being computed by the CIA algorithm.
IP_POA reg:0C:04 bits:21–8	Phase of arrival as computed from the preamble CIR. It is adjusted by the state of the carrier recovery algorithm's correction when the CIR is finalized. This register is used when implementing a two-chip PDoA system.
- reg:0C:04 bits:23–22	These bits are unused / reserved.
IP_TOAST reg:0C:04 bits:31–24	Preamble sequence Time of Arrival status indicator. The low order 5-bits of this octet are a set of error flags indicating various conditions in the CIA algorithm that mean that the IP_TOA value is not reliable and should not be used. Status word of the preamble analysis [31] – Reserved [30] – Reserved [29] – Reserved [28] – First path too close to the end to be plausible [27] – Coarse first path estimate too close to end to be plausible [26] – CIR too weak to get any estimate [25] – Noise threshold had to be artificially lowered to find any first path. [24] – No strong rising edge on the first path so estimate is vulnerable to noise

8.2.13.2 Sub-register 0x0C:08 – STS receive time stamp and status

ID	Length (octets)	Type	Mnemonic	Description
0C:08	8	RO	STS_TS	STS receive time stamp and status

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x08 of register file 0x0C is an 8-octet register reporting the 40-bit STS based Time of Arrival estimate along with status diagnostic octet relating to it.

REG:0C:08 – STS_TS – STS receive time stamp and status (Octets 0 to 3, 32-bits)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS_TOA (low 32 bits of 40-bit value)																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REG:0C:0C – STS_TS – STS receive time stamp and status (Octets 4 to 7, 32-bits)																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS_TOAST									-	STS_POA														STS_TOA (high 8 bits of 40)							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the STS_TS register are described below:

Field	Description of fields within Sub-register 0x0C:08 – STS receive time stamp and status
STS_TOA reg:0C:08 bits:39-0	STS Time of Arrival estimate. This 40-bit (5-octet) field reports the time of reception as estimated by the CIA algorithm operating on the CIR accumulated using the STS sequence. The units of the low order bit are approximately 15.65 picoseconds. The actual unit may be calculated as 1/ (128*499.2×10^6) seconds. The STS_TOA value is available here when the leading edge determination and timestamp adjustments are completed (when the CIADONE status bit is set). Note the STS_TOA value is also written into the RX_STAMP field in *Sub-register 0x00:60 – Receive time stamp*. This will overwrite an existing value in the RX_STAMP field. So CP_TOA takes precedence over IP_TOA.
STS_POA reg:0C:0C bits:21–8	Phase of arrival as computed from the STS CIR and is adjusted by the state of the carrier recovery algorithm's correction when the CIR estimate is complete. It can be used in a two-chip PDoA system.
– reg:0C:0C bit:22	These bits are unused / reserved.
STS_TOAST reg:0C:08 bits:31-23	STS sequence Time of Arrival status indicator. These bits are a set of error flags indicating various conditions in the CIA algorithm. If STS_TOAST is non-zero it means that the CIR has failed one or more confidence tests. As a result, the STS_TOA and STS_POA value is not reliable and should not be used. Status bits from the STS analysis: [31] : STS_PGR_EN test fail. [30] : STS_SS_EN test fail. [29] : STS_CQ_EN test fail. [28] : First path too close to the end to be plausible. [27] : Coarse first path estimate too close to end to be plausible. [26] : CIR too weak to get any estimate. [25] : Noise threshold had to be artificially lowered to find any first path. [24] : No strong rising edge on the first path so estimate is vulnerable to noise. [23] : Reserved.

8.2.13.3 Sub-register 0x0C:10 – STS second RX time stamp and status

ID	Length (octets)	Type	Mnemonic	Description
0C:10	8	RO	STS1_TS	STS second receive time stamp and status (**valid in PDoA Mode 3 only**)

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x08 of register file 0x0C is an 8-octet register reporting the 40-bit STS second Time of Arrival estimate along with status diagnostic octet relating to it. This is only valid when operating in PDoA Mode 3.

REG:0C:10 – STS1_TS – STS second receive time stamp and status (Octets 0 to 3, 32-bits)																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS1_TOA (low 32 bits of 40-bit value)																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REG:0C:14 – STS1_TS – STS second receive time stamp and status (Octets 4 to 7, 32-bits)																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS1_TOAST									-	STS1_POA														STS1_TOA (high 8 bits of 40)							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the STS1_TS register are described below:

Field	Description of fields within Sub-register 0x0C:10 – STS second RX time stamp and status
STS1_TOA reg:0C:10 bits:39-0	STS second Time of Arrival estimate. This 40-bit (5-octet) field reports the time of reception as estimated by the CIA algorithm operating on the CIR accumulated using the STS. The units of the low order bit are approximately 15.65 picoseconds. The actual unit may be calculated as 1/ (128*499.2×10^6) seconds. The STS1_TOA value is available here when the leading edge determination and timestamp adjustments are completed (when the CIADONE status bit is set).
STS1_POA reg:0C:14 bits:21–8	Phase of arrival as computed from the STS based CIR estimate and is adjusted by the state of the carrier recovery algorithm's correction when the CIR estimate is complete. It can be used in a two-chip PDoA system.
– reg:0C:14 bit:22	These bits are unused / reserved.
STS1_TOAST reg:0C:14 bits:31-23	STS second Time of Arrival status indicator. These bits are a set of error flags indicating various conditions in the CIA algorithm. If STS_TOAST is non-zero it means that the CIR has failed one or more confidence tests. As a result, the STS1_TOA and STS1_POA value is not reliable and should not be used. Status bits from the STS1 analysis: [31] : STS_PGR_EN test fail. [30] : STS_SS_EN test fail. [29] : STS_CQ_EN test fail. [28] : First path too close to the end to be plausible. [27] : Coarse first path estimate too close to end to be plausible. [26] : CIR too weak to get any estimate. [25] : Noise threshold had to be artificially lowered to find any first path. [24] : No strong rising edge on the first path so estimate is vulnerable to noise. [23] : Reserved.

8.2.13.4 Sub-register 0x0C:18 – Time difference of arrival result

ID	Length (octets)	Type	Mnemonic	Description
0C:18	6	RO	TDOA	The TDoA between two CIRs

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x18 of register file 0x0C is a 6 octet register that provides the 41-bit value that is the TDoA between two CIR TOAs. The bit 40 is a sign bit. If PDoA Mode 1 is selected (see PDOA_MODE) then this TDoA refers to the TDoA between the IP_TOA and the STS_TOA value. In PDoA Mode 3 this is the TDoA between the STS_TOA and the STS1_TOA value. All three CIRs are estimates of the same (or a very similar) channel so the TDoA should be small. Its value can help when determining how reliable the PDoA/ToA estimate is. TDoA is described in 4.2 - TDoA and PDoA support.

8.2.13.5 Sub-register 0x0C:1E – Phase difference of arrival result

ID	Length (octets)	Type	Mnemonic	Description
0C:1E	2	RO	PDOA	PDoA and First Path (FP) threshold status

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x1E of register file 0x0C is a 2-octet register reporting the phase difference of arrival. This is only valid when operating one of PDoA modes (PDOA_MODE). The PDOA register contains the following bitmapped sub-fields:

REG:0C:1E – PDOA – PDoA result																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	FP_TH_MD	PDOA													
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of PDOA register are described below:

Field	Description of fields within Sub-register 0x0C:1E – Phase difference of arrival result
PDOA reg:0C:1E bits: 13–0	Phase difference result. This is further described in 4.2 TDoA and PDoA support.
FP_TH_MD reg:0C:1E bit: 14	First path threshold test mode. 0 - > absolute difference of two first paths is smaller or equal the FP_AGREED_TH 1 - > absolute difference is larger than FP_AGREED_TH
– reg:0C:1E bit:15	These bits are unused / reserved.

8.2.13.6 Sub-register 0x0C:20 – CIA diagnostic 0

ID	Length (octets)	Type	Mnemonic	Description
0C:20	4	RO	CIA_DIAG_0	CIA diagnostic 0

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x20 of register file 0x0C is a 4-octet register reporting CIA diagnostics after CIR analysis. The CIA_DIAG_0 register contains the following bitmapped sub-fields:

REG:0C:20 – CIA_DIAG_0 – CIA diagnostic 0																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	COE_PPM												
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of CIA_DIAG_0 register are described below:

Field	Description of fields within Sub-register 0x0C:20 – CIA diagnostic 0
COE_PPM reg:0C:20 bits: 12–0	Clock offset estimate. This 13-bit field reports an estimate of the clock offset of the remote transmitting device using an s[-15:-26] fixed point. The value is available here when the CIADONE status bit is set. To get PPM (part per million) the integer 13-bit value should be divided by 2^26 and then multiplied by 10^6.
– reg:0C:20 bits: 31–13	These bits are unused / reserved.

8.2.13.7 Sub-register 0x0C:24 – Reserved

ID	Length (octets)	Type	Mnemonic	Description
0C:24	4	RO	CIA_DIAG_1	Reserved diagnostic data

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x24 of register file 0x0C is a 4-octet reserved register.

8.2.13.8 Sub-register 0x0C:28 – Preamble diagnostic 0

ID	Length (octets)	Type	Mnemonic	Description
0C:28	4	RO	IP_DIAG_0	Preamble diagnostic 0 – peak

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x28 of register file 0x0C is a 4-octet register reporting diagnostics relating to the preamble sequence. The IP_DIAG_0 register contains the following bitmapped sub-fields:

REG:0C:28 – IP_DIAG_0 – Preamble diagnostic 0 – peak																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	IP_PEAKI										IP_PEAKA																				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of IP_DIAG_0 register are described below:

Field	Description of fields within Sub-register 0x0C:28 – Preamble diagnostic 0
IP_PEAKA reg:0C:18 bits:20–0	This 21-bit field reports the amplitude of the sample in the CIR accumulated using the preamble sequence that has the largest amplitude. This may be of diagnostic interest in certain circumstances. The value here is available when the CIADONE status bit is set.
IP_PEAKI reg:0C:18 bits:30–21	This 10-bit field reports the index of the sample in the CIR accumulated using the preamble sequence that has the largest amplitude. The value here is available when the CIADONE status bit is set.
reg:0C:18 bit:31	This bit is reserved.

8.2.13.9 Sub-register 0x0C:2C – Preamble diagnostic 1

ID	Length (octets)	Type	Mnemonic	Description
0C:2C	4	RO	IP_DIAG_1	Preamble diagnostic 1 –power indication

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x2C of register file 0x0C is a 4-octet register reporting diagnostics relating to the preamble sequence. The IP_DIAG_1 register contains the following bitmapped sub-fields:

REG:0C:2C – IP_DIAG_1 – Preamble diagnostic 1 –power indication																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	IP_CAREA																			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of IP_DIAG_1 register are described below:

Field	Description of fields within Sub-register 0x0C:2C – Preamble diagnostic 1
IP_CAREA reg:0C:2C bits:19–0	This 20-bit field reports the channel area in the CIR accumulated using the preamble sequence. This gives a measure of the receive power of the preamble. This may be of diagnostic interest in certain circumstances. The value here is available when the CIADONE status bit is set.
- reg:0C:2C bit:31–20	Bits marked '-' in this register are reserved.

8.2.13.10 Sub-register 0x0C:30 – Preamble diagnostic 2

ID	Length (octets)	Type	Mnemonic	Description
0C:30	4	RO	IP_DIAG_2	Preamble diagnostic 2 –magnitude @ FP + 1

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x30 of register file 0x0C is a 4-octet register reporting diagnostics relating to the preamble sequence. The IP_DIAG_2 register contains the following bitmapped sub-fields:

REG:0C:30 – IP_DIAG_2 – Preamble diagnostic 2 –magnitude @ FP + 1																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	IP_FP1M																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of IP_DIAG_2 register are described below:

Field	Description of fields within Sub-register 0x0C:30 – Preamble diagnostic 2
IP_FP1M reg:0C:30 bits:21–0	This 22-bit field reports the magnitude of the sample at the first index immediately after the estimated first path position in the CIR accumulated using the preamble sequence. This may be of diagnostic interest in certain circumstances. The value here is available when the CIADONE status bit is set.
- reg:0C:30 bit:31–22	Bits marked '-' in this register are reserved.

8.2.13.11 Sub-register 0x0C:34 – Preamble diagnostic 3

ID	Length (octets)	Type	Mnemonic	Description
0C:34	4	RO	IP_DIAG_3	Preamble diagnostic 3 – magnitude @ FP + 2

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x34 of register file 0x0C is a 4-octet register reporting diagnostics relating to the preamble sequence. The IP_DIAG_3 register contains the following bitmapped sub-fields:

REG:0C:34 – IP_DIAG_3 – Preamble diagnostic 3 –magnitude @ FP + 2																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	IP_FP2M																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of IP_DIAG_3 register are described below:

Field	Description of fields within Sub-register 0x0C:34 – Preamble diagnostic 3
IP_FP2M reg:0C:34 bits:21–0	This 22-bit field reports the magnitude of the sample at second index after the estimated first path position in the CIR accumulated using the preamble sequence. The value here is available when the CIADONE status bit is set.
- reg:0C:34 bit:31–22	Bits marked '-' in this register are reserved.

8.2.13.12 Sub-register 0x0C:38 – Preamble diagnostic 4

ID	Length (octets)	Type	Mnemonic	Description
0C:38	4	RO	IP_DIAG_4	Preamble Diagnostic 4 –magnitude @ FP + 3

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x38 of register file 0x0C is a 4-octet register reporting diagnostics relating to the preamble sequence. The IP_DIAG_4 register contains the following bitmapped sub-fields:

REG:0C:38 – IP_DIAG_4 – Preamble diagnostic 4 –magnitude @ FP + 3																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	IP_FP3M																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of IP_DIAG_4 register are described below:

Field	Description of fields within Sub-register 0x0C:38 – Preamble diagnostic 4
IP_FP3M reg:0C:38 bits:21–0	This 22-bit field reports the magnitude of the sample at third index after the estimated first path position in the CIR accumulated using the preamble sequence. The value here is available when the CIADONE status bit is set.
- reg:0C:38 bit:31–22	Bits marked '-' in this register are reserved.

8.2.13.13 Sub-register 0x0C:48 – Preamble diagnostic 8

ID	Length (octets)	Type	Mnemonic	Description
0C:48	4	RO	IP_DIAG_8	Preamble diagnostic 8 –first path result

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x48 of register file 0x0C is a 4-octet register reporting diagnostics relating to the preamble sequence. The IP_DIAG_8 register contains the following bitmapped sub-fields:

REG:0C:48 – IP_DIAG_8 – Preamble diagnostic 8 –first path																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	IP_FP															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of IP_DIAG_8 register are described below:

Field	Description of fields within Sub-register 0x0C:48 – Preamble diagnostic 8
IP_FP reg:0C:48 bits:15–0	This 16-bit field reports the estimated first path location within the CIR accumulated using the preamble sequence. The value here uses a [10.6] fixed point format with a 10-bit unsigned integer part and a 6-bit fractional part. This value is available when the CIADONE status bit is set. This is the index relative to the start of preamble CIR within the accumulator, i.e. beginning at accumulator index 0.
- reg:0C:48 bit:31–16	Bits marked '-' in this register are reserved.

8.2.13.14 Sub-register 0x0C:58 – Preamble diagnostic 12

ID	Length (octets)	Type	Mnemonic	Description
0C:58	4	RO	IP_DIAG_12	Diagnostic 12 – symbols accumulated

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x58 of register file 0x0C is a 4-octet register reporting diagnostics relating to the preamble sequence. The IP_DIAG_12 register contains the following bitmapped sub-fields:

REG:0C:58 – IP_DIAG_12 – Preamble diagnostic 12 – symbols accumulated																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	IP_NACC											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of IP_DIAG_12 register are described below:

Field	Description of fields within Sub-register 0x0C:58 – Preamble diagnostic 12
IP_NACC reg:0C:58 bits:11–0	This 12-bit field reports the number of preamble sequence symbols that were accumulated to form the preamble CIR. This value is available after the CIADONE status bit is set.
- reg:0C:58 bit:31–12	Bits marked '-' in this register are reserved.

8.2.13.15 Sub-register 0x0C:5C – STS diagnostic 0

ID	Length (octets)	Type	Mnemonic	Description
0C:5C	4	RO	STS_DIAG_0	STS Diagnostic 0 – STS CIA peak amplitude

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x5C of register file 0x0C is a 4-octet register reporting diagnostics relating to the STS. The STS_DIAG_0 register contains the following bitmapped sub-fields:

REG:0C:5C – STS_DIAG_0 – STS diagnostic 0 – STS CIA peak amplitude																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	CP_PEAKI									CP_PEAKA																				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS_DIAG_0 register are described below:

Field	Description of fields within Sub-register 0x0C:5C – STS diagnostic 0
CP_PEAKA reg:0C:5C bits:20–0	This 21-bit field reports the amplitude of the sample in the CIR accumulated using the STS that has the largest amplitude. This may be of diagnostic interest in certain circumstances. The value here is available when the CIADONE status bit is set.
CP_PEAKI reg:0C:5C bits:29–21	This 9-bit field reports the index of the sample in the CIR accumulated using the STS that has the largest amplitude. The value here is available when the CIADONE status bit is set.
- reg:0C:5C bits:31,30	These bits are reserved.

8.2.13.16 Sub-register 0x0C:60 – STS 0 diagnostic 1

ID	Length (octets)	Type	Mnemonic	Description
0C:60	4	RO	STS_DIAG_1	STS Diagnostic 1 – STS power indication

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x60 of register file 0x0C is a 4-octet register reporting diagnostics relating to the STS. The STS_DIAG_1 register contains the following bitmapped sub-fields:

REG:0C:60 – STS_DIAG_1 – STS diagnostic 1 – STS power indication																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	CP_CAREA																			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS_DIAG_1 register are described below:

Field	Description of fields within Sub-register 0x0C:60 – STS 0 diagnostic 1
CP_CAREA reg:0C:60 bits:19–0	This 20-bit field reports the channel area in the CIR accumulated using the STS. This gives a measure of the receive power of the STS. This may be of diagnostic interest in certain circumstances. The value here is available when the CIADONE status bit is set.
- reg:0C:60 bit:31–20	Bits marked '-' in this register are reserved.

8.2.13.17 Sub-register 0x0C:64 – STS 0 diagnostic 2

ID	Length (octets)	Type	Mnemonic	Description
0C:64	4	RO	STS_DIAG_2	STS Diagnostic 2 – STS magnitude @ FP + 1

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x64 of register file 0x0C is a 4-octet register reporting diagnostics relating to the STS. The STS_DIAG_2 register contains the following bitmapped sub-fields:

REG:0C:64 – STS_DIAG_2 – STS diagnostic 2 – STS magnitude @ FP + 1																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	CP_FP1M																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS_DIAG_2 register are described below:

Field	Description of fields within Sub-register 0x0C:64 – STS 0 diagnostic 2
CP_FP1M reg:0C:64 bits:21–0	This 22-bit field reports the magnitude of the sample at the first index immediately after the estimated first path position in the CIR accumulated using the STS. This may be of diagnostic interest in certain circumstances. The value here is available when the CIADONE status bit is set.
- reg:0C:64 bit:31–22	Bits marked '-' in this register are reserved.

8.2.13.18 Sub-register 0x0C:68 – STS 0 diagnostic 3

ID	Length (octets)	Type	Mnemonic	Description
0C:68	4	RO	STS_DIAG_3	STS Diagnostic 3 – STS magnitude @ FP + 2

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 068 of register file 0x0C is a 4-octet register reporting diagnostics relating to the STS. The STS_DIAG_3 register contains the following bitmapped sub-fields:

REG:0C:68 – STS_DIAG_3 – STS diagnostic 3 – STS magnitude @ FP + 2																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	CP_FP2M																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS_DIAG_3 register are described below:

Field	Description of fields within Sub-register 0x0C:68 – STS 0 diagnostic 3
CP_FP2M reg:0C:68 bits:21–0	This 22-bit field reports the magnitude of the sample at second index after the estimated first path position in the CIR accumulated using the STS. The value here is available when the CIADONE status bit is set.
- reg:0C:68 bit:31–22	Bits marked '-' in this register are reserved.

8.2.13.19 Sub-register 0x0D:00 – STS 0 diagnostic 4

ID	Length (octets)	Type	Mnemonic	Description
0D:00	4	RO	STS_DIAG_4	STS diagnostic 4 – STS magnitude @ FP + 3

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x00 of register file 0x0D is a 4-octet register reporting diagnostics relating to the STS. The STS_DIAG_4 register contains the following bitmapped sub-fields:

REG:0D:00 – STS_DIAG_4 – STS diagnostic 4 – STS magnitude @ FP + 3																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	CP_FP3M																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS_DIAG_4 register are described below:

Field	Description of fields within Sub-register 0x0D:00 – STS 0 diagnostic 4
CP_FP3M reg:0D:00 bits:21–0	This 22-bit field reports the magnitude of the sample at third index after the estimated first path position in the CIR accumulated using the STS. The value here is available when the CIADONE status bit is set.
- reg:0D:00 bit:31–22	Bits marked '-' in this register are reserved.

8.2.13.20 Sub-register 0x0D:10 – STS 0 diagnostic 8

ID	Length (octets)	Type	Mnemonic	Description
0D:10	4	RO	STS_DIAG_8	STS Diagnostic 8 – STS first path

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x10 of register file 0x0D is a 4-octet register reporting diagnostics relating to the STS. The STS_DIAG_8 register contains the following bitmapped sub-fields:

REG:0D:10 – STS_DIAG_8 – STS diagnostic 8 – STS first path																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CP_FP														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS_DIAG_8 register are described below:

Field	Description of fields within Sub-register 0x0D:10 – STS 0 diagnostic 8
CP_FP reg:0D:10 bits:14–0	This 15-bit field reports the estimated first path location within the CIR accumulated using the STS. The value here uses a [9.6] fixed point format with a 9-bit unsigned integer part and a 6-bit fractional part. This value is available when the CIADONE status bit is set. This is the index relative to the start of the STS CIR within the accumulator, i.e. beginning at accumulator index 1024.
- reg:0D:10 bit:31–15	Bits marked '-' in this register are reserved.

8.2.13.21 Sub-register 0x0D:20 – STS 0 diagnostic 12

ID	Length (octets)	Type	Mnemonic	Description
0D:20	4	RO	STS_DIAG_12	STS diagnostic 12 – accumulated STS length

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x20 of register file 0x0D is a 4-octet register reporting diagnostics relating to the STS. The STS_DIAG_12 register contains the following bitmapped sub-fields:

REG:0D:20 – STS_DIAG_12 – STS diagnostic 12 – accumulated STS length																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CP_NACC											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS_DIAG_12 register are described below:

Field	Description of fields within Sub-register 0x0D:20 – STS 0 diagnostic 12
CP_NACC reg:0D:20 bits:11–0	This 11-bit field reports the length of STS that has been accumulated in units of 512 chips (~1 µs) to form the STS CIR. This value is available after the CIADONE status bit is set.
- reg:0D:20 bit:31–24	Bits marked '-' in this register are reserved.

8.2.13.22 Sub-register 0x0D:38 – STS 1 diagnostic 0

ID	Length (octets)	Type	Mnemonic	Description
0D:38	4	RO	STS1_DIAG_0	STS 1 diagnostic 0 – peak amplitude

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x38 is a 4-octet register reporting diagnostics relating to the STS where it is present and being used to measure the PDOA.This register will only be valid when PDOA mode 3 is used. The STS1_DIAG_0 register contains the following bitmapped sub-fields:

REG:0D:38 – STS1_DIAG_0 – STS 1 diagnostic 0 – peak amplitude																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	CP_PEAKI									CP_PEAKA																				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS1_DIAG_0 register are described below:

Field	Description of fields within Sub-register 0x0D:38 – STS 1 diagnostic 0
CP_PEAKA reg:0D:38 bits:20–0	This 21-bit field reports the amplitude of the sample in the CIR accumulated using the STS that has the largest amplitude. This may be of diagnostic interest in certain circumstances. The value here is available when the CIADONE status bit is set.

Field	Description of fields within Sub-register 0x0D:38 – STS 1 diagnostic 0
CP_PEAKI reg:0D:38 bits:29–21	This 9-bit field reports the index of the sample in the CIR accumulated using the STS that has the largest amplitude. The value here is available when the CIADONE status bit is set.
- reg:0D:38 bits:31,30	These bits are reserved.

8.2.13.23 Sub-register 0x0D:3C – STS 1 diagnostic 1

ID	Length (octets)	Type	Mnemonic	Description
0D:3C	4	RO	STS1_DIAG_1	STS 1 diagnostic 1 – STS 1 power indication

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x3C of register file 0x0D is a 4-octet register reporting diagnostics relating to the STS where it is present and being used to measure the PDOA. This register will only be valid when PDOA mode 3 is used. The STS1_DIAG_1 register contains the following bitmapped sub-fields:

REG:0D:3C – STS1_DIAG_1 – STS 1 diagnostic 1 – STS 1 power indication																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	CP_CAREA																			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit CIA_CONF has been cleared to zero.

The fields of STS1_DIAG_1 register are described below:

Field	Description of fields within Sub-register 0x0D:3C – STS 1 diagnostic 1
CP_CAREA reg:0D:3C bits:19–0	This 20-bit field reports the channel area in the CIR accumulated using the STS. This gives a measure of the receive power of the STS. This may be of diagnostic interest in certain circumstances. The value here is available when the CIADONE status bit is set.
- reg:0D:3C bit:31–20	Bits marked '-' in this register are reserved.

8.2.13.24 Sub-register 0x0D:40 – STS 1 diagnostic 2

ID	Length (octets)	Type	Mnemonic	Description
0D:40	4	RO	STS1_DIAG_2	STS 1 diagnostic 2 – STS 1 magnitude @ FP + 1

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x40 of register file 0x0D is a 4-octet register reporting diagnostics relating to the STS where it is present and being used to measure the PDOA. This register will only be valid when PDOA mode 3 is used. The STS1_DIAG_2 register contains the following bitmapped sub-fields:

REG:0D:40 – STS1_DIAG_2 – STS 1 diagnostic 2 – STS 1 magnitude @ FP + 1																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	CP_FP1M																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS1_DIAG_2 register are described below:

Field	Description of fields within Sub-register 0x0D:40 – STS 1 diagnostic 2
CP_FP1M reg:0D:40 bits:21–0	This 22-bit field reports the magnitude of the sample at the first index immediately after the estimated first path position in the CIR accumulated using the STS. This may be of diagnostic interest in certain circumstances. The value here is available when the CIADONE status bit is set.
- reg:0D:40 bit:31–22	Bits marked '-' in this register are reserved.

8.2.13.25 Sub-register 0x0D:44 – STS 1 diagnostic 3

ID	Length (octets)	Type	Mnemonic	Description
0D:44	4	RO	STS1_DIAG_3	STS 1 diagnostic 3 – STS 1 magnitude @ FP + 2

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 044 of register file 0x0D is a 4-octet register reporting diagnostics relating to the STS where it is present and being used to measure the PDOA. This register will only be valid when PDOA mode 3 is used. The STS1_DIAG_3 register contains the following bitmapped sub-fields:

REG:0D:44 – STS1_DIAG_3 – STS 1 diagnostic 3 – STS 1 magnitude @ FP + 2																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	CP_FP2M																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS1_DIAG_3 register are described below:

Field	Description of fields within Sub-register 0x0D:44 – STS 1 diagnostic 3
CP_FP2M reg:0D:44 bits:21–0	This 22-bit field reports the magnitude of the sample at second index after the estimated first path position in the CIR accumulated using the STS. The value here is available when the CIADONE status bit is set.
- reg:0D:44 bit:31–22	Bits marked '-' in this register are reserved.

8.2.13.26 Sub-register 0x0D:48 – STS 1 diagnostic 4

ID	Length (octets)	Type	Mnemonic	Description
0D:48	4	RO	STS1_DIAG_4	STS 1 diagnostic 4 – STS 1 magnitude @ FP + 3

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x48 of register file 0x0D is a 4-octet register reporting diagnostics relating to the STS where it is present and being used to measure the PDOA. This register will only be valid when PDOA mode 3 is used. The STS1_DIAG_4 register contains the following bitmapped sub-fields:

REG:0D:48 – STS1_DIAG_4 – STS 1 diagnostic 4 – STS 1 magnitude @ FP + 3																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	CP_FP3M																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS1_DIAG_4 register are described below:

Field	Description of fields within Sub-register 0x0D:48 – STS 1 diagnostic 4
CP_FP3M reg:0D:48 bits:21–0	This 22-bit field reports the magnitude of the sample at third index after the estimated first path position in the CIR accumulated using the STS. The value here is available when the CIADONE status bit is set.
- reg:0D:48 bit:31–22	Bits marked '-' in this register are reserved.

8.2.13.27 Sub-register 0x0D:58 – STS 1 diagnostic 8

ID	Length (octets)	Type	Mnemonic	Description
0D:58	4	RO	STS1_DIAG_8	STS 1 diagnostic 8 – STS 1 first path

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x58 of register file 0x0D is a 4-octet register reporting diagnostics relating to the STS where it is present and being used to measure the PDOA. This register will only be valid when PDOA mode 3 is used. The STS1_DIAG_8 register contains the following bitmapped sub-fields:

REG:0D:58 – STS1_DIAG_8 – STS 1 diagnostic 8 – STS 1 first path																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CP_FP														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit in CIA_CONF has been cleared to zero.

The fields of STS1_DIAG_8 register are described below:

Field	Description of fields within Sub-register 0x0D:58 – STS 1 diagnostic 8
CP_FP reg:0D:58 bits:14–0	This 15-bit field reports the estimated first path location within the CIR accumulated using the STS where it is present and being used to measure the PDOA. The value here uses a [9.6] fixed point format with a 9-bit unsigned integer part and a 6-bit fractional part. This value is available when the CIADONE status bit is set. This is the index relative to the start of the STS CIR within the accumulator, i.e. beginning at accumulator index 1024.
- reg:0D:58 bit:31–15	Bits marked '-' in this register are reserved.

8.2.13.28 Sub-register 0x0D:68 – STS 1 diagnostic 12

ID	Length (octets)	Type	Mnemonic	Description
0D:68	4	RO	STS1_DIAG_12	STS 1 diagnostic 12 – STS 1 accumulated STS length

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x68 of register file 0x0D is a 4-octet register reporting diagnostics relating to the STS where it is present and being used to measure the PDOA. This register will only be valid when PDOA mode 3 is used. The STS1_DIAG_12 register contains the following bitmapped sub-fields:

REG:0D:68 – STS1_DIAG_12 – STS 1 diagnostic 12 – STS 1 accumulated STS length																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CP_NACC											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Please note: This diagnostics register will not be updated unless the MINDIAG configuration bit CIA_CONF has been cleared to zero.

The fields of STS1_DIAG_12 register are described below:

Field	Description of fields within Sub-register 0x0D:68 – STS 1 diagnostic 12
CP_NACC reg:0D:68 bits:11–0	This 12-bit field reports the length of STS that has been accumulated in units of 512 chips (~1 µs) to form the STS CIR. This value is available after the CIADONE status bit is set.
- reg:0D:68 bit:31–11	Bits marked '-' in this register are reserved.

8.2.13.29 Sub-register 0x0E:00 – RX antenna delay and CIA diagnostic enable

ID	Length (octets)	Type	Mnemonic	Description
0E:00	4	RW	CIA_CONF	RX antenna delay and CIA diagnostic enable

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x00 of register file 0x0E is a 4-octet configuration register. The CIA_CONF register contains the following bitmapped sub-fields:

REG:0E:00 – CIA_CONF – RX antenna delay and CIA diagnostic enable

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	MINDIAG	-	-	-	-	RXANTD															
0	0	0	1	0	1	1	0	0	0	1	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	1

The fields of CIA_CONF register are described below:

Field	Description of fields within Sub-register 0x0E:00 – RX antenna delay and CIA diagnostic enable
RXANTD reg:0E:00 bits:15–0	This 16-bit field configures the receive antenna delay. The receiver antenna delay is used to account for the delay between the arrival of the RMARKER at the antenna and the time the RMARKER is detected and time-stamped by the internal digital RX circuitry. The units here are the same as those used for system time and time stamps, i.e. 1/(499.2 MHz × 128), so the least significant bit about 15.65 picoseconds. . The default antenna delay is 0x4015, which is approximately 256.74 ns. The value programmed in RXANTD register value is subtracted (by the CIA algorithm) from the raw timestamp *RX_RAWST* to by the CIA algorithm which performs a number of other updates and adjustments(including detecting and accounting for the first path position in the accumulator) in order to generate the fully adjusted *RX_STAMP* value also in *Sub-register 0x00:60 – Receive time stamp*. Please see § *10.3 – IC calibration – antenna delay* for details of calibration of antenna delay.

Field	Description of fields within Sub-register 0x0E:00 – RX antenna delay and CIA diagnostic enable
MINDIAG reg:0E:00 bit:20	Minimum Diagnostics. This bit applies to the operation of the CIA for both preamble and STS sequences. This bit specifies if the CIA algorithm will log data in the CIA diagnostic registers. The MINDIAG bit is 1 by default to minimise the diagnostics information generated by the CIA process. To generate the extended diagnostics MINDIAG may be cleared to 0, however **please note:** when MINDIAG = 0, the CIA processing takes longer, delaying the assertion of the CIADONE and RXFR event status flag bits, (i.e. so ranging exchanges will take a little longer and consume a little more energy). When MINDIAG is 1 the CIA does NOT update any of the preamble CIA diagnostic registers (IP_DIAG_0 – IP_DIAG_12) nor any of the STS CIA diagnostic registers (STS_DIAG_0–STS_DIAG_12) nor any of the second STS CIA diagnostic registers (STS1_DIAG_0 –STS1_DIAG_12) when PDoA Mode 3 is used.
- reg:0E:00 bits:various	Bits marked '-' in this register are reserved. **N.B.: Any change in these bits may cause the QM33100 to malfunction.**

8.2.13.30 Sub-register 0x0E:04 – First path Confidence Limit

ID	Length (octets)	Type	Mnemonic	Description
0E:04	4	RW	FP_CONF	First path confidencethresholds

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x04 of register file 0x0E is a 4-octet register containing first path confidencethreshold values. The FP_CONF register contains the following control bits:

REG:0E:04 – FP_CONF – First path confidence thresholds

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	TC_RXDLY_EN	-	CAL_TEMP								FP_AGREED_TH			-	-	-	-	-	-	-	-
0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0

The fields of FP_CONF register are described below:

Field	Description of fields within Sub-register 0x0E:04 – First path Confidence Limit
FP_AGREED_TH reg:0E:04 bits:10–8	The threshold to use when performing the FP_AGREE test. The LSB is 1/(998.4 MHz) (approximately 1ns).
CAL_TEMP reg:0E:04 bits:18–11	Temperature at which the device was calibrated. The units are those from the on-chip temperature sensor, as read from SAR_LTEMP in SAR_READING register.

Field	Description of fields within Sub-register 0x0E:04 – First path Confidence Limit
TC_RXDLY_EN reg:0E:04 bit:20	Temperature compensation for RX antenna delay. When set to 1, the device will compensate for temperature differences. Must specify the calibration temperature (CAL_TEMP above). Its default corresponds to absolute zero Kelvin.
- reg:0E:04 bit: various	Bits marked '-' in this register are reserved. **N.B.: Any change in these bits may cause the QM33100 to malfunction.**

8.2.13.31 Sub-register 0x0E:0C – CIA preamble configuration

ID	Length (octets)	Type	Mnemonic	Description
0E:0C	6	RW	IP_CONF	Preamble config – CIA preamble configuration

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x0C of register file 0x0E is a 6-octet configuration register relating to the CIA configuration for the preamble sequence CIR analysis. The IP_CONF register contains the following control bits:

REG:0E:0C – IP_CONF – CIA preamble configuration (Octets 0 to 3)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																		IP_SPEQE	-	-	-	-	-	-	IP_PMULT		IP_NTM				
																		0	0	0	0	1	0	0	1	0	0	1	1	0	1

REG:0E:10 – IP_CONF – CIA preamble configuration (Octets 4 and 5)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																											IP_RTM				
																											0	1	1	0	1

The fields of IP_CONF register are described below:

<table>
<tr><th>Field</th><th>Description of fields within Sub-register 0x0E:0C – CIA preamble configuration</th></tr>
<tr><td>IP_NTM

reg:0E:0C
bits:4–0</td><td>Preamble Noise Threshold Multiplier. This 5-bit field is a factor by which the measured noise level in the preamble generated CIR is multiplied to set the threshold for the CIA algorithm’s first path search. The value here sets the compromise between falsely triggering on noise peaks and missing real attenuated (non-line-of-sight) first paths. Where NLOS performance is more important to the application a lower value, of 12 say, could be used. The resulting threshold shall not be smaller than the value defined, but may be larger depending on other factors used to choose the threshold.</td></tr>
<tr><td>IP_PMULT

reg:0E:0C
bits:6–5</td><td>Preamble Peak Multiplier. This 2-bit field sets a factor by which the peak value of estimated noise is increased in order to set the threshold for first path searching. The values of IP_PMULT and the resultant peak multiplier value are given by the table below.

<table><tr><th>IP_PMULT value</th><th>Peak Multiplier</th></tr><tr><td>0</td><td>1.00</td></tr><tr><td>1</td><td>1.25</td></tr><tr><td>2 (default)</td><td>1.50</td></tr><tr><td>3</td><td>1.75</td></tr></table>
The resulting threshold shall not be smaller than the value defined by this, but may be larger depending on other factors used to choose the threshold.</td></tr>
<tr><td>IP_SPEQE

reg:0E:0C
bits:13</td><td>SRRC Pulse Equaliser Enable. When set to 1, this IP_SPEQE bit enables an equaliser that adjusts the CIR to give improved receive timestamp results when the remote transmitter is using a Symmetric Root Raised Cosine pulse shape. Normally, this IP_SPEQE bit should be set to zero (the default value), which gives the best receive timestamp performance when interworking with devices (like this IC) that use the IEEE 802.15.4z recommended minimum precursor pulse shape.</td></tr>
<tr><td>IP_RTM

reg:0E:10
bits:4–0</td><td>Preamble replica threshold multiplier, IP_RTM, is a 5-bit field. The accumulation of the preamble sequence to produce the preamble CIR is not perfect when there is a significant clock offset between the remote transmitter and the local receiver. In these circumstances small amplitude replicas of the channel impulse response appear repeatedly throughout the accumulator span. The magnitude of this effect is dependent on the clock offset and on the preamble code being employed. The QM33100 automatically measures the clock offset and computes a preamble code appropriate replica avoidance threshold value to avoid the replicas. The IP_RTM parameter allows for the possibility of tuning the choice of replica avoidance threshold, however it is not expected that the default value of 4 will need to be modified.</td></tr>
<tr><td>-
reg:0E:0C
bit:various</td><td>Bits marked ‘-’ in this register are reserved.
N.B.: Any change in these bits may cause the QM33100 to malfunction.</td></tr>
</table>

8.2.13.32 Sub-register 0x0E:14 – CIA STS configuration 0

ID	Length (octets)	Type	Mnemonic	Description
0E:14	4	RW	STS_CONF_0	STS Config 0 – CIA configuration for STS

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x14 of register file 0x0E is a 4-octet configuration register relating to the CIA configuration for the STS based CIR analysis. The STS_CONF_0 register contains the following control bits:

REG:0E:14 – STS_CONF_0 – CIA configuration for STS

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	STS_MNTH							-	-	-	-	-	-	-	-	-	STS_PMULT		STS_NTM				
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	1	1	0	0

The fields of STS_CONF_0 register fields are described below:

Field	Description of fields within Sub-register 0x0E:14 – CIA STS configuration 0
STS_NTM reg:0E:14 bits:4–0	STS Noise Threshold Multiplier. This 5-bit field is a factor by which the measured noise level in the STS generated CIR is multiplied to set the threshold for the CIA algorithm's first path search. The value here sets the compromise between falsely triggering on noise peaks and missing real attenuated (non-line-of-sight) first paths.
STS_PMULT reg:0E:14 bits:6–5	STS Peak Multiplier. This 2-bit field sets a factor by which the peak value of the estimated noise in the STS generated CIR is increased in order to set the threshold for first path searching. The values of STS_PMULT and the resultant peak multiplier value are given by the table below. STS_PMULT value / Peak Multiplier 0 (default) / 0.00 1 / 1.25 2 / 1.50 3 / 1.75 The resultant threshold is constrained to be not smaller than the value defined by this STS_PMULT parameter, but it may be larger depending on other factors used to select the threshold value. A value of 0 allows the first path to be in region of the CIR that is used for gathering noise statistics. This helps when the channel is severely NLOS with a large delay spread.
- reg:0E:14 bits:15-7	Bits marked '-' in this register are reserved. **N.B.: Any change in these bits field may cause the QM33100 to malfunction.**

Field	Description of fields within Sub-register 0x0E:14 – CIA STS configuration 0
STS_MNTH reg:0E:14 bits:22–16	STS Minimum Threshold. This 7-bit field sets a lower bound on the threshold for first path searching. In low noise scenarios the threshold resulting from the action of the STS_NTM and STS_PMULT parameters may end up so small that the CIA finds a noise peak in its search for the first path. The resultant threshold is constrained not to be smaller than the value defined by this STS_MNTH parameter. The default STS_MNTH value is 12 which is suitable for the default 64 MHz PRF and STS length of 64 (CPS_LEN = 7). The table below gives the recommended values for other STS lengths PRF / STS length in units of 512 chips (~1 µs) / STS_MNTH value (PDoA Mode Off/1 / PDoA Mode 3): 64 MHz / 64 / 12 / 0 64 MHz / 128 / 17 / 12 64 MHz / 256 / 24 / 17 Note these values are scaled by a factor of $\sqrt{2}$ as the STS length is doubled from 64 to 128 and again when increasing to 256 and so on.
- reg:0E:14 bit:31–23	Bits marked '-' in this register are reserved. **N.B.: Any change in these bits field may cause the QM33100 to malfunction.**

8.2.13.33 Sub-register 0x0E:18 – CIA STS configuration 1

ID	Length (octets)	Type	Mnemonic	Description
0E:18	4	RW	STS_CONF_1	STS Config 1 – CIA configuration for STS

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x18 of register file 0x0E is a 4-octet configuration register relating to the CIA configuration for the STS based CIR analysis. The STS_CONF_1 register contains the following control bits:

REG:0E:18 – STS_CONF_1 – CIA configuration for STS

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS_PGR_EN	STS_SS_EN	STS_CQ_EN	FP_AGREED_EN	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	–	-	-	-	-	RES_B0							
1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	0	1	1	1	0	1	1	0	1	1	0	0	1	1	0	1	1

The individual fields of STS_CONF_1 register are described below:

Field	Description of fields within Sub-register 0x0E:18 – CIA STS configuration 1
RES_B0 reg:0E:18 bits:7-0	This 8-bit reserved tuning value should be changed from default 0x9B to 0x94 for optmimised performance.
FP_AGREED_EN reg:0E:18 bit:28	Checks to see if the two ToA estimates are within allowed tolerances. Tolerance is set in FP_AGREED_TH.
STS_CQ_EN reg:0E:18 bit:29	Checks how consistent the impulse response stays during the accumulation of the STS. Since the RF channel should remain fairly constant over the sequence this check helps to ensure the integrity of the STS based RX timestamp. If the test fails, the appropriate bit in the STS_TOAST status indicator will be set to indicate that the STS_TOA value is not reliable and should not be used.
STS_SS_EN reg:0E:18 bit:30	Compare the sampling statistics of the STS reception to those of the earlier reception of the preamble sequence. Since these should relate to the same transmitter over the same channel, the preamble and STS sequence statistics should match (within a certain tolerance). If the test fails, the appropriate bit in the STS_TOAST status indicator will be set to indicate that the STS_TOA value is not reliable and should not be used.
STS_PGR_EN reg:0E:18 bit:31	Test the growth rate of the STS based CIR to the earlier growth rate of the preamble based CIR. Since both the preamble and the STS are estimating the channel impulse response of the same channel they should grow at the same rate (within a certain tolerance). If this test fails, the appropriate bit in the STS_TOAST status indicator will be set to indicate that the STS_TOA value is not reliable and should not be used.
- reg:0E:18 bits:various	Bits marked '-' in this register are reserved. **N.B.: Any change in these bits field may cause the QM33100 to malfunction.**

8.2.13.34 Sub-register 0x0E:1C – CIA adjustment

ID	Length (octets)	Type	Mnemonic	Description
0E:1C	4	RW	CIA_ADJUST	User adjustments to the CIA PDoA calculation

Register files: 0x0C, 0x0D, 0x0E – CIA Interface, sub-register 0x1C of register file 0x0E is a 2-octet configuration register relating to the CIA configuration for the STS based CIR analysis. The CIA_ADJUST register contains the following control bits:

REG:0E:1C – PDOA_ADJ – User adjustments to the PDoA																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	PDOA_ADJ													
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The sub-fields of this CIA_ADJUST register are as follows:

Field	Description of fields within Sub-register 0x0E:1C – CIA adjustment
PDOA_ADJ reg:0E:1C bits: 13:0	Adjustment value to account for non-balanced antenna circuits. The LSB is 2^{-11} radians. This angle is simply added to the PDoA that is estimated from the CIR. It is interpreted by the device as an unsigned integer. For example, a value comprised between 0 and 12868 will represent an offset comprised in 0 to 2π radians interval.
- reg:0E:1C bits:15–14	Bits marked '-' in this register are reserved. **N.B.: Any change in these bits field may cause the QM33100 to malfunction.**

8.2.14 Register file: 0x0F – Digital diagnostics interface

ID	Length (octets)	Type	Mnemonic	Description
0x0F	89	-	DIG_DIAG	Digital diagnostics interface

Register map register file 0x0F is the Digital Diagnostics interface. It contains a number of Sub-registers that give diagnostics information. An overview of these is given by Table 42. Each of these Sub-registers is separately described in the sub-sections below.

Table 42: Register file: 0x0F – Digital diagnostics interface overview

OFFSET in Register 0x0F	Mnemonic	Description
0x00	EVC_CTRL	Event counter control
0x04	EVC_PHE	PHR error counter
0x06	EVC_RSE	RSD error counter
0x08	EVC_FCG	Frame check sequence good counter
0x0A	EVC_FCE	Frame check sequence error counter
0x0C	EVC_FFR	Frame filter rejection counter
0x0E	EVC_OVR	RX overrun error counter
0x10	EVC_STO	SFD timeout counter
0x12	EVC_PTO	Preamble timeout counter
0x14	EVC_FWTO	RX frame wait timeout counter
0x16	EVC_TXFS	TX frame sent counter
0x18	EVC_HPW	Half period warning counter
0x1A	EVC_SWCE	SPI write CRC error counter
0x20	EVC_CPQE	STS quality error counter
0x22	EVC_VWARN	Low voltage warning error counter
0x28	DIAG_TMC	Test mode control register
0x2C	SPI_MODE	SPI mode
0x30	SYS_STATE	System state
0x40	FCMD_STAT	Fast command status
0x4C	CTR_DBG	Current value of the low 32-bits of the STS IV
0x50	SPICRCINIT	SPI CRC LFSR initialisation code

8.2.14.1 Sub-register 0x0F:00 – Event counter control

ID	Length (octets)	Type	Mnemonic	Description
0F:00	1	SRW	EVC_CTRL	Event counter control

Register file: 0x0F – Digital diagnostics interface, sub-register 0x00 is the event counter control register.

REG:0F:00 – EVC_CTRL – Event counter control																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	-	-	-	EVC_CLR	EVC_EN
																								-	-	-	-	-	-	0	0

Fields in the EVC_CTRL register are intended to be self-clearing. So, the event counters can be enabled or cleared, but cannot be disabled. The bits of the EVC_CTRL register identified above are individually described below:

Field	Description of fields within Sub-register 0x0F:00 – Event counter control
EVC_EN reg:0F:00 bit:0	Event Counters Enable. The EVC_EN bit acts to enable the event counters. When EVC_EN bit is zero none of the event counters will update. When EVC_EN bit is set to 1 it enables event counting. A number of Sub-registers of Register file: 0x0F – Digital diagnostics interface, contain counters of various system events – see below for the detailed description of the parameters counted. If the host system has no interest in these event counters then a small amount of power is saved by not enabling event counting.
EVC_CLR reg:0F:00 bit:1	Event Counters Clear. The EVC_CLR bit acts to clear event counters to zero. This cannot be done while EVC_EN bit is set. The correct procedure to clear the event counters is to write 0x02 to EVC_CTRL to disable counting and clear the counter values to zero, and then to write 0x01 to EVC_CTRL to re-enable counting if required.
- reg:0F:00 bits:7–2	The remaining bits of EVC_CTRL are reserved and should always be set to zero to avoid any malfunction of the device.

8.2.14.2 Sub-register 0x0F:04 – PHR error counter

ID	Length (octets)	Type	Mnemonic	Description
0F:04	2	RO	EVC_PHE	PHR error event counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x04 is the PHY Header Error event counter.

REG:0F:04 – EVC_PHE – PHR Error Counter																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	EVC_PHE											
																-	-	-	-	0											

The fields of the EVC_PHE register are described below:

Field	Description of fields within Sub-register 0x0F:04 – PHR error counter
EVC_PHE reg:0F:04 bits:11–0	PHR Error Event Counter. The EVC_PHE field is a 12-bit counter of PHY Header Errors. This counts the reporting of RXPHE error events in *Sub-register 0x00:44 – System event status*. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.
- bits:15–12	The remaining bits of this register are reserved.

8.2.14.3 Sub-register 0x0F:06 – RSD error counter

ID	Length (octets)	Type	Mnemonic	Description
0F:06	2	RO	EVC_RSE	RSD error event counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x06 is the RSD error event counter.

REG0F:06 – EVC_RSE – RSD error counter																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	EVC_RSE											
																-	-	-	-	0											

The fields of the EVC_RSE register are described below:

Field	Description of fields within Sub-register 0x0F:06 – RSD error counter
EVC_RSE reg:0F:06 bits:11–0	Reed Solomon decoder (Sync Loss) Error Event Counter. The EVC_RSE field is a 12-bit counter of the non-correctable error events that can occur during Reed Solomon decoding. This counts the reporting of RXFSL error events in *Sub-register 0x00:44 – System event status*. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.
- bits:15–12	The remaining bits of this register are reserved.

8.2.14.4 Sub-register 0x0F:08 – FCS good counter

ID	Length (octets)	Type	Mnemonic	Description
0F:08	2	RO	EVC_FCG	Frame Check Sequence good event counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x08 is the FCS good event counter.

REG:0F:08 – EVC_FCG – Frame Check Sequence good event counter																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	EVC_FCG											
																-	-	-	-	0											

The bits of the EVC_FCG register are described below:

Field	Description of fields within Sub-register 0x0F:08 – FCS good counter
EVC_FCG reg:0F:08 bits:11–0	Frame Check Sequence Good Event Counter. The EVC_FCG field is a 12-bit counter of the frames received with good CRC/FCS sequence. This counts the reporting of RXFCG events in *Sub-register 0x00:44 – System event status*. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.
- bits:15–12	The remaining bits of this register are reserved.

8.2.14.5 Sub-register 0x0F:0A – FCS error counter

ID	Length (octets)	Type	Mnemonic	Description
0F:0A	2	RO	EVC_FCE	Frame Check Sequence error counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x0A is the FCS error event counter.

REG:0F:0A – EVC_FCE – FCS error counter																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	EVC_FCE											
																-	-	-	-	0											

The bits of the EVC_FCE register are described below:

Field	Description of fields within Sub-register 0x0F:0A – FCS error counter
EVC_FCE reg:0F:0A bits:11–0	Frame Check Sequence Error Event Counter. The EVC_FCE field is a 12-bit counter of the frames received with bad CRC/FCS sequence. This counts the reporting of RXFCE events in *Sub-register 0x00:44 – System event status*. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.
- bits:15–12	The remaining bits of this register are reserved.

8.2.14.6 Sub-register 0x0F:0C – Frame filter rejection counter

ID	Length (octets)	Type	Mnemonic	Description
0F:0C	1	RO	EVC_FFR	Frame filter rejection counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x0C is the frame filter rejection counter.

REG:0F:0C – EVC_FFR – Frame filter rejection counter																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								EVC_FFR							
																								0							

The bits of the EVC_FFR register are described below:

Field	Description of fields within Sub-register 0x0F:0C – Frame filter rejection counter
EVC_FFR reg:0F:0C bits:7–0	Frame Filter Rejection Event Counter. The EVC_FFR field is an 8-bit counter of the frames rejected by the receive frame filtering function. This is essentially a count of the reporting of ARFE events in *Sub-register 0x00:44 – System event status*. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.

8.2.14.7 Sub-register 0x0F:0E – RX overrun error counter

ID	Length (octets)	Type	Mnemonic	Description
0F:0E	1	RO	EVC_OVR	RX overrun error counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x0E is the RX overrun error counter.

REG:0F:0E – EVC_OVR – RX overrun error counter																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								EVC_OVR							
																								0							

The bits of the EVC_OVR register are described below:

Field	Description of fields within Sub-register 0x0F:0E – RX overrun error counter
EVC_OVR reg:0F:0E bits:7–0	RX Overrun Error Event Counter. The EVC_OVR field is a 12-bit counter of receive overrun events. This is essentially a count of the reporting of RXOVRR events in *Sub-register 0x00:44 – System event status*. The EVC_OVR will be incremented once for each RX frame discarded that happens while an overrun condition persists. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.

8.2.14.8 Sub-register 0x0F:10 – SFD timeout error counter

ID	Length (octets)	Type	Mnemonic	Description
0F:10	2	RO	EVC_STO	SFD timeout error counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x10 is the SFD timeout error counter.

REG:0F:10 – EVC_STO – SFD Timeout Error Counter																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	EVC_STO											
																-	-	-	-	0											

The bits of the EVC_STO register are described below:

Field	Description of fields within Sub-register 0x0F:10 – timeout error counter
EVC_STO reg:0F:10 bits:11–0	SFD timeout errors Event Counter. The EVC_STO field is a 12-bit counter of SFD Timeout Error events. This is essentially a count of the reporting of RXSTO events in *Sub-register 0x00:44 – System event status*. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.
- bits:15–12	The remaining bits of this register are reserved.

8.2.14.9 Sub-register 0x0F:12 – Preamble detection timeout event counter

ID	Length (octets)	Type	Mnemonic	Description
0F:12	2	RO	EVC_PTO	Preamble detection timeout event counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x12 is the preamble timeout event counter.

REG:0F:12 – EVC_PTO – Preamble detection timeout event counter

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	EVC_PTO											
																-	-	-	-	0											

The bits of the EVC_PTO register are described below:

Field	Description of fields within Sub-register 0x0F:12 – Preamble detection timeout event counter
EVC_PTO reg:0F:12 bits:11–0	Preamble Detection Timeout Event Counter. The EVC_PTO field is a 12-bit counter of Preamble detection Timeout events. This is essentially a count of the reporting of RXPTO events, in *Sub-register 0x00:44 – System event status*. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.
- bits:15–12	The remaining bits of this register are reserved.

8.2.14.10 Sub-register 0x0F:14 – RX frame wait timeout event counter

ID	Length (octets)	Type	Mnemonic	Description
0F:14	1	RO	EVC_FWTO	RX frame wait timeout counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x14 is the RX frame wait timeout event counter.

REG:0F:14 – EVC_FWTO – RX frame wait timeout event counter																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								EVC_FWTO							
																								0							

The bits of the EVC_FWTO register are described below:

Field	Description of fields within Sub-register 0x0F:14 – RX frame wait timeout event counter
EVC_FWTO reg:0F:14 bits:7–0	RX Frame Wait Timeout Event Counter. The EVC_FWTO field is an 8-bit counter of receive frame wait timeout events. This is essentially a count of the reporting of the RXFTO events in *Sub-register 0x00:44 – System event status*. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.

8.2.14.11 Sub-register 0x0F:16 – TX frame sent counter

ID	Length (octets)	Type	Mnemonic	Description
0F:16	2	RO	EVC_TXFS	TX frame sent counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x16 is the TX frame sent counter.

REG:0F:16 – EVC_TXFS – TX frame sent counter																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	EVC_TXFS											
																-	-	-	-	0											

The bits of the EVC_TXFS register are described below:

Field	Description of fields within Sub-register 0x0F:16 – TX frame sent counter
EVC_TXFS reg:0F:16 bits:11–0	TX Frame Sent Event Counter. The EVC_TXFS field is a 12-bit counter of transmit frames sent. This is incremented every time a frame is sent. It is essentially a count of the reporting of the TXFRS events in *Sub-register 0x00:44 – System event status*. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.
- bits:15–12	The remaining bits of this register are reserved.

8.2.14.12 Sub-register 0x0F:18 – Half period warning counter

ID	Length (octets)	Type	Mnemonic	Description
0F:18	1	RO	EVC_HPW	Half period warning counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x18 is the half period warning counter.

REG:0F:18 – EVC_HPW – Half period warning counter																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								EVC_HPW							
																								0							

The bits of the EVC_HPW register are described below:

Field	Description of fields within Sub-register 0x0F:18 – Half period warning counter
EVC_HPW reg:0F:18 bits:7–0	Half Period Warning Event Counter. The EVC_HPW field is an 8-bit counter of "Half Period Warnings". This is a count of the reporting of the HPDWARN events in *Sub-register 0x00:44 – System event status*. These relate to late invocation of delayed transmission or reception functionality. Please refer to the description of the HPDWARN bit for more details of this event and its meaning. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in Sub-register 0x0F:00 – Event counter control.

8.2.14.13 Sub-register 0x0F:1A – SPI write CRC error counter

ID	Length (octets)	Type	Mnemonic	Description
0F:1A	1	RO		SPI write CRC error counter

Register file: 0x0F – Digital diagnostics interface, sub-register 0x1A is the SPI write CRC error counter.

REG:0F:1A – EVC_SWCE – SPI write CRC error																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								EVC_SWCE							
																								0							

The bits of the register are described below:

Field	Description of fields within Sub-register 0x0F:1A – SPI write CRC error counter
EVC_SWCE reg:0F:1A bits:7–0	SPI Write CRC Error Counter. The EVC_SWCE is an 8-bit counter of "SPI Write CRC Error" events. This is a count of the reporting of the SPICRCE events in *Sub-register 0x00:44 – System event status*. These events only arise when SPI CRC mode is enabled. See section *1.1.1 –* for more details. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL

8.2.14.14 Sub-register 0x0F:1C – Preamble rejection counter

ID	Length (octets)	Type	Mnemonic	Description
0F:1C	1	RO	EVC_PREJ	Preamble rejection counter

Register file: 0x0F – Digital diagnostics interface, Sub-register 0x1C is the Preamble rejection counter.

REG:0F:1C – EVC_PREJ – Preamble rejection counter																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								EVC_PREJ							
																								0							

The bits of the EVC_PREJ register are described below:

Field	Description of fields within Sub-register 0x0F:20 – STS Quality Error Counter
EVC_PREJ reg:0F:1C bits:7–0	Preamble rejection Counter. The EVC_PREJ field is an 8-bit counter of preamble rejection As a part of preamble detection, there is preamble confirmation step. If we have preamble detection but no preamble confirmation we'll get preamble rejection. After preamble rejection the receiver wil go back into preamble hunt mode and continue looking for preamble. When preamble is confirmed and the receiver continues receiving the preamble and looking for SFD. This counter can increment multiple times as part of preamble detection.

8.2.14.15 Sub-register 0x0F:1D – SFD detects counter

ID	Length (octets)	Type	Mnemonic	Description
0F:1D	2	RO	COUNT_RXSFDD	SFD detects counter

Register file: 0x0F – Digital Diagnostics Interface, sub-register 0x1D is the SFD detects counter.

REG:0F:1D – COUNT_RXSFDD– SFD detects counter																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																				COUNT_RXSFDD											
																				0											

The bits of the COUNT_RXSFDD register are described below:

Field	Description of fields within Sub-register 0x0F:22 – Low voltage warning Error Counter
COUNT_RXSFDD reg:0F:1D bits:11–0	The COUNT_RXSFDD field is a 12-bit counter of SFD detects. As part of packet reception, this will increment each time when SFD is detected.

8.2.14.16 Sub-register 0x0F:20 – STS quality error counter

ID	Length (octets)	Type	Mnemonic	Description
0F:20	1	RO	EVC_CPQE	STS quality error counter

Register file: 0x0F – Digital Diagnostics Interface, sub-register 0x20 is the STS quality error counter.

REG:0F:20 – EVC_CPQE – STS quality error counter																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								EVC_CPQE							
																								0							

The bits of the EVC_CPQE register are described below:

Field	Description of fields within Sub-register 0x0F:20 – STS Quality Error Counter
EVC_CPQE reg:0F:20 bits:7–0	STS Quality Error Counter. The EVC_CPQE field is an 8-bit counter of receive frames for which the STS quality assessment measurements are below the threshold. This is a count of the reporting of received frames with STS where the CIA algorithm has detected a quality problem and set one of the error flags of the CP_TOAST field within *Sub-register 0x0C:08 – STS Receive Time Stamp and Status*. The CPERR count can increment by 2 per frame (two STS blocks in PDOA mode 3 configuration), thus potentially halving the count range. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.

8.2.14.17 Sub-register 0x0F:22 – Low voltage warning error counter

ID	Length (octets)	Type	Mnemonic	Description
0F:22	1	RO	EVC_VWARN	Low voltage warning error counter

Register file: 0x0F – Digital Diagnostics Interface, sub-register 0x22 is the low voltage warning error counter.

REG:0F:22 – EVC_VWARN – Low voltage warning error counter																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								EVC_VWARN							
																								0							

The bits of the register are described below:

Field	Description of fields within Sub-register 0x0F:22 – Low voltage warning Error Counter
EVC_VWARN reg:0F:22 bits:7–0	Low voltage warning Error Counter. The EVC_VWARN field is an 8-bit counter of brown-out warnings. This is a count of the occurrence of low voltage warnings, as reported through the VWARN events status flag in *Sub-register 0x00:44 – System Event Status*. This counts individual brown-out events detected even when the VWARN event flag is not being cleared by the host. NB: For this counter to be active, counting needs to be enabled by the setting the EVC_EN bit in EVC_CTRL.

8.2.14.18 Sub-register 0x0F:28 – Digital diagnostics test mode control

ID	Length (octets)	Type	Mnemonic	Description
0F:28	4	RW	DIAG_TMC	Test mode control register

Register file: 0x0F – Digital diagnostics interface, sub-register 0x24 is the test mode control register.

REG:0F:28 – DIAG_TMC – Test mode control register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	FIXED_ST	-	-	CIA_RUN	-	CIA_WDE	-	-	HIRQ_PO	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TX_PSTM	-	-	-	-
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The bits of the DIAG_TMC register identified above are individually described below:

Field	Description of fields within Sub-register 0x0F:28 – Digital diagnostics test mode control
- reg:0F:28 bit: various	These bits of the DIAG_TMC register are reserved and should always be set to zero to avoid any malfunction of the device.
TX_PSTM reg:0F:28 bit:4	Transmit Power Spectrum Test Mode. This test mode is provided to help support regulatory approvals spectral testing. When the TX_PSTM bit is set it enables a repeating transmission of the data from the TX_BUFFER. To use this test mode, the operating channel, preamble code, data length, offset, etc. should all be set-up as if for a normal transmission. The start-to-start delay between packets is programmed in the DX_TIME register. This is a special use of that register, and the value is programmed in periods of one half of the 499.2 MHz fundamental frequency, (~ 4 ns). To send one packets per millisecond, a value of 249600 or 0x0003CF00 should be programmed into the DX_TIME register. The minimum valid value is to be programmed to DX_TIME register is 2. A time value less than the packet's duration will cause packets to be sent back-to-back. When the mode, the delay and TX buffer have been configured and the TX_PSTM bit is set, the repeated TX mode is initiated by issuing a TX start command, CMD_TX. For full use of this feature please see API and example code provided [3]. Other register settings are needed to make this operational.
HIRQ_POL reg:0F:28 bit:21	Host interrupt polarity. This bit allows the system integrator the ability to control the polarity of the IRQ line from the QM33100. When HIRQ_POL is 1 the IRQ output line from the QM33100 is active high, and, when HIRQ_POL is 0 the IRQ output line from the QM33100 is active low. Active high operation is recommended for low power applications so that the interrupt is in its 0 V logical inactive state when the QM33100 is in **SLEEP** or **DEEPSLEEP** states.

Field	Description of fields within Sub-register 0x0F:28 – Digital diagnostics test mode control
CIA_WDEN reg:0F:28 bit:24	Enable the CIA watchdog. When this configuration is 1 (the default) an internal watchdog timer is started whenever the CIA begins the processing a received CIR, for either the preamble or STS sequences. The watchdog time is fixed at 120 µs. If the CIA completes before the watchdog timer elapses the watchdog timer is stopped, otherwise the event status flag is asserted and the CIA is stopped. This avoids any possibility of run-away processing in the CIA block. If CIA_WDEN is set to 0, the CIA watchdog will be disabled. This might be tried if the events occur to see if good RX timestamp results can be achieved if the CIA was given more time. If the CIA watchdog is disabled the host should include its own watchdog timeout to recover in the event that the CIA takes too long, i.e. the CIADONE event status flag is not arriving.
CIA_RUN reg:0F:28 bit:26	Run the CIA manually. Normally this control bit will not be required because by default the CIARUNE configuration in *Sub-register 0x11:08 – Sequencing control* is set to 1 which causes the CIA to be run automatically when a packet is received. If CIARUNE is 0, then the CIA_RUN bit, may be used to run the CIA after the packet is received. The CIA_IPATOV and CIA_STS bits in *Sub-register 0x00:10 – System configuration* should be set to select which CIA analysis is required. The CIA_RUN bit will automatically clear when it is acted upon. NB: To run the CIA manually, after a receive event when the receiver is off, the receive clock will need to be forced on using the RX_CLK control in *Sub-register 0x11:04 – Clock control*.
FIXED_STS reg:0F:28 bit:29	Fixed STS (cipher) is generated by applying the restart from last on every packet. This ensures that the same IV is applied to the PRNG for each TX or RX packet.

8.2.14.19 Sub-register 0x0F:2C – SPI mode configuration

ID	Length (octets)	Type	Mnemonic	Description
0F:2C	1	RO	SPI_MODE	SPI mode

Register file: 0x0F – Digital diagnostics interface, sub-register 0x2C is the SPI mode configuration.

REG:0F:2C – SPI_MODE – SPI mode configuration

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	-	-	-	SPI_MODE	
																								0	0	0	0	0	0	0	0

The bits of the SPI_MODE register are described below:

Field	Description of fields within Sub-register 0x0F:2C – SPI mode configuration
SPI_MODE reg:0F:2C bits:1–0	SPI Mode. These two bits can be used to read the SPI mode of the device: SPI Clock Phase and Polarity Bit [0] = SPI CLK polarity Bit [1] = SPI CLK phase
- bits:7–2	The remaining bits of this register are reserved.

8.2.14.20 Sub-register 0x0F:30 – System state

ID	Length (octets)	Type	Mnemonic	Description
0F:30	4	RO	SYS_STATE	System states

Register file: 0x0F – Digital diagnostics interface, sub-register 0x30 is the system states diagnosic register.

REG:0F:30 – SYS_STATE – System state																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	TSE_STATE					-	-	RX_STATE						-	-	-	-	TX_STATE			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The bits of the SYS_STATE register are described below:

Field	Description of fields within Sub-register 0x0F:30 – System state	
TX_STATE reg:0F:30 bit: 3-0	<u>Current Transmit State Machine value:</u> 0x0 - IDLE 0x1 - PREAMBLE 0x2 - SFD 0x3 - PHR 0x4 - SDE 0x5 - DATA	<u>Description</u> Transmitter is IDLE Transmitting preamble, Transmitting SFD Transmitting PHY Header data Tranmitting PHR parity SECDED bits Transmitting data block (330 symbols)
reg:0F:30 bit: 7-4	Reserved	

Field	Description of fields within Sub-register 0x0F:30 – System state	
RX_STATE reg:0F:30 bit: 13-8	Current Receive State Machine value 0x00 - IDLE 0x01 - START_ANALOG. 0x04 - RX_RDY 0x05 - PREAMBLE_FND 0x06 - PRMBL_TIMEOUT 0x07 - SFD_FND 0x08 - CNFG_PHR_RX 0x09 - PHR_RX_STRT 0x0A - DATA_RATE_RDY 0x0C - DATA_RX_SEQ 0x0D - CNFG_DATA_RX 0x0E - PHR_NOT_OK 0x0F - LAST_SYMBOL 0x10 - WAIT_RSD_DONE 0x11 - RSD_OK 0x12 - RSD_NOT_OK	Description Receiver is in idle Start analog receiver blocks Receiver ready Receiver is waiting to detect preamble Preamble timeout SFD found Configure for PHR reception PHR reception started Ready for data reception Data reception Configure for data PHR error Received last symbol Wait for Reed Solomon decoder to finish Reed Solomon correct Reed Solomon error
PMSC_STATE reg:0F:30 bit: 20:16	Current PMSC_State Machine value 0x0 - WAKEUP 0x1, 0x2 – IDLE_RC 0x3 – IDLE 0x8 – 0xF – TX 0x12 – 0x19 – RX	Description QM33100 is in WAKEUP QM33100 is in IDLE_RC QM33100 is in IDLE QM33100 is in TX state QM33100 is in RX state
reg:0F:30 bit: various	Reserved	

8.2.14.21 Sub-register 0x0F:40 – FCMD status

ID	Length (octets)	Type	Mnemonic	Description
0F:40	1	RO	FCMD_STAT	Fast command status

Register file: 0x0F – Digital diagnostics interface, sub-register 0x40 is the fast command status.

REG:0F:40 – FCMD_STAT– Fast command status register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	FCMD_STAT				
																								0	0	0	0	0	0	0	0

The bits of the FCMD_STAT register are described below:

Field	Description of fields within Sub-register 0x0F:40 – FCMD status
FCMD_STAT reg:0F:40 bits:4–0	Fast command status. It stores the value of the currently executing fast command. For more information on fast commands see section 9
- bits:7–5	The remaining bits of this register are reserved.

8.2.14.22 Sub-register 0x0F:4C – Counter debug

ID	Length (octets)	Type	Mnemonic	Description
0F:48	4	RO	CTR_DBG	Current value of the low 32-bits of the STS IV

Register file: 0x0F – Digital diagnostics interface, sub-register 0x4C is the counter debug register. It contains the current value of the low 32-bits of STS IV.

8.2.14.23 Sub-register 0x0F:50 – SPI CRC Initialisation

ID	Length (octets)	Type	Mnemonic	Description
0F:50	1	RO	SPICRCINIT	SPI CRC LFSR initialisation code.

Register file: 0x0F – Digital diagnostics interface, sub-register 0x50 is the register that contains SPI CRC LFSR initialisation code for the SPI CRC function. The value here is used to initialise the SPI CRC calculation shift register at the start of each SPI transaction, when SPI CRC mode is enabled via the SPI_CRCEN bit in *Sub-register 0x00:10 – System configuration*. For more details of SPI CRC operation please refer to § *1.1.1 –* , and § 1.1.1 for details of the CRC generation polynomial and shift register implementation.

The SPICRCINIT register contains the following sub-fields:

REG:0F :50 – SPICRCINIT – SPI CRC initialisation code register																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								SPICRCINIT							
																								0	0	0	0	0	0	0	0

The bits of the SPICRCINIT register are described below:

Field	Description of fields within Sub-register 0x0F:40 – FCMD status
SPICRCINIT reg:00:18 bits:7–0	SPI CRC LFSR initialisation code for the SPI CRC function.
- bits:31–0	The remaining bits of this register are reserved.

8.2.15 Register file: 0x11 – PMSC control and status

ID	Length (octets)	Type	Mnemonic	Description
0x11	75	-	PMSC_CTRL	Power management, timing and seq control

Register map register file 0x11 is concerned with the use of the PMSC. It contains a number of Sub-registers. An overview of these is given by Table 20. Each of these Sub-registers is separately described in the sub-sections below.

Table 43: Register file: 0x11 – PMSC control and status overview

OFFSET in Register 0x11	Mnemonic	Description
00	SOFT_RST	Soft reset of the device blocks
04	CLK_CTRL	PMSC clock control register
08	SEQ_CTRL	PMSC control register 1
10	TXFSEQ	PMSC fine grain TX sequencing control
18	LED_CTRL	LED control register
1C	RX_SNIFF	Sniff mode configuration
30	BIAS_CTRL	Analog blocks' calibration values
3C	TIMER_CTRL	Timer control register
40	TIMER0_CNT_SET	Timer 0 count / set register
44	TIMER1_CNT_SET	Timer 1 count / set register
48	TIMER_STATUS	Timer status register

8.2.15.1 Sub-register 0x11:00 – Soft reset

ID	Length (octets)	Type	Mnemonic	Description
11:00	2	RW	SOFT_RST	Soft reset of the device blocks

Register file: 0x11 – PMSC control and status, Sub-register 0x00 is the soft reset command register. This register allows a software applied reset to be applied to the IC.

REG:11:00 – SOFT_RST– Soft reset of the device blocks																												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	-	-	-	DIGAON_RST_N	TIM_RST	-	GPIO_RST	PMSC_RST	HIF_RST	TX_RST	RX_RST	BIST_RST	CIA_RST	PRGN_RST	ARM_RST
																							SOFTRESET								
																-	-	-	-	1	1	1	1	1	1	1	1	1	1	1	1

The bits of the SOFT_RST register are described below:

Field	Description of fields within Sub-register 0x11:00 – Soft reset
SOFTRESET reg:11:00 bits:8–0	These nine bits reset various the IC blocks, e.g. GPIO, TX, RX, Host Interface and the PMSC itself, essentially allowing a reset of the IC under software control. These bits should be cleared to zero to force a reset and then returned to one for normal operation. The correct procedure to achieve this reset is to: (a) Set SYS_CLK to 01 (b) Clear SOFTRESET to all zero's (c) Set SOFTRESET to all ones The AON block is not reset by this activity and so may take action following the reset depending on the configuration within *Sub-register 0x0A:00 – AON on* wake configuration.
TIM_RST reg:11:00 bits:10	Timer block soft reset. Setting the active-low TIM_RST bit to zero initiates a reset of the timer block, which resets the state of both Timer 0 and Timer 1. The TIM_RST bit automatically returns to its default value of 1. Unless other resets are required the other bits in the SOFT_RST register should be left at 1 when setting TIM_RST bit to zero to effect this reset. See also the discussion in 7.6.1 - Dual-timer block – reset and soft reset
DIGAON_RST_N reg:11:00 bits:11	This will perform a power cycle and reset of the main digital core.
- bits:15–12	The remaining bits of this register are reserved.

8.2.15.2 Sub-register 0x11:04 – Clock control

ID	Length (octets)	Type	Mnemonic	Description
11:04	4	RW	CLK_CTRL	PMSC clock control register

Register file: 0x11 – PMSC control and status, Sub-register 0x00 is a 32-bit control register relating to enabling clocking to various digital blocks within the QM33100. The CLK_CTRL register contains the following sub-fields:

REG:11:04 – CLK_CTRL – PMSC clock control register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-				TIM_RST_MODE	-	-	-	LP_CLK_EN	-	-	-	GPIO_DRST_N	GPIO_DCLK_EN	-	GPIO_CLK_EN	ACC_MCLK_EN	-	-	TX_BUF_CLK_ON	RX_BUF_CLK_ON	SAR_CLK_EN	OTP_CLK_EN	CIA_CLK_EN	-	ACC_CLK_EN	TX_CLK		RX_CLK		SYS_CLK	
1	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

The fields of the CLK_CTRL register identified above are individually described below:

Field	Description of fields within Sub-register 0x11:04 – Clock control
-	Bits marked '-' are reserved and should be preserved at their reset value.
SYS_CLK reg:11:04 bits:1,0	System Clock Selection field. The SYS_CLK field selects the source of clock for QM33100 system clock. Allowed values are: 00: Auto – The system clock will run off the FAST_RC/4 clock until (assuming AINIT2IDLE is set to 1) the AON transfer has completed, it will then switch to FAST_RC(120MHz) clock until the PLL is calibrated and locked, and then it will switch to the 125 MHz PLL clock. 01: Force system clock to be the FAST_RC/4 clock. 10: Force system clock to the 125 MHz PLL clock. (If this clock is not present the IC will essentially lock up with further SPI communications impossible. In this case an external reset will be needed to recover). 11: Force system clock to FAST_RC. This control is used for certain procedures, e.g. before a soft reset (SOFTRESET)
RX_CLK reg:11:04 bits:3,2	Receiver Clock Selection. This selects the source of clock for the QM33100 receiver. Allowed values are: 00: Auto – The RX clock will be disabled until it is required for an RX operation, at which time it will be enabled to use the 125 MHz PLL clock. 01, 10, 11: Force RX clock enable and sourced from the 125 MHz PLL clock. (NB: ensure PLL clock is present). This RXCLKS control is used for certain procedures.

Field	Description of fields within Sub-register 0x11:04 – Clock control
TX_CLK reg:11:04 bits:5,4	Transmitter Clock Selection. This selects the source of clock for the QM33100 transmitter. Allowed values are: 00: Auto – The TX clock will be disabled until it is required for a TX operation, at which time it will be enabled to use the 125 MHz PLL clock. 01, 10, 11: Force TX clock enable and sourced from the 125 MHz PLL clock. (NB: ensure PLL clock is present). This control is used for certain procedures, e.g. when setting up the continuous transmission mode that is used during power output calibration and regulatory testing.
ACC_CLK_EN reg:11:04 bit:6	Force Accumulator Clock Enable. In normal operation this bit should be set to 0 to allow the PMSC to control the accumulator clock as necessary for normal receiver operation. If the host system wants to read the accumulator data, both this ACC_CLK_EN bit and the ACC_MCLK_EN bit (below) need to be set to 1 to allow the accumulator reading to operate correctly.
CIA_CLK_EN reg:11:04 bit:8	Force CIA Clock Enable. In normal operation this bit should be set to 0 to allow the PMSC to control the CIA clock as necessary for normal CIA operation. If the host system wants to run CIA manually using CIA_RUN then this bit should be set to 1.
OTP_CLK_EN reg:11:04 bit:9	Force on the NVM, AON interface clocks.
SAR_CLK_EN reg:11:04 bit:10	(temperature and voltage) Analog-to-Digital Convertor Clock Enable. The QM33100 is equipped with 8-bit A/D convertors to sample the IC temperature and its input battery voltage. The IC can automatically sample the temperature and voltage as it wakes up from **SLEEP** or **DEEPSLEEP**. This is controlled by the ONW_RUN_SAR bit in *Sub-register 0x0A:00 – AON on* wake configuration. If the host system wants to initiate temperature and/or voltage measurements at other times then the clock to the Analog-to-Digital Convertor needs to be enabled via this SAR_CLK_EN bit. For more details of this functionality, please refer to section *7.4 – Measuring IC temperature and voltage.*
RX_BUF_CLK_ON reg:11:04 bit:11	Force the RX buffer clock on.
TX_BUF_CLK_ON reg:11:04 bit:12	Force the TX buffer clock on.
ACC_MCLK_EN reg:11:04 bit:15	Accumulator Memory Clock Enable. In normal operation this bit should be set to 0 to allow the PMSC to control the accumulator memory clock as necessary for normal receiver operation. If the host system wants to read the accumulator data, both this ACC_MCLK_EN bit and ACC_CLK_EN bit (above) need to be set to 1 to allow the accumulator reading to operate correctly.
GPIO_CLK_EN reg:11:04 bit:16	GPIO clock Enable. In order to use the GPIO port lines the GPIO_CLK_EN enable must be set to 1 to enable the clock into the GPIO block. The bit must also be set to 1 to take the GPIO port out of its reset state.

Field	Description of fields within Sub-register 0x11:04 – Clock control
GPIO_DCLK_EN reg:11:04 bit:18	GPIO De-bounce Clock Enable. The QM33100 GPIO port includes a de-bounce functionality that may be applied to input lines being used as an interrupt source. The de-bounce filter circuit clocks the GPIO inputs into the QM33100 and removes short transients by requiring that the input persists for two cycles of this clock before it will be seen by the interrupt handling logic. (See *Sub-register 0x05:28 – GPIO interrupt* de-bounce enable for more details). In order to use the GPIO port de-bounce functionality this GPIO_DCLK_EN bit must be set to 1 to enable the clock into the GPIO block. The GPIO_DRST_N bit (below) must also be set to 1 to take the GPIO port de-bounce filter circuit out of its reset state. This GPIO_DCLK_EN bit also serves to enable the clock that controls the LED blink functionality and so must be enabled in order for the LEDs to function correctly. See *Sub-register 0x05:00 – GPIO mode* control for details of enabling LED functionality on GPIO lines. Note: As this clock employs the kilohertz clock, the appropriate dividers and enables need to be configured according to the desired functionality. See LP_CLK_EN below and LP_CLK_DIV in *Sub-register 0x11:08 – Sequencing control*.
GPIO_DRST_N reg:11:04 bit:19	GPIO de-bounce reset (NOT), active low. In order to use the GPIO port de-bounce filter circuit the GPIO_DRST_N bit must be set to 1 to take the de-bounce filter circuit out of its reset state. The GPIO_DCLK_EN enable bit (above) must also be set to 1 to enable the clock into the GPIO de-bounce circuit.
LP_CLK_EN reg:11:04 bit:23	Kilohertz clock Enable. When this bit is set to 1 it enables the divider. The divider value is set by LP_CLK_DIV in *Sub-register 0x11:08 – Sequencing control*.
TIM_RST_MODE reg:11:04 bit:27	Timer reset mode configuration. The TIM_RST_MODE bit controls what happens when a general soft reset is applied to the IC, (e.g. via the SRESET bit in the SPI_SEM register). When TIM_RST_MODE is set to 1, the soft reset will reset the timer block. When TIM_RST_MODE = 0, (the default value), the soft reset does not reset the timer block. The TIM_RST bit in the SOFT_RST register allows separate reset of the timer block whenever this is needed. See also the discussion in 7.6.1 - Dual-timer block – reset and soft reset
-	Bits marked '-' are reserved and should be preserved at their reset value.

8.2.15.3 Sub-register 0x11:08 – Sequencing control

ID	Length (octets)	Type	Mnemonic	Description
11:08	4	RW	SEQ_CTRL	PMSC sequencing control register

Register file: 0x11 – PMSC control and status, Sub-register 0x08 is a 32-bit control register. The SEQ_CTRL register contains the following sub-fields:

REG:11:08 – SEQ_CTRL – Sequencing control register																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LP_CLK_DIV						-	-	FORCE2INIT	-	-	-	-	-	CIARUNE	-	PLL_SYNc	-	-	ARX2SLP	ATX2SLP	-	-	AINIT2IDLE	-	-	-	-	-	-	-	-
1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	1	1	1	0	0	0

The fields of the SEQ_CTRL register identified above are individually described below:

Field	Description of fields within Sub-register 0x11:08 – Sequencing control
-	Bits marked '-' are reserved and should be preserved at their reset value.
AINIT2IDLE reg:11:08 bit:8	Automatic **IDLE_RC** to **IDLE_PLL**. The AINIT2IDLE bit is responsible for transitioning the IC from **IDLE_RC** state to **IDLE_PLL** state. If AINIT2IDLE is set to 1 before entering SLEEP then upon wake up the IC will automatically transition into **IDLE_PLL** state as soon as the clock PLL has locked. By default AINIT2IDLE is clear which means that after a reset, (or when coming out of SLEEP), the IC will stay in the **IDLE_RC** state until AINIT2IDLE is set. This may facilitate lower energy use while setting up the IC before transitioning through IDLE state into TX or RX states. See *§ 2.4 – QM33100 operational states* for more details.
ATX2SLP reg:11:08 bit:11	After TX automatically Sleep. If this bit is set then the QM33100 will automatically transition into **SLEEP** or **DEEPSLEEP** state after transmission of a packet has completed so long as there are no unmasked interrupts pending. This bit is cleared when the QM33100 wakes from sleep. Before using this ATX2SLP feature the AON configurations in *Register file: 0x0A – Always-on system control interface* should be set to allow for the appropriate QM33100 wake up functionality. One of the uses for this would be in a device that periodically transmits a message (e.g. TDoA RTLS Tag) to return the QM33100 to its lowest power state immediately after the transmission, saving power. Note: The SLEEP_EN bit in *Sub-register 0x0A:14 – AON* configuration has to be set to enable this functionality.
ARX2SLP reg:11:08 bit:12	After RX automatically Sleep. If this bit is set then the QM33100 will automatically transition into **SLEEP** state after a receive attempt so long as there are no unmasked interrupts pending. Before using ARX2SLP the AON configurations in *Register file: 0x0A – Always-on system control interface* should be set to allow for the appropriate QM33100 wake up functionality. This bit is cleared when the QM33100 wakes from sleep. Note: SLEEP_EN bit in *Sub-register 0x0A:14 – AON* configuration has to be set to enable this functionality.
PLL_SYNC reg:11:08 bit:15	This enables a 1 GHz clock used for some external SYNC modes. If this is not required then to save power the PLL_SYN configuration should be left set to 0. See *Register file: 0x04 – External* sync control and RX calibration for more details.
CIARUNE reg:11:08 bit:17	CIA run enable. This bit enables the running of the CIA algorithm. CIARUNE is 1 by default which means that the CIA algorithm is run automatically. When CIARUNE is set to zero the CIA algorithm will not be run and the RX_STAMP in *Sub-register 0x00:60 – Receive time stamp* will not be updated, however the CIA_RUN bit, may be used to run the CIA after the frame is received. The CIA_IPATOV and CIA_STS bits in *Sub-register 0x00:10 – System configuration* should be set to select which CIA analysis is required.

Field	Description of fields within Sub-register 0x11:08 – Sequencing control
FORCE2INIT reg:11:08 bit:23	Force to **IDLE_RC** state. When device is in **IDLE_PLL**, but host needs to change the sate back to **IDLE_RC**, e.g. it is not actively ranging and needs to save power. The host needs to clear the AINIT2IDLE bit and set this FORCE2INIT bit. Prior to this the host needs to set SYS_CLK to 0x3 to force it to FAST_RC. The device will then transition into the **IDLE_RC** state. The host should finally clear this FORCE2INIT bit back to 0. See *§ 2.4 – QM33100 operational states* for more details
LP_CLK_DIV reg:11:08 bits:31–26	Kilohertz clock divisor. This field specifies a clock divider designed to give a kilohertz range clock that is used in the QM33100 for the LED blink functionality and also for the GPIO de-bounce functionality. The input to the kHz divider is the 19.2 MHz clock (which is the raw 38.4 MHz XTAL ÷ 2). The LP_CLK_DIV field specifies the top 6 bits of a 10-bit counter allowing divisors up to 2016 or clock frequencies from 9.5 kHz up to 600 kHz. The resultant clock is used directly in the GPIO de-bounce circuit (see *Sub-register 0x05:28 – GPIO interrupt* de-bounce enable). A further divider is applied for the LED blink functionality, see *Sub-register 0x11:18 – LED control*.

8.2.15.4 Sub-register 0x11:10 – TX fine control

ID	Length (octets)	Type	Mnemonic	Description
11:10	4	RW	TXFSEQ	PMSC fine grain TX sequencing control

Register file: 0x11 – PMSC control and status, Sub-register 0x10 is used to control TX fine grain power sequencing function. The TXFSEQ register contains the following sub-fields:

REG:11:10 – TXFSEQ – Fine grain TX sequencing control register																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXFINESEQ																															
0	1	0	0	1	1	0	1	0	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	1	1	0	1	0	0

The fields of the TXFSEQ register identified above are described below:

Field	Description of fields within Sub-register 0x11:10 – TX fine control
TXFINESEQ reg:11:10 bits:31–0	Writing 0x0d20874 to this field will disable TX fine grain power sequencing, this is required for certain test and calibration modes (Continuous Wave transmission). To enable fine grain power sequencing the default value of 0x4d28874 should be written back to this register.

8.2.15.5 Sub-register 0x11:18 – LED control

ID	Length (octets)	Type	Mnemonic	Description
11:18	4	RW	LED_CTRL	LED control register

Register file: 0x11 – PMSC control and status, Sub-register 0x18 is a 32-bit LED control register. The LED_CTRL register contains the following sub-fields:

REG:11:18 – LED_CTRL – LED control register																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	FORCE_TRIG				-	-	-	-	-	-	-	BLINK_EN	BLINK_TIM							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

The fields of the LED_CTRL register identified above are individually described below:

Field	Description of fields within Sub-register 0x11:18 – LED control
-	Bits marked '-' are reserved and should be preserved at their reset value.
BLINK_TIM reg:11:18 bits:7–0	Blink time count value. This field determines how long the LEDs remain lit after an event that causes them to be set on. This time is specified in units of 14 ms so the default value of 0x20 will give an on blink of 400 ms followed by an off blink of 400 ms.
BLINK_EN reg:11:18 bit:8	Blink Enable. When this bit is set to 1 the LED blink feature is enabled. Because the LED blink counter uses the low frequency KHZCLK timer, this timer must be enabled as per *Sub-register 0x11:04 – Clock* control and configured as per *Sub-register 0x11:08 – Sequencing control*
FORCE_TRIG reg:11:18 bits:19:16	Manually triggers an LED blink. There is one trigger bit per LED IO.

8.2.15.6 Sub-register 0x11:1C – SNIFF mode

ID	Length (octets)	Type	Mnemonic	Description
0x11:1C	4	RW	RX_SNIFF	Receiver SNIFF mode configuration

Register file: 0x11 – PMSC control and status, Sub-register 0x1C is a 32-bit register used for configuration of SNIFF mode, which is a power saving technique that can be employed to reduce the power consumption of preamble detection. For normal preamble reception the receiver searches for preamble continually, while in SNIFF mode the receiver samples ("sniffs") the air periodically on a timed basis returning to receiver idle mode in between.

The transmitting device needs to be sending a sufficiently long preamble to allow for the SNIFF mode to operate and leave sufficient preamble remaining thereafter to get a good reception and RX timestamp. The power saving is dependent on the configured on/off times for this sampling. See additionally section 4.5 Low-Power SNIFF mode. The configuration consists of the fields identified and described below:

REG:11:1C – RX_SNIFF –SNIFF Mode Configuration																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SNIFF_OFF								-	-	-	-	SNIFF_ON			
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0								0	0	0	0	0			

The individual sub-fields are described below:

Field	Description of fields within Sub-register 0x11:1C – SNIFF mode
-	Bits marked '-' are reserved.
SNIFF_ON reg:11:1C bits:3–0	SNIFF Mode ON time. This parameter is specified in units of PAC. For details of PAC and its role in preamble, please refer to section 4.1.1. A value of zero will disable SNIFF Mode. A non-zero value will enable Preamble Detection Mode and select how long the receiver is turned on during the preamble hunt. NB: This must be a minimum of 2 for the IC to correctly make a preamble detection decision. If preamble is detected during this time window the receiver will remain on and will continue to attempt reception of the packet. If no preamble is detected the receiver will be returned to idle mode for the time configured by the SNIFF_OFF parameter before sampling the air again.
SNIFF_OFF reg:11:1C bits:15–8	SNIFF Mode OFF time specified in µs. This parameter is specified in units of approximately 1 µs, or 128 system clock cycles. A value of zero will disable SNIFF Mode. A non-zero value will enable SNIFF Mode and select how long the receiver is turned off for during the preamble hunt. Please refer to the SNIFF_ON description above for more details of this feature.

As an example, with a 1024 preamble length, a roughly 50% duty cycle (on 50% and off 50%) can be configured with a PAC of 8 symbols, SNIFF_ON set to 3 PAC intervals, and SNIFF_OFF set to 24 microseconds. The performance cost of this in terms of receiver sensitvity is < 1 dB.

8.2.15.7 Sub-register 0x11:30 – Bias control

ID	Length (octets)	Type	Mnemonic	Description
11:30	2	RW	BIAS_CTRL	Analog blocks' calibration values

Register file: 0x11 – PMSC control and status, Sub-register 0x30 is used to configure analog blocks It stores per device calibration values. Each device will have calibration values stored in OTP and these should be written here on power up to ensure optimal device operation. The BIAS_CTRL register contains the following sub-fields:

REG:11:30 – BIAS_CTRL – Analog blocks'calibration values															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	BIAS_CTRL													
0	0	0	0	0	0	1	0	0	1	1	1	0	0	0	0

The fields of the BIAS_CTRL register identified above are described below:

Field	Description of fields within Sub-register 0x11:30 – Bias control
BIAS_CTRL reg:11:30 bits:13–0	This register controls analog blocks. Each device will have optimised values stored in OTP (BIASTUNE_CAL) and that value should be written to this register on power up, and also after SLEEP/DEEPSLEEP. This will ensure the device will perfom optimally.
-	Bits marked '-' are reserved and should be preserved at their reset value.

8.2.15.8 Sub-register 0x11:3C – Timer control register

ID	Length (octets)	Type	Mnemonic	Description
11:3C	4	RW	TIMER_CTRL	Timer control register

Register 11:3C is the timer control register. The TIMER_CTRL register contains the following sub-fields:

REG:11:3C– TIMER_CTRL – Timer control register																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	TIMER1_COEX	TIMER1_GPIO	TIMER1_TRIG	TIMER1_MODE	TIMER1_DIV			-	TIMER0_COEX	TIMER0_GPIO	TIMER0_TRIG	TIMER0_MODE	TIMER0_DIV			-	-	-	-	TIMER1_RDC	TIMER0_RDC	TIMER1_START	TIMER0_START
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0

The fields of the TIMER_CTRL register identified above are individually described below:

Field	Description of fields within Sub-register 0x11:3C – Timer control register
-	Bits marked '-' are reserved and should be preserved at their reset value.
TIMER0_START reg:11:3C bit:0	Timer 0 start control. When the TIMER0_START bit state is changed from zero to one, this transition enables the counting of Timer 0 to start. Setting TIMER0_START to zero does not stop the timer. Timer 0 is stopped by any the following three events: • Issuing as timer reset using the using the timer reset using the TIM_RST • bit in the SOFT_RST register. NB: This will also stop Timer 1. • When the timer count reaches the terminal value programmed in the TIMER0_CNT_SET register and the timer is operating in single shot mode, i.e., TIMER0_MODE = 0. • If the timer is configured, (by TIMER0_GPIO = 1), to stop on the assertion of the GPIOIRQ event status bit the SYS_STATUS register, and GPIOIRQ is asserted. (*Please refer to the IC user manual for details of the GPIO interrupt functionality and programming*). For predictable operation, the clock divider in the TIMER0_DIV field and the terminal count in the TIMER0_CNT field should be configured before Timer 0 is enabled.

<table>
<tr><th>Field</th><th>Description of fields within Sub-register 0x11:3C – Timer control register</th></tr>
<tr><td>TIMER1_START
reg:11:3C
bit:1</td><td>Timer 1 enable control. When the TIMER1_START bit state is changed from zero to one, this transition enables the counting of Timer 1 to start. Setting TIMER1_START to zero does not stop the timer. Timer 1 is stopped by any the following three events:
(a) Issuing as timer reset using the using the timer reset using the TIM_RST
(b) bit in the SOFT_RST register. NB: This will also stop Timer 0.
(c) When the timer count reaches the terminal value programmed in the TIMER1_CNT_SET register and the timer is operating in single shot mode, i.e., TIMER1_MODE = 0.
(d) If the timer is configured, (by TIMER1_GPIO = 1), to stop on the assertion of the GPIOIRQ event status bit the SYS_STATUS register, and GPIOIRQ is asserted. (Please refer to the IC user manual for details of the GPIO interrupt functionality and programming).
For predictable operation, the clock divider in the TIMER1_DIV field and the terminal count in the TIMER1_CNT field should be configured before Timer 1 is enabled.</td></tr>
<tr><td>TIMER0_RDC
reg:11:3C
bit:2</td><td>Timer 0 read control. The TIMER0_RDC bit determines what is returned when reading from the TIMER0_CNT register. When TIMER0_RDC = 0, reading TIMER0_CNT returns the terminal count setting for timer 0. When TIMER0_RDC = 1, reading TIMER0_CNT returns current running counter value for timer 0.</td></tr>
<tr><td>TIMER1_RDC
reg:11:3C
bit:3</td><td>Timer 1 read control. The TIMER1_RDC bit determines what is returned when reading from the TIMER1_CNT register. When TIMER1_RDC = 0, reading TIMER1_CNT returns the terminal count setting for timer 1. When TIMER1_RDC = 1, reading TIMER1_CNT returns current running counter value for timer 1.</td></tr>
<tr><td>TIMER0_DIV
reg:11:3C
bits:10–8</td><td>Timer 0 divider configuration. This field configures the divider to apply to the clock input to Timer 0, which selects the rate at which the timer counts. Allowed values of the TIMER0_DIV field and the resultant clock rates are given in Table 44, (applying for the TIMER1_DIV field also).

Table 44: Timer divider values

<table>
<tr><th>Value programmed</th><th>Divider</th><th>Resultant Clock rate (divided external clock)</th></tr>
<tr><td>000b</td><td>1</td><td>38.4 MHz</td></tr>
<tr><td>001b</td><td>2</td><td>19.2 MHz</td></tr>
<tr><td>010b</td><td>4</td><td>9.6 MHz</td></tr>
<tr><td>011b</td><td>8</td><td>4.8 MHz</td></tr>
<tr><td>100b</td><td>16</td><td>2.4 MHz</td></tr>
<tr><td>101b</td><td>32</td><td>1.2 MHz</td></tr>
<tr><td>110b</td><td>64</td><td>600 kHz</td></tr>
<tr><td>111b</td><td>128</td><td>300 kHz</td></tr>
</table>

The default value for TIMER0_DIV is five giving a divide by 32 so that the timer will count at a 1.2 MHz rate. For predictable operation, the TIMER0_DIV field should be configured before Timer 0 is enabled.</td></tr>
<tr><td>TIMER0_MODE
reg:11:3C
bit:11</td><td>Timer 0 mode selection. The TIMER0_MODE bit controls whether the timer runs as a single shot timer as a repeating interval timer. When TIMER0_MODE is zero this selects single-shot mode. In the single-shot mode when the counting timer reaches the terminal value programmed in the TIMER0_CNT_SET register, the timer stops counting. When TIMER0_MODE is one this selects repeating mode. In the repeating mode when the counting timer reaches the terminal value, the timer count value reverts to zero and counting will continue. In either case the TIMER0 event status bit is set (in the SYS_STATUS register), and the TIMER0_EVC is incremented to count this event.</td></tr>
<tr><td>TIMER0_TRIG
reg:11:3C
bit:12</td><td>Halt timer on interrupt flag</td></tr>
</table>

Field	Description of fields within Sub-register 0x11:3C – Timer control register
TIMER0_GPIO reg:11:3C bit:13	Timer 0 GPIO halt mode configuration. The TIMER0_GPIO bit when set to 1 enables a feature whereby a running Timer 0 will be stopped upon the assertion the GPIOIRQ event status bit (in the SYS_STATUS register). When TIMER0_GPIO is zero, the GPIOIRQ event has no effect on the timer. (*Please refer to the IC user manual for details of the GPIO interrupt functionality and programming*).
TIMER0_COEX reg:11:3C bit:14	Timer 0 coexistence trigger enable. The TIMER0_COEX bit, when set to 1, enables Timer 0 events to pass into the coexistence block, where depending on the configuration of the COEX_OUT_MODE (in the SYS_CFG register) this can cause the assertion of the coexistence output pin. When TIMER0_COEX is zero, timer 0 events do not affect the coexistence block.
TIMER1_DIV reg:11:3C bits:18–16	Timer 1 – Divider. This field configures the divider to apply to the clock input to Timer 1, which selects the rate at which the timer counts. Allowed values of the TIMER1_DIV field and the resultant clock rates are given in Table 44 above, (applying for the TIMER0_DIV field also). The default value for TIMER1_DIV is five giving a divide by 32 so that the timer will count at a 1.2 MHz rate. For predictable operation, the TIMER1_DIV field should be configured before Timer 1 is enabled.
TIMER1_MODE reg:11:3C bit:19	Timer 1 mode selection. The TIMER1_MODE bit controls whether the timer runs as a single shot timer as a repeating interval timer. When TIMER1_MODE is zero this selects single-shot mode. In the single-shot mode when the counting timer reaches the terminal value programmed in the TIMER1_CNT_SET register, the timer stops counting. When TIMER1_MODE is one this selects repeating mode. In the repeating mode when the counting timer reaches the terminal value, the timer count value reverts to zero and counting will continue. In either case the TIMER1 event status bit is set (in the SYS_STATUS register), and the TIMER1_EVC is incremented to count this event.
TIMER1_TRIG reg:11:3C bit:20	Halt timer on interrupt flag
TIMER1_GPIO reg:11:3C bit:21	Timer 1 GPIO halt mode configuration. The TIMER1_GPIO bit when set to 1 enables a feature whereby a running Timer 1 will be stopped upon the assertion the GPIOIRQ event status bit (in the SYS_STATUS register). When TIMER1_GPIO is zero, the GPIOIRQ event has no effect on the timer. (*Please refer to the IC user manual for details of the GPIO interrupt functionality and programming*).
TIMER1_COEX reg:11:3C bit:22	Timer 1 coexistence trigger enable. The TIMER1_COEX bit, when set to 1, enables Timer 1 events to pass into the coexistence block, where depending on the configuration of the COEX_OUT_MODE (in the SYS_CFG register) this can cause the assertion of the coexistence output pin. When TIMER1_COEX is zero, timer 1 events do not affect the coexistence block.

8.2.15.9 Sub-register 0x11:40 – Timer 0 count / set register

ID	Length (octets)	Type	Mnemonic	Description
11:40	4	RW	TIMER0_CNT_SET	Timer 0 count/set register

Register 11:40 is the timer 0 count/set register. The TIMER0_CNT_SET register contains the following sub-fields:

REG:11:40 – TIMER0_CNT_SET – Timer 0 count / set register																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	TIMER0_CNT																					
-	-	-	-	-	-	-	-	-	-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0	0	0

The fields of the TIMER0_CNT_SET register identified above are described below:

Field	Description of fields within Sub-register 0x11:40 – Timer 0 count / set register
-	Bits marked '-' are reserved.
TIMER0_CNT reg:11:40 bits:0–21	Timer 0 count. Writing to the TIMER0_CNT field sets the terminal count for Timer 0. When the time is running and it reaches this count, the TIMER0 event status bit is set the SYS_STATUS register. The result of reading from the TIMER0_CNT field depends on the setting of the TIMER0_RDC bit in the TIMER_CTRL register. When TIMER0_RDC = 0, reading TIMER0_CNT returns the terminal count setting for the timer. When TIMER0_RDC = 1, reading TIMER0_CNT returns current running counter value. The default TIMER0_CNT value is 0x4B0, i.e., 1200 decimal, which with the default TIMER0_DIV divider value setting a 1.2 MHz count rate, gives the timer a 1 ms duration by default. NB: For predictable operation, the required TIMER0_CNT value should be programmed before Timer 0 is enabled.

8.2.15.10 Sub-register 0x11:44 – Timer 1 count / set register

ID	Length (octets)	Type	Mnemonic	Description
11:44	4	RW	TIMER1_CNT_SET	Timer 0 count/set register

Register 11:44 is the timer 1 count/set register. The TIMER1_CNT_SET register contains the following sub-fields:

REG:11:44 – TIMER1_CNT_SET – Timer control register																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	TIMER1_CNT																					
-	-	-	-	-	-	-	-	-	-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0	0	0

The fields of the TIMER1_CNT_SET register identified above are described below:

Field	Description of fields within Sub-register 0x11:44 – Timer 1 count / set register
-	Bits marked '-' are reserved.
TIMER1_CNT reg:11:44 bits:0–21	Timer 1 count. Writing to the TIMER1_CNT field sets the terminal count for Timer 0. When the time is running and it reaches this count, the TIMER1 event status bit is set the SYS_STATUS register. The result of reading from the TIMER1_CNT field depends on the setting of the TIMER1_RDC bit in the TIMER_CTRL register. When TIMER1_RDC = 0, reading TIMER1_CNT returns the terminal count setting for the timer. When TIMER1_RDC = 1, reading TIMER1_CNT returns current running counter value. The default TIMER1_CNT value is 0x4B0, i.e., 1200 decimal, which with the default TIMER1_DIV divider value setting a 1.2 MHz count rate, gives the timer a 1 ms duration by default. NB: For predictable operation, the required TIMER1_CNT value should be programmed before Timer 1 is enabled.

8.2.15.11 Sub-register 0x11:48 – Timer status register

ID	Length (octets)	Type	Mnemonic	Description
11:48	2	RO	TIMER_STATUS	Timer status register

Register 11:48 is the timer status register. The TIMER_STATUS register contains the following sub-fields:

REG:11:48 – TIMER_STATUS – Timer control register																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																TIMER1_EVC								TIMER0_EVC							
																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields of the TIMER_STATUS register identified above are described below:

Field	Description of fields within Sub-register 0x11:48 – Timer status register
TIMER0_EVC reg:11:48 bits:0–7	Timer 0 event counter. The TIMER0_EVC field counts and reports a count of events relating to Timer 0. TIMER0_EVC is incremented when the running Timer 0 reaches the terminal count set in the TIMER0_CNT_SET register. TIMER0_EVC is also incremented if Timer 0 is stopped by the assertion of the GPIOIRQ event when TIMER0_GPIO bit is set to 1 to allow this. The host software should read both TIMER0_EVC and TIMER1_EVC at the same time because the act of reading from this TIMER_STATUS register clears both values (i.e., returning both TIMER0_EVC and TIMER1_EVC to zero).
TIMER1_EVC reg:11:48 bits:15–8	Timer 1 event counter. The TIMER1_EVC field counts and reports a count of events relating to Timer 1. TIMER1_EVC is incremented when the running Timer 1 reaches the terminal count set in the TIMER1_CNT_SET register. TIMER1_EVC is also incremented if Timer 1 is stopped by the assertion of the GPIOIRQ event when TIMER1_GPIO bit is set to 1 to allow this. Reading the TIMER1_EVC also to clear the count (i.e., returning TIMER1_EVC to zero). The host software should read both TIMER0_EVC and TIMER1_EVC at the same time because the act of reading from this TIMER_STATUS register clears both values (i.e., returning both TIMER0_EVC and TIMER1_EVC to zero).

8.2.16 Register file: 0x12 – RX frame buffer 0

ID	Length (octets)	Type	Mnemonic	Description
12	1024	ROD	RX_BUFFER_0	RX frame data buffer 0

Register map register file 0x12:00 is the main receive data buffer. The data from the received frame is available in the received buffer. Assuming successful reception of a good frame, the full length of received data (as reported by the RXFLEN field of REG_RX_FINFO), will be available in the RX_BUFFER_0 beginning at offset 0. Note since the reported length includes the FCS the host system will probably choose not to read these final two octets.

Write operations to the RX_BUFFER_0 are NOT supported; a write operation to the RX_BUFFER_0 will corrupt the buffer contents.

8.2.17 Register file: 0x13 – RX frame buffer 1

ID	Length (octets)	Type	Mnemonic	Description
13	1024	ROD	RX_BUFFER_1	RX frame data buffer 1

Register map register file 0x13:00 is the second receive data buffer. The data from the received frame is available in this received buffer, only when double buffer mode is enabled and the main buffer (RX_BUFFER_0) is full. Assuming successful reception of a good frame, the full length of received data (as reported (in SET_2) by the RXFLEN field of register 0x18:E8, copy of the RX_FINFO), will be available in the RX_BUFFER_1 beginning at offset 0. Note since the reported length includes the FCS the host system will probably choose not to read these final two octets.

Write operations to the RX_BUFFER_1 are NOT supported; a write operation to the RX_BUFFER_0 will corrupt the buffer contents.

8.2.18 Register file: 0x14 – Transmit data buffer

ID	Length (octets)	Type	Mnemonic	Description
14	1024	WO	TX_BUFFER	Transmit data buffer

Register map register file 0x14:00 is the transmit data buffer. Data from the transmit buffer is transmitted during the data payload portion of the transmitted frame. Section *3 Message transmission* discusses the basics of frame transmission and details the various parts of the TX frame.

The general procedure is to write the data frame for transmission into the TX_BUFFER, set the frame length and other details in the TX_FCTRL register and initiate transmission using in the start TX command (CMD_TX).

Note that read operations from the transmit data buffer are NOT supported.

8.2.19 Register file: 0x15 – Accumulator CIR memory

ID	Length (octets)	Type	Mnemonic	Description
15	12288	RO	ACC_MEM	Read access to accumulator data memory

Register map register file 0x15:00 is a large bank of memory that holds the accumulated channel impulse response (CIR) data. Really this is a channel impulse response estimate (CIRE) but the abbreviation CIR is used to mean the same thing throughout this manual. To accurately determine this timestamp the QM33100 incorporates a channel impulse analyser (CIA) algorithm to processes the CIR (in the accumulator) to find the leading edge and adjust the RMARKER receive timestamp as reported in Sub-register 0x00:60 – Receive time stamp.

The host system does not need to access the ACC_MEM in normal operation, however it may be of interest to the system design engineers to visualise the radio channel for diagnostic purposes.

The accumulator data memory actually contains one or two CIR depending on the STS packet configuration being employed. One CIR is generated by the accumulation of the repeated symbols of the preamble correlated against the expected symbol as determined by the RX preamble code configuration, PCODE in Sub-register 0x01:08 – Channel control. The other CIR is generated by the accumulation of the STS with each symbol received being correlated against the expected STS pulse pattern cryptographically generated locally in the receiver. Please refer to *§ 6 – Secure ranging / timestamping* for details of the operation of the STS for secure ranging operations.

Accessing the accumulator CIR memory is a little different than accessing other register files in two ways: Firstly for every SPI read access of the accumulator memory, a single dummy octet is output before the first byte of valid accumulator data. Secondly the offset specified in the SPI transaction is not the byte index, but instead is the sample index, where each accumulator sample is an complex value provided as a 24-bit (3-octet) real value followed by a 24-bit (3-octet) imaginary value. Each value is actually 18-bit precision, with the upper 6-bits being all zero or ones depending on the sign of the value.

Prior to reading from the accumulator CIR memory both ACC_CLK_EN bit and the ACC_MCLK_EN bit (in Sub-register 0x11:04 – Clock control) need to be set to 1 to allow the accumulator reading to operate correctly.

The preamble CIR begins at sample index 0 and has a span of one symbol, which is 992 samples at 16 MHz PRF, and, 1016 samples at 64 MHz PRF. The STS CIR begins at sample index 1024 and has a span of half a symbol time which is 512 samples irrespective of PRF setting. When PDoA Mode 3 is used the STS is used to determine two CIR estimates. The first will begin at sample 1024 and the second at sample 1536. Both CIRs are 512 samples long.

Table 45: Example SPI indexed read of accumulator CIR memory

Input Octets	Output Octets	Comment
0x2A	-	Indexed read from register file 0x15
0xDF	-	0x5F + bit 7 indicates it's a 15-bit offset specifier
0x05	-	High 8 bits of the 15-bit offset specifier
-	<Dummy Octet>	Ignore/skip this 1st octet output
-	sample [735] real part low 8-bits	First Complex value (6 octets)
-	sample [735] real part middle 8-bits	
-	sample [735] real part high 8-bits	
-	sample [735] imaginary part low 8-bits	
-	sample [735] imaginary part middle 8-bits	
-	sample [735] imaginary part high 8-bits	
-	sample [736] real part low 8-bits	Second Complex value (6 octets)
-	sample [736] real part middle 8-bits	
-	sample [736] real part high 8-bits	
-	sample [736] imaginary part low 8-bits	
-	sample [736] imaginary part middle 8-bits	
-	sample [736] imaginary part high 8-bits	

Input Octets	Output Octets	Comment
-	… etc. …	

8.2.20 Register file: 0x16 – Scratch RAM

ID	Length (octets)	Type	Mnemonic	Description
16	127	RW	SCRATCH_RAM	Scratch RAM memory buffer

Register map register file 0x16:00 is the scratch RAM memory buffer. This memory can be used as a temporary store, or during AES/DMA operations. The data will not be preserved if the device is powerd off or when it enters **SLEEP** or **DEEPSLEEP** state.

8.2.21 Register file: 0x17 – AES KEY RAM

ID	Length (octets)	Type	Mnemonic	Description
17		RW	AES_KEY_RAM	AES KEY RAM - storage for up to 8 x 128 bit AES KEYs

Register map register file 0x17 is a large bank of memory that holds up to 8 128-bit AES KEYs as shown in Table 45 below. The 32-bit KEY words need to be programmed starting with the MSB 32-bit word in the lowest register memory address, as shown in Table 45 below.

Table 46: Register file: 0x17 – AES KEY RAM overview

OFFSET in Register 0x17	Register data
00	1st AES key in RAM: For sample 128-bit AES KEY: 0x00112233445566778899aabbccddeeff, It should be written as follows: 0x17:00 = AES_KEY [127-96], byte[3] = 0x00 … byte[0] = 0x33 0x17:04 = AES_KEY [95-64], byte[3] = 0x44 … byte[0] = 0x77 0x17:08 = AES_KEY [63-32], byte[3] = 0x88 … byte[0] = 0xbb 0x17:0C = AES_KEY [31-0], byte[3] = 0xcc … byte[0] = 0xff
10	2nd AES key in RAM, for formatting see above
20	3rd AES key in RAM, for formatting see above
30	4th AES key in RAM, for formatting see above
40	5th AES key in RAM, for formatting see above
50	6th AES key in RAM, for formatting see above
60	7th AES key in RAM, for formatting see above
70	8th AES key in RAM, for formatting see above

8.2.22 Register file: 0x18 – Double buffer diagnostic register set

ID	Length (octets)	Type	Mnemonic	Description
0x18		RO	DB_DIAG	Double buffer diagnostic register set

Register map register file 0x18 is a large bank of memory that holds the double buffer diagnostic register set. It contains two swinging sets corresponding to two receive buffers (SET_1 and SET_2). Each set is 232 bytes long. The first set starts at address 0x18:00, and the second at address 0x18:E8, as shown in Table 47 below. The RDB_DMODE configuration specifies how much diagnostic data will be logged. The minimum configuration (i.e. when RDB_DMODE is set to 1) only logs 7 registers, the maximum logs all CIA diagnostic registers.

Table 47: Register file: 0x18 – Double buffer diagnostic register set overview

OFFSET in Register 0x18		RDB_DMODE	MINDIAG	Register data
SET_1	SET_2			
00	E8	1 or 2 or 4	0 or 1	RX_FINFO
04	EC	1 or 2 or 4	0 or 1	RX_TIME
0C	F4	1 or 2 or 4	0 or 1	CIA_DIAG_0
10	F8	1 or 2 or 4	0 or 1	TDOA
14	FC	1 or 2 or 4	0 or 1	PDOA
18	100	1 or 2 or 4	0 or 1	Reserved
1C	104	1 or 2 or 4	0 or 1	IP_DIAG_12
20	108	2 or 4	0 or 1	IP_TS
24	10C	2 or 4	0 or 1	Reserved
28	110	2 or 4	0 or 1	STS_TS
2C	114	2 or 4	0 or 1	Reserved
30	118	2 or 4	0 or 1	STS1_TS
34	11C	2 or 4	0 or 1	Reserved
38	120	4	0 or 1	CIA_DIAG_1
3C	124	4	0	IP_DIAG_0
40	128	4	0	IP_DIAG_1
44	12C	4	0	IP_DIAG_2
48	130	4	0	IP_DIAG_3
4C	134	4	0	IP_DIAG_4
50	138	4	0	Reserved
54	13C	4	0	Reserved
58	140	4	0	Reserved
5C	144	4	0	IP_DIAG_8
60	148	4	0	Reserved
64	14C	4	0	Reserved
68	150	4	0	Reserved
6C	154	4	0	STS_DIAG_0

OFFSET in Register 0x18		RDB_DMODE	MINDIAG	Register data
SET_1	SET_2			
70	158	4	0	STS_DIAG_1
74	15C	4	0	STS_DIAG_2
78	160	4	0	STS_DIAG_3
7C	164	4	0	STS_DIAG_4
80	168	4	0	Reserved
84	16C	4	0	Reserved
88	170	4	0	Reserved
8C	174	4	0	STS_DIAG_8
90	178	4	0	Reserved
94	17C	4	0	Reserved
98	180	4	0	Reserved
9C	184	4	0	STS_DIAG_12
A0	188	4	0	Reserved
A4	18C	4	0	Reserved
A8	190	4	0	Reserved
AC	194	4	0	Reserved
B0	198	4	0	Reserved
B4	19C	4	0	STS1_DIAG_0
B8	1A0	4	0	STS1_DIAG_1
BC	1A4	4	0	STS1_DIAG_2
C0	1A8	4	0	STS1_DIAG_3
C4	1AC	4	0	STS1_DIAG_4
C8	1B0	4	0	Reserved
CC	1B4	4	0	Reserved
D0	1B8	4	0	Reserved
D4	1BC	4	0	STS1_DIAG_8
D8	1C0	4	0	Reserved
DC	1C4	4	0	Reserved
E0	1C8	4	0	Reserved
E4	1CC	4	0	STS1_DIAG_12

8.2.23 Register file: 0x1D – Indirect pointer A

ID	Length (octets)	Type	Mnemonic	Description
1D	-	RW	INDIRECT_PTR_A	Indirect pointer A

Register map register file 0x1D:00 is the indirect pointer A. When reading or writing to this register the host will access register that was programmed into PTR_ADDR_A, at offset programmed into PTR_OFFSET_A. The indirect register addess is needed when accessing the register contents starting at offset > 127.

8.2.24 Register file: 0x1E – Indirect pointer B

ID	Length (octets)	Type	Mnemonic	Description
1E	-	RW	INDIRECT_PTR_B	Indirect pointer B

Register map register file 0x1E:00 is the indirect pointer B. When reading or writing to this register the host will access register that was programmed into PTR_ADDR_B, at offset programmed intoPTR_OFFSET_B. The indirect register addess is needed when accessing the register contents starting at offset > 127

8.2.25 Register file: 0x1F – FINT status and indirect pointer interface

ID	Length (octets)	Type	Mnemonic	Description
1F	19	-	IN_PTR_CFG	Indirect pointer configuration and fast interrupt status register

Register map register file 0x1F contains the reduced status events set status register and the indirect pointer configuration interface. The latter allows read/write access to registers with register subaddresses > 127. For example to read 10 bytes from RX_BUFFER_0 from any offset <= 127, a normal SPI read transaction should be used. However to read the same bytes from RX_BUFFER_0 with offset > 127, the indirect access needs to be used. This is achieved as follows (either pointer A (INDIRECT_PTR_A) or pointer B (INDIRECT_PTR_B) can be used for this):

- Program the base addres of the register to be accessed through the indirect pointer, if using pointer A then program PTR_ADDR_A
- Program the offset of the register to be accessed, if using pointer A then program PTR_OFFSET_A
- Read/Write the desired data by using a standard SPI read/write transaction from INDIRECT_PTR_A.

There are a number of sub-registers in this register file. An overview of these sub-registers is given by Table 48, and each is then separately described in the sub-sections below.

Table 48: Register file: 0x1F – FINT status and indirect pointer interface overview

OFFSET in Register 0x1F	Mnemonic	Description
0x00	FINT_STAT	Fast System Event Status Register
0x04	PTR_ADDR_A	Base address of register to be accessed through pointer A
0x08	PTR_OFFSET_A	Offset address of register to be accessed through pointer A
0x0C	PTR_ADDR_B	Base address of register to be accessed through pointer B
0x10	PTR_OFFSET_B	Offset address of register to be accessed through pointer B

8.2.25.1 Sub-register 0x1F:00 – Fast system event status

ID	Length (octets)	Type	Mnemonic	Description
1F:00	1	RO	FINT_STAT	Fast system event status register

Register file: 0x1F – FINT status and indirect pointer interface, sub-register 0x00 is the reduced system event status register, FINT_STAT. It contains status bits that indicate the occurrence of different system events or status changes. It is a reduced set of SYS_STATUS events. Reading the FINT_STAT register returns the state of the status bits as long as the matching SYS_ENABLE event bit is set. Thus it will report the events which gave rise to the interrupt. Generally these event status bits are latched so that the event is captured. The bits will have to be cleared by writing to relevant SYS_STATUS bits. The FINT_STAT register contains the system event status bits identified and described below:

REG:1F:00 – FINT_STAT – Fast system status register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								SYS_PANIC	SYS_EVENT	RXTO	RXERR	RXOK	RXTSERR	CCA_FAIL	TXOK
																								0	0	0	0	0	0	0	0

The system event status bits of the FINT_STAT register identified above are individually described below:

Field	Description of fields within Sub-register 0x1F:00 – Fast system event status
– reg:1F:00 bit:various	Bits marked '-' are reserved and should not be changed.
TXOK reg:1F:00 bit:0	This bit will be set if any of the following events are set in the SYS_STATUS register: TXFRB or TXPRS or TXPHS or TXFRS. This is a READ ONLY status flag – it will be cleared by writing 1 to relevant status bits (as listed above) in *Sub-register 0x00:44 – System event status*, whichever ones are set.
CCA_FAIL reg:1F:00 bit:1	This bit will be set if any of the following events are set in the SYS_STATUS register: AAT or CCA_FAIL. This is a READ ONLY status flag – it will be cleared by writing 1 to …. Status bit in *Sub-register 0x00:44 – System event status*.
RXTSERR reg:1F:00 bit:2	This bit will be set if event is set in the SYS_STATUS register: This is a READ ONLY status flag – it will be cleared by writing 1 to status bit in *Sub-register 0x00:44 – System event status*.
RXOK reg:1F:00 bit:3	This bit will be set if any of the following events are set in the SYS_STATUS register: RXFR and CIADONE or RXFCG. This is a READ ONLY status flag – it will be cleared by writing 1 to relevant status bits (as listed above) in *Sub-register 0x00:44 – System event status*.
RXERR reg:1F:00 bit:4	This bit will be set if any of the following events are set in the SYS_STATUS register: RXFCE or RXFSL or RXPHE or ARFE or RXSTO or RXOVRR. This is a READ ONLY status flag – it will be cleared by writing 1 to relevant status bits (as listed above) in *Sub-register 0x00:44 – System event status*, whichever ones are set.

Field	Description of fields within Sub-register 0x1F:00 – Fast system event status
RXTO reg:1F:00 bit:5	This bit will be set if any of the following events are set in the SYS_STATUS register: RXFTO or RXPTO. This is a READ ONLY status flag – it will be cleared by writing 1 to RXFTO or RXPTO status bits in *Sub-register 0x00:44 – System event status*, whichever ones are set.
SYS_EVENT reg:1F:00 bit:6	This bit will be set if any of the following events are set in the SYS_STATUS register: TIMER0 or TIMER1 or VT_DET or GPIOIRQ or RCINIT or SPIRDY. This is a READ ONLY status flag – it will be cleared by writing 1 to relevant status bits (as listed above) in *Sub-register 0x00:44 – System event status*, whichever ones are set.
SYS_PANIC reg:1F:00 bit:7	This bit will be set if any of the following events are set in the SYS_STATUS register: PGFCAL_ERR or AES_ERR or CMD_ERR or SPI_UNF or SPI_OVF or SPIERR or PLL_HILO or VWARN. This is a READ ONLY status flag – it will be cleared by writing 1 to relevant status bits (as listed above) in *Sub-register 0x00:44 – System event status*, whichever ones are set.

8.2.25.2 Sub-register 0x1F:04 – Pointer A reg base address

ID	Length (octets)	Type	Mnemonic	Description
1F:04	1	RW	PTR_ADDR_A	Base address of the register to be accessed through indirect pointer A

Register file: 0x1F – FINT status and indirect pointer interface, sub-register 0x04 contains the base address of the register to be accessed through indirect pointer A. The PTR_ADDR_A register is described below:

REG:1F:04 – PTR_ADDR_A – Base address of the register to access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	PTRA_BASE				
																								0	0	0	0	0	0	0	0

The bits of the PTR_ADDR_A register identified above are individually described below:

Field	Description of fields within Sub-register 0x1F:04 – Pointer A reg base address
PTRA_BASE reg:1F:04 bits:4-0	The base address of the register to be accessed through indirect pointer A (INDIRECT_PTR_A)
– reg:1F:04 bits:7-5	Bits marked '-' are reserved and should not be changed.

8.2.25.3 Sub-register 0x1F:08 – Pointer A reg offset address

ID	Length (octets)	Type	Mnemonic	Description
1F:08	2	RW	PTR_OFFSET_A	Offset address of the register to be accessed through indirect pointer A

Register file: 0x1F – FINT status and indirect pointer interface, sub-register 0x08 contains the offset address of the register to be accessed through indirect pointer A. The PTR_OFFSET_A register is described below:

REG:1F:08 – PTR_OFFSET_A – Base address of the register to access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	PTRA_OFS														
																-	0														

The bits of the PTR_OFFSET_A register identified above are individually described below:

Field	Description of fields within Sub-register 0x1F:08 – Pointer A reg offset address
PTRA_OFS reg:1F:08 bits:14-0	The offset address of the register to be accessed through indirect pointer A (INDIRECT_PTR_A).
– reg:1F:04 bit:15	Bits marked '-' are reserved and should not be changed.

8.2.25.4 Sub-register 0x1F:0C – Pointer B reg base address

ID	Length (octets)	Type	Mnemonic	Description
1F:0C	1	RW	PTR_ADDR_B	Base address of the register to be accessed through indirect pointer B

Register file: 0x1F – FINT status and indirect pointer interface, sub-register 0x0C contains the base address of the register to be accessed through indirect pointer B. **Note: in the API [3] indirect pointer B is reserved for double buffer SET_2 access.**The PTR_ADDR_B register is described below:

REG:1F:0C – PTR_ADDR_B – Base address of the register to access																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								-	-	-	PTRB_BASE				
																								0	0	0	0	0	0	0	0

The bits of the PTR_ADDR_B register identified above are individually described below:

Field	Description of fields within Sub-register 0x1F:0C – Pointer B reg base address
PTRB_BASE reg:1F:0C bits:4-0	The base address of the register to be accessed through indirect pointer B (INDIRECT_PTR_B).
– reg:1F:0C bits:7-5	Bits marked '-' are reserved and should not be changed.

8.2.25.5 Sub-register 0x1F:10 – Pointer B reg offset address

ID	Length (octets)	Type	Mnemonic	Description
1F:10	2	RW	PTR_OFFSET_B	Offset address of the register to be accessed through indirect pointer B

Register file: 0x1F – FINT status and indirect pointer interface, sub-register 0x08 contains the offset address of the register to be accessed through indirect pointer B. **Note: in the API [3] indirect pointer B is reserved for double buffer SET_2 access.** The PTR_OFFSET_B register is described below:

REG:1F:10 – PTR_OFFSET_B – Base address of the register to access																													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																-	PTRB_OFS														
																-	0														

The bits of the PTR_OFFSET_B register identified above are individually described below:

Field	Description of fields within Sub-register 0x1F:10 – Pointer B reg offset address
PTRA_OFS reg:1F:10 bits:14-0	The offset address of the register to be accessed through indirect pointer B (INDIRECT_PTR_B).
– reg:1F:10 bit:15	Bits marked '-' are reserved and should not be changed.

9 Fast Commands

This section lists and describes the single-octet commands used initiate specific IC activities. For details of format for the fast command SPI transaction please refer to section *2.3 –* . Table 49 lists the supported commands and their hex codes, and each command is described in separate sub-sections following this.

Table 49: List of supported fast commands

Command ID	Code	Brief Description
CMD_TXRXOFF	0x0	Puts the device into IDLE state and clears any events.
CMD_TX	0x1	Immediate start of transmission
CMD_RX	0x2	Enable RX immediately
CMD_DTX	0x3	Delayed TX w.r.t. DX_TIME
CMD_DRX	0x4	Delayed RX w.r.t. DX_TIME
CMD_DTX_TS	0x5	Delayed TX w.r.t. TX timestamp + DX_TIME
CMD_DRX_TS	0x6	Delayed RX w.r.t. TX timestamp + DX_TIME
CMD_DTX_RS	0x7	Delayed TX w.r.t. RX timestamp + DX_TIME
CMD_DRX_RS	0x8	Delayed RX w.r.t. RX timestamp + DX_TIME
CMD_DTX_REF	0x9	Delayed TX w.r.t. DREF_TIME + DX_TIME
CMD_DRX_REF	0xA	Delayed RX w.r.t. DREF_TIME + DX_TIME
CMD_CCA_TX	0xB	TX frame if no preamble detected
CMD_TX_W4R	0xC	Start TX immediately, then when TX is done, enable the receiver
CMD_DTX_W4R	0xD	Delayed TX w.r.t. DX_TIME, then enable receiver
CMD_DTX_TS_W4R	0xE	Delayed TX w.r.t. TX timestamp + DX_TIME, then enable receiver
CMD_DTX_RS_W4R	0xF	Delayed TX w.r.t. RX timestamp + DX_TIME, then enable receiver
CMD_DTX_REF_W4R	0x10	Delayed TX w.r.t. DREF_TIME + DX_TIME, then enable receiver
CMD_CCA_TX_W4R	0x11	TX packet if no preamble detected, then enable receiver
CMD_CLR_IRQS	0x12	Clear all interrupt events
CMD_DB_TOGGLE	0x13	Toggle double buffer pointer / notify the device that the host has finished processing the received buffer/data.

9.1 CMD_TXRXOFF

This command puts the device into **IDLE_PLL** state, it will turn off the transmitter or receiver if the device is actively transmitting or receiving, thus any active transmission or reception is aborted.

This command must be issued only when device is in **IDLE_PLL**, **TX** or **RX** state, otherwise the device will need a restart. It should not be issued when device is in **INIT_RC** or **IDLE_RC** states.

Note:

If host has configured the device into TX test mode: e.g. continuous frame mode (when TX_PSTM **bit is set), issuing TRXOFF will not put the device into IDLE state.**

9.2 CMD_TX

This is a start TX command. It commands the QM33100 to start transmission. The transmission will start immediately once the TX blocks are powered up. In general it would be expected that the user has a prepared frame in the transmit buffer and has configured the desired transmit mode and set the frame length. For general discussion of transmission see section 3 Message transmission.

9.3 CMD_RX

This command enables the receiver. It commands the QM33100 to turn on its receiver and begin looking for the configured preamble sequence. It is assumed that the all necessary configurations have been made before turning on the receiver. For general discussion of reception see section 4 Message reception.

9.4 CMD_DTX

This command instructs the device to do a delayed transmission. This control works in conjunction with the DX_TIME value specified by Sub-register 0x00:28 – Delayed send or receive time. When the user wants to control the time of sending of a packet, the send time is programmed into DX_TIME, and then this command issued.

When delayed sending is used the QM33100 precisely controls the transmission start time so that the internal TX timestamp occurs at the point when SYS_TIME is equal to the DX_TIME value. The actual time of TX then is calculable as DX_TIME plus the TX antenna delay.

9.4.1 **Delayed TX notes:**

CMD_DTXhas a number of uses: -

• It can be used to give precise control of the transmission time of a response message, which would allow a receiver that knows this response time to only turn on at the correct time to receive the response, thus saving power.

• In symmetric double-sided two-way ranging, the RX to TX response times at either end should be the same so that their differences in local clocks correctly cancel out. This may be ensured by setting DX_TIME to a value that is a fixed delta added to the RX time-stamp.

• In two-way ranging the TX timestamp of the final message exchange needs to be communicated to the receiving end to allow the round-trip delay to be calculated. Using CMD_DTX allows this time to be predicted, pre-calculated and embedded into the final message itself. This may save the need for an additional message interchange which will give a power saving, and save time too.

• Embedding the TX time in this way may also reduce the number of messages in a wireless clock synchronisation scheme.

9.5 CMD_DRX

This command instructs the device to turn on the receiver with a delay. This command works in conjunction with the DX_TIME value specified by Sub-register 0x00:28 – Delayed send or receive time. When the user wants to control the time of turning on the receiver, the turn on time is programmed into DX_TIME, and then this command executed. The QM33100 then precisely controls the RX turn on time so that it is ready to receive the first symbol of preamble at the specified DX_TIME start time. In cases when the received time can be known precisely, for example when a response is expected at a well-defined time, employing CDM_DRX will give a power saving as it allows the IC to remain idle until the moment it is required to act for the reception

9.6 CMD_DTX_TS

This command will do a delayed transmission with respect to the last transmission timestamp. The new TX timestamp will be the combination of the previous TX timestamp and the delay programmed in the DX_TIME value specified by Sub-register 0x00:28 – Delayed send or receive time (i.e. TX_TSnew = TX_Tsold + DX_TIME). See also notes in delayed TX section 9.4.1.

9.7 CMD_DRX_TS

This command will instruct the device to turn on its receiver with respect to the last transmission timestamp. The receiver turn on time will be the combination of the previous TX timestamp and the delay programmed in the DX_TIME value specified by Sub-register 0x00:28 – Delayed send or receive time.

9.8 CMD_DTX_RS

This command will do a delayed transmission with respect to the last receive timestamp. The new TX timestamp will be the combination of the previous RX timestamp (RX_TIME) and the delay programmed in the DX_TIME value specified by Sub-register 0x00:28 – Delayed send or receive time (i.e. TX_Tsnew = RX_Tsold + DX_TIME). See also notes in delayed TX section 9.4.1.

9.9 CMD_DRX_RS

This command will instruct the device to turn on its receiver with respect to the last receive timestamp. The receiver turn on time will be the combination of the previous RX timestamp and the delay programmed in the DX_TIME value specified by Sub-register 0x00:28 – Delayed send or receive time

9.10 CMD_DTX_REF

This command will do a delayed transmission with respect to the time programmed into the DREF_TIME register. The new TX timestamp will be the combination of the DREF_TIME and the delay programmed in the DX_TIME value specified by Sub-register 0x00:28 – Delayed send or receive time (i.e. TX_Tsnew = DREF + DX_TIME). See also notes in delayed TX section 9.4.1.

9.11 CMD_DRX_REF

This command will instruct the device to turn on its receiver with respect to the time programmed into the DREF_TIME register. The receiver turn on time will be the combination of the DREF_TIME and the delay programmed in the DX_TIME value specified by Sub-register 0x00:28 – Delayed send or receive time

9.12 CMD_CCA_TX

This command instructs the device to perform a pseudo CCA before starting transmission. Once this command is issued the device will turn on the receiver and failing to detect preamble within the time programmed in the preamble detection timeout register (*Sub-register 0x06:04 – Preamble detection* timeout count) it will start the transmission of the packet. If the preamble detection occurs, the transmission will be aborted and preamble detection signalled (RXPRD).

9.13 CMD_TX_W4R

Similarly to the start TX command above. It initially commands the QM33100 to start transmission, and then once the transmission is complete the device will enable its receiver. Optionally receiver can be enabled with a delay programmed into W4R_TIM in Sub-register 0x01:00 – Acknowledgement time and response time.

9.14 CMD_DTX_W4R

Similarly to the delayed TX command above. It initially commands the QM33100 to start delayed transmission (as described in CMD_DTX), and then once the transmission is complete the device will enable its receiver. Optionally receiver can be enabled with a delay programmed into W4R_TIM in Sub-register 0x01:00 – Acknowledgement time and response time. See also notes in delayed TX section 9.4.1.

9.15 CMD_DTX_TS_W4R

Similarly to the delayed TX command above. It initially commands the QM33100 to start delayed transmission (as described in CMD_DTX_TS), and then once the transmission is complete the device will enable its receiver. Optionally receiver can be enabled with a delay programmed into W4R_TIM in Sub-register 0x01:00 – Acknowledgement time and response time. See also notes in delayed TX section 9.4.1.

9.16 CMD_DTX_RS_W4R

Similarly to the delayed TX command above. It initially commands the QM33100 to start delayed transmission (as described in CMD_DTX_RS), and then once the transmission is complete the device will enable its receiver. Optionally receiver can be enabled with a delay programmed into W4R_TIM in Sub-register 0x01:00 – Acknowledgement time and response time. See also notes in delayed TX section 9.4.1.

9.17 CMD_DTX_REF_W4R

Similarly to the delayed TX command above. It initially commands the QM33100 to start delayed transmission (as described in CMD_DTX_REF), and then once the transmission is complete the device will enable its receiver. Optionally receiver can be enabled with a delay programmed into W4R_TIM in Sub-register 0x01:00 – Acknowledgement time and response time. See also notes in delayed TX section 9.4.1.

9.18 CMD_CCA_TX_W4R

Similarly to the pseudo CCA TX command above. It initially commands the QM33100 to start transmission (as described in CMD_CCA_TX), and then once the transmission is complete the device will enable its receiver. Optionally receiver can be enabled with a delay programmed into W4R_TIM in Sub-register 0x01:00 – Acknowledgement time and response time

9.19 CMD_CLR_IRQS

This command will clear all of the status events and will clear the interrupt if set.

9.20 CMD_DB_TOGGLE

This command is only applicable when the device is using double buffering (the DIS_DRXB is set to 0). It will notify the device that the host has finished with the last RX buffer and the buffer is free for the device to use for another RX packet reception.

10 Calibration

The operating characteristics and performance of the QM33100 is dependent on the IC itself and on its external circuitry and on its operating environment. To give optimum performance it is necessary to calibrate the IC to account for factors which affect its operation.

Some calibration parameters are dependent solely on process variations that occur within the silicon of the IC during its manufacture. These are typically measured during IC production test and the required calibration parameters are written to the OTP memory of the QM33100. The host system software can then use these values during QM33100 configuration to optimise the QM33100 performance.

Some calibration parameters are dependent on circuit elements external to the IC. These can only be determined during the manufacture of the product e.g. a module into which the QM33100 is soldered. These parameters are typically measured during product test and the required calibration parameters are stored in an area of the QM33100's OTP memory which has been allocated for test calibration parameters. The host system software will use this calibration data during QM33100 configuration to optimise the QM33100 performance.

Some calibration parameters may vary according to the operational environment of the QM33100. For example some parameters may need to be changed if there are large variations in the ambient temperature (e.g. moving from a warm area into a cold store). In such circumstances in order to optimise the QM33100 performance the host system software can monitor the voltage and temperature using QM33100 and adjust configuration accordingly.

Elements of the QM33100 that may be subject to calibration are:

- Crystal trimming – the QM33100 contains trimming capacitors that can fine tune the operating frequency of its crystal oscillator.
- Transmitter output power and spectrum – the QM33100 output spectrum is tuneable to meet regional spectral regulations and maximise the output power to achieve the greatest operating range.
- Antenna delay – the QM33100 antenna delay may be fine-tuned to give best possible ranging or location accuracy.
- IC calibration – PLL calibration over temperature.

The sub-sections below detail the calibration of these QM33100 parameters.

10.1 IC calibration – crystal oscillator trim

QM33100 is specified to operate with clock offsets between the transmitting and receiving nodes of up to ±20 ppm. The receiver sensitivity of QM33100 can be improved by reducing the relative offset in clocks between the transmitting and receiving nodes. One way to reduce the offset is to use temperature compensated crystal oscillators (TCXO) in both transmitting and receiving nodes or just in one side of the link. Using TCXOs increases system cost and current consumption so generally they are only used in fixed anchor scenarios. Where crystals are used as the clock reference, QM33100 provides a facility to trim the

oscillator frequency by switching in internal capacitor banks in parallel with the external loading capacitors associated with the chosen crystal. This trimming can be used to reduce the crystal initial frequency error and to compensate for temperature and aging drift, if required.

The amount of trimming is programmable through Sub-register 0x09:14 – Crystal trim register.

10.1.1 Calibration method

Please see API and example code [3].

10.2 IC calibration – transmit power and spectrum

QM33100 provides registers to adjust the transmit power and spectrum bandwidth to meet regulatory limits. Qorvo provides an application note (APS312 [6]) which describes this in detail. The following registers are provided for control of the transmit power and bandwidth..

Transmit Power	TX_POWER
Bandwidth	PG_DELAY

10.2.1 Calibration method

Please see API and example code [3], , and Application Note APS312 [6]

10.3 IC calibration – antenna delay

In order to measure range accurately, precise calculation of timestamps is required. To do this, a delay called the antenna delay must be known. The QM33100 allows this delay to be calibrated and provides the facility to compensate for delays introduced by PCB, external components, antenna and internal QM33100 delays.

To calibrate the antenna delay, range is measured at a known distance using two QM33100 systems. Antenna delay is adjusted until the known distance and reported range agree. The antenna delay can be stored in OTP memory.

There is a Transmitter Antenna Delay and a Receiver Antenna Delay. The Transmitter Antenna Delay is used to account for the delay between the internal digital timestamp of the RMARKER (as shown in Figure 13) inside the QM33100 and the time the RMARKER is transmitted from the antenna. The Receiver Antenna Delay is used to account for the delay between the time of arrival of the RMARKER at the antenna and the internal digital timestamp of the RMARKER inside the QM33100.

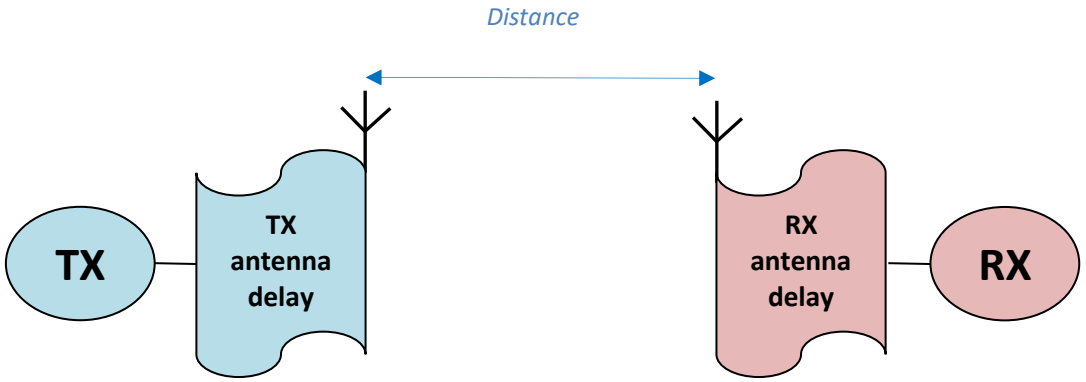

Figure 30: Transmit and Receive Antenna Delay

10.3.1 Calibration method

See API and examples [3], and Application Note APS014

10.4 IC Calibration -PLL calibration over temperature

For proper device operation, when using channel 9, the PLL will need to be re-calibrated if the device temperature changes by 20° C, using the steps below:

- firstly the device state needs to be changed to **IDLE_RC** state, by clearing AINIT2IDLE configuration bit in SEQ_CTRL and setting the FORCE2INIT bit
- then setting system clocks to FAST_RC, by setting the SYSCLKS field to 0x3
- then clearing the FORCE2INIT bit
- and restoring system clocks to Auto mode, by setting setting the SYSCLKS field to 0x0
- finally the host should set the AINIT2IDLE configuration bit in SEQ_CTRL register to cause the IC to enable the PLL and wait for it to lock before entering the **IDLE_PLL** state

This only applies to channel 9. When using channel 5 the PLL does not need to be re-calibrated if temperature changes.

11 Location schemes

This part of the discussion on operational design choices relates to RTLS location schemes. Some of the ideas and points discussed may be more generally applicable.

In general to locate a mobile node measurements are needed to be referenced to a number of fixed known location “anchor” nodes. Typically a minimum of three anchor nodes are needed to locate a mobile node in two dimensions, while a minimum four non-coplanar anchors are needed to locate a mobile node in three dimensions. The spacing of anchors nodes in an installation has to be such that four anchors are always in communication range of the mobile tag no matter where it is within the operating space. The communication range is dependent on data rate and preamble length, the choice of which is influenced by the node density requirements and perhaps also power consumption.

There are two general methods of doing location. These are time difference of arrival (TDoA) based location and time of flight (ToF) based location. The main operational points of each are outlined below. In either case the calculation of location, combining measurements from multiple anchors, is typically done by a software functionality called the *central location engine*.

Time of flight location requires two-way communication from the mobile node (tag) to each of the anchor nodes in its vicinity. Periodic message exchanges are used to measure the round trip delay and hence calculate the one way flight time between tag and anchor. The ToF times multiplied by the speed of light (and radio waves) gives the distance between the tag and the anchor. Each distance estimate defines a spherical surface, centred on the anchor, on which the tag must lie. The tag’s 3D location is yielded by the intersection of the spheres resulting from ToF measurements to the four anchors.

In time difference of arrival (TDoA) location the mobile tag blinks periodically and the blink message is received by the anchor nodes in its vicinity. When the anchor nodes have synchronised clocks so that the arrival time of the blink message at all nodes can be compared, then for each pair of anchors the time difference in the arrival of the blink message defines a hyperbolic surface on which the sending tag must lie. The tag’s 3D location is yielded by the intersection of the hyperbolic surfaces defined by the TDoA of the blink at four pairs of anchors.

For low power RTLS deployments the TDoA scheme has benefits in that the tag needs to only send a single message in order for it to be located. In contrast in the ToF scheme the tag has to send and receive multiple messages with multiple anchors, and it needs to know what anchors are in the vicinity so it can address each of them in turn correctly. ToF does not need synchronised anchors, and may suit the case where a hand held device calculates its own location as part of a navigation system. The TDoA is a lower power solution as there are fewer messages involved, and this also suits higher density deployments. The TDoA anchor clock synchronisation may be achieved via a wired clock distribution. Alternatively there are wireless techniques for clock synchronisation. Wired synchronisation may suit higher tag densities as it allows anchors to listen all the time so no tag blinks are missed or collide with the wireless clock sync messages (potentially disrupting synchronisation). Anchors should be wired for power and also with Ethernet to communicate the arrival times to the central location engine.

Two-way ranging (ToF) is good for proximity detection and separation alarms, especially when both parties in the exchange are mobile nodes.

In an RTLS the accuracy of the QM33100's RX timestamps can give sub 10 cm resolution. Note, however that the geometry of anchors with respect to the tag can smear the accuracy of the calculated location when individual measurements are combined. Having additional anchors in range of the tag can offset this if it allows the system to select anchors with best geometry and best receive signal quality with respect to the tag being located.

12 APPENDIX 1: Two-way ranging

12.1 Introduction

This appendix is for information only and describes various methods of implementing a two-way ranging scheme between two nodes.

The chosen two-way ranging algorithm is implemented by host system software and is not a feature of the QM33100. The QM33100 just provides the facilities for message time-stamping and precise control of message transmission times that enable these algorithms. See section *4.1.7 – RX message* timestamp, *3.2 – Transmission* timestamp and *3.3 – Delayed* transmission for details of this.

In all of the schemes that follow one node acts as *Initiator*, initiating a range measurement, while the other node acts as a *Responder* listening and responding to the initiator, and calculating the range.

12.2 Single-sided two-way ranging

Single-sided two-way ranging (SS-TWR) involves a simple measurement of the round trip delay of a single message from one node to another and a response sent back to the original node.

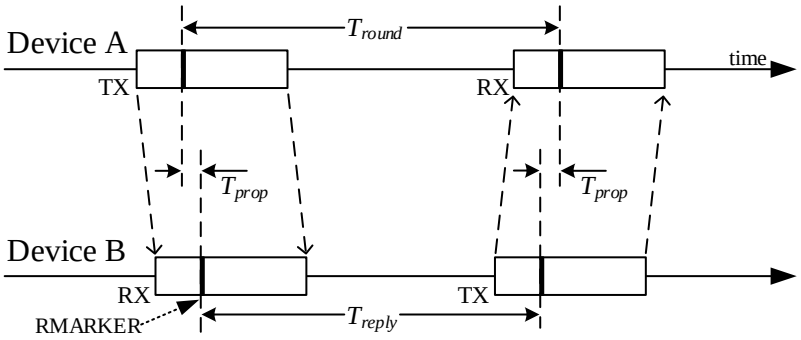

Figure 31: Single-sided two-way ranging

The operation of SS-TWR is as shown in Figure 31, where device A initiates the exchange and device B responds to complete the exchange and each device precisely timestamps the transmission and reception times of the packets, and thus can calculate times T_{round} and T_{reply} by simple subtraction. And the resultant time-of-flight, T_{prop} may be estimated by the equation:

$$\hat{T}_{prop} = \frac{1}{2}(T_{round} - T_{reply})$$

The times T_{round} and T_{reply} are measured independently by device A and B using their respective local clocks, which both have some clock offset error e_A and e_B from their nominal frequency, and so the resulting time-of-flight estimate has a considerable error that increases as T_{reply} increases. QM33100 however is able to measure the clock offset of the remote transmitter, (see REG_DRX_DIAG3), and this may be used to compensate for that error, using the modified equation below, to produce results that are as good as can be achieved using DS-TWR where the reply times are not too long, (i.e. < 5 ms).

$$\hat{T}_{prop} = \frac{1}{2}\left(T_{round} - T_{reply}\,(1 - C_{offest})\right)$$

12.3 Double-sided two-way ranging

12.3.1 Using four messages

Double-sided two-way ranging (DS-TWR) is an extension of the basic single-sided two-way ranging in which two round trip time measurements are used and combined to give a time-of-flight result which has a reduced error even for quite long response delays.

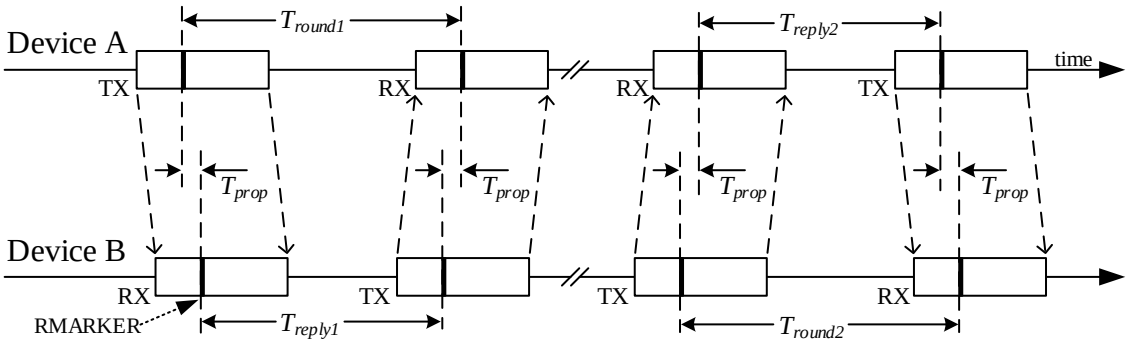

Figure 32: Double-sided two-way ranging with four messages

The operation of DS-TWR is as shown in Figure 32, where device A initiates the first round trip measurement to which device B responds, after which device B initiates the second round trip measurement to which device A responds completing the full DS-TWR exchange. Each device precisely timestamps the transmission and reception times of the messages.

12.3.2 Using three messages

The four messages of DS-TWR, shown in Figure 32, can be reduced to three messages by using the reply of the first round-trip measurement as the initiator of the second round-trip measurement. This is shown in Figure 33.

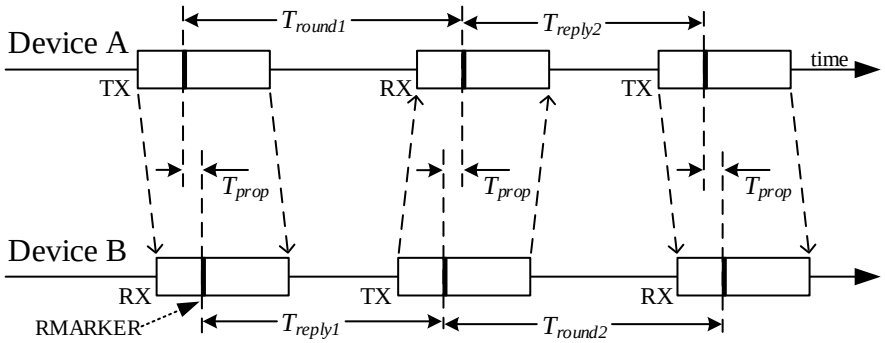

Figure 33: Double-sided two-way ranging with three messages

The resultant time-of-flight estimate, T_{prop}, in both the three and four message cases may be calculated using the expression:

$$\hat{T}_{prop} = \frac{(T_{round1} \times T_{round2} - T_{reply1} \times T_{reply2})}{(T_{round1} + T_{round2} + T_{reply1} + T_{reply2})}$$

This scheme is denoted **ASYMMETRIC** because it does not require the reply times from each device to be the same. Using this scheme, the typical clock induced error is in the low picosecond range even with 20 ppm crystals.

The asymmetric method allows complex ranging schemes to be achieved with a small number of messages. For example ranging from a tag to three anchors, can be achieved as per in Figure 34 where the tag can be located after sending only 2 messages and receiving 3.

This represents a substantial saving in message traffic thereby saving battery power and air-time. This assumes that the anchors are networked and pool the range measurements in some centralised location engine function that calculates the estimate of the tag's location.

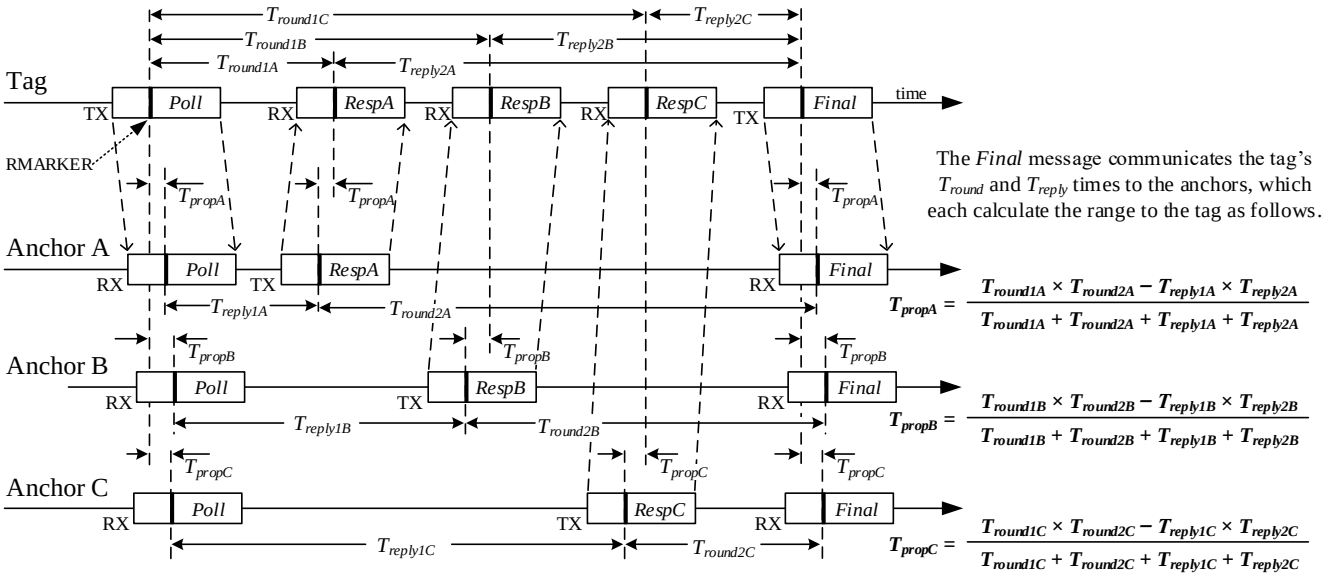

Figure 34: Ranging to 3 anchors with just 5 messages where each anchor calculates its own range result

13 APPENDIX 2: Abbreviations and acronyms

Abbreviation	Full Title	Explanation
ACK	acknowledgement (frame)	A frame sent in response to a received frame indicating successful reception. DW3000QM33100 allows the automatic generation of such frames when appropriately configured.
AES	advanced encryption standard	AES is a symmetric block cipher used to encrypt data. In the DW3000QM33100 this is implemented in a dedicated hardware block. A separate AES block is used for generating (and receiving) the STS if this is enabled to be included in the the packet.
AGC	automatic gain control	A scheme that automatically adjusts the gain of the receiver depending on the power in the received signal
AON	always-on block (for saving configuration)	A section of memory in DW3000QM33100 whose contents are retained provided VDD1 is maintained above the minimum limit specified in the DW3000QM33100 Datasheet [5]. Intended for saving device configuration during the **SLEEP** and **DEEPSLEEP** states and restoring it thereafter.
BPM	burst position modulation	A modulation scheme in which information is conveyed by the position of a burst of pulses in one of a number of possible positions in a symbol
BPSK	binary phase-shift keying	A modulation scheme in which information is conveyed by whether pulses are positive or negative
CDI	Controller Data input	SPI interface I/O
CDO	Controller Data output	SPI interface I/O
CIA	channel impulse analyser	This is the algorithm that processes the CIR (in the accumulator) to find the leading edge during the process of estimating the RX timestamp for a received packet. The CIA may be applied to CIR resulting from preamble and/or STS sequences.
CIR	channel impulse response	The impulse response of the communications channel between the transmitter and receiver as detected by DW3000QM33100 for the most recently received packet. Really this is a channel impulse response estimate (CIRE) but the abbreviation CIR is used to mean the same thing throughout this manual.
CRC	cyclic redundancy check (an FCS)	Error detecting code appended to the frame in the transmitter to allow detection of errors at the receiver.
DPS	dynamic preamble select	Anti-spoofing mechanism to allow IEEE 802.15.4 [1] devices change their preamble codes during ranging from those in normal use.
ESD	electrostatic discharge	A sudden flow of electrical current between two electrically charged objects caused by contact, an electrical short circuit or dielectric breakdown. Can cause failure of semiconductor devices. DW3000QM33100 is resistant to ESD up to the limits specified in the Datasheet.
EUI	extended unique identifier	64-bit IEEE device address. Refer to *Sub-register 0x00:04 – Extended* Unique Identifier.

Abbreviation	Full Title	Explanation
FAST_RC	fast RC oscillator	Fast RC oscillator which runs at approx. 120 MHz.
FCS	frame check sequence (the CRC)	A CRC appended to the frame in the transmitter to allow detection of errors at the receiver.
IF	intermediate frequency	A frequency to which a carrier frequency is shifted as an intermediate step in transmission or reception.
LCSS	Low cross-correlation sum set	This is a block in the DW3000QM33100 receiver that modifies the STS correlation function to reduce side-lobes and produce a cleaner CIR accumulation of the STS.
LDC	low duty-cycle	Certain regulatory jurisdictions define rules that limit the duration of UWB transmissions per unit time in certain channels. These rules are generally referred to as low duty cycle rules
LDO	low drop-out voltage regulator	Linear voltage regulator that requires only a small differential between its input source voltage and its output regulated voltage below which it can no longer regulate correctly. DW3000QM33100 uses a number of such regulators.
LNA	low noise amplifier	Circuit normally found at the front-end of a radio receiver designed to amplify very low level signals while keeping any added noise to as low a level as possible
LOS	line of sight	Physical radio channel configuration in which there is a direct line of sight between the transmitter and the receiver
NLOS	non line of sight	Physical radio channel configuration in which there is no direct line of sight between the transmitter and the receiver
OTP	one-time programmable (memory)	Internal memory in DW3000QM33100 that can be programmed once to store various identification and calibration values
PAC	preamble acquisition chunk	A group of preamble symbols which are correlated together in the preamble detection process in the receiver. The size of the PAC is configurable – see DTUNE0.
PDoA	phase difference of arrival	The difference in the phase of a signal as received at a pair of antennas, which can provide information on the bearing of the transmitting device.
PHR	PHY header	Part of the packet that comes before the PHY payload.
PHY	physical layer	Defined in the context of the OSI 7-layer model for communications systems in general and the IEEE802.15.4 [1] standard in particular, the PHY layer is the lowest layer in the 7-layermodel and defines the physical interface to the communications medium
PLL	phase locked loop	Phase locked loop used to generate stable frequency clocks. These are used in DW3000QM33100 to generate carrier frequencies and system clocks.
PRF	pulse repetition frequency	Defined in the context of the IEEE802.15.4 [1] standard. This is the frequency at which pulses are repeated in the preamble and data portions of a packet depending on the chosen configuration.

Abbreviation	Full Title	Explanation
PSR	preamble symbol repetitions	Used to define the overall preamble length. A larger number of preamble symbol repetitions give a longer preamble.
RESV	Reserved	Reserved mode, software should switch to preferred mode on start up.
RF	radio frequency	Generally used to refer to signals in the range of 3 kHz to 300 GHz. In the context of a radio receiver, the term is generally used to refer to circuits in a receiver before down-conversion takes place and in a transmitter after up-conversion takes place
RMARKER	ranging marker	The RMARKER is the time when the peak pulse location associated with the first chip period following the SFD is at the local antenna.
RSSI	received signal strength indication	An abbreviation for the measured (or estimated) receive power level.
RTLS	real time location systems	System intended to provide information on the location of various items in real-time.
RX	receive or receiver	Term used to refer to the receiver section of a transceiver or the operation of receiving signals
SAR	Successive Approximation Register ADC	A type of Analog to Digital converter that used a digital binary search to converge on the correct digital representation of the analog input level.
SECDED	single error correct, double error detect	A parity check sequence (used in the PHR) that allows the: • detection and correction of a single bit error in a group of bits or • the detection but not the correction of a double bit error
SFD	start of frame delimiter	Defined in IEEE802.15.4 [1]. The SFD marks the completion of the preamble section of the packet and the start of the payload or STS section. depending on the packet type.
SHR	synchronisation header	Defined in the context of the IEEE802.15.4 standard. The SHR consists of the preamble and the SFD.
SP3	STS packet configuration 3	A packet configuration where the packet consists of just preamble, SFD and STS.
SPI	serial peripheral interface	An industry accepted method for interfacing between IC's using a synchronous serial scheme first introduced by Motorola
STS	scrambled timestamp sequence	A term used to referThis refers to a sequence of pseudo-randomized pulses with cryptographicallygenerated using a deterministicpseudo random positive or negative phase, that isbit generator (DRBG) and included in the UWB packet as per Figure 13 depending on the STS packet configuration.
TDoA	time difference of arrival	Method of deriving information on the location of a transmitter. The time of arrival of a packet at two physically different locations whose clocks are synchronized is noted and the difference in the arrival times provides information on the location of the transmitter. A number of such TDoA measurements at different locations can be used to uniquely determine the position of the transmitter. Refer to Decawave's Qorvo's website for further information.

Abbreviation	Full Title	Explanation
ToA	Time of arrival	The receive time of a packet, generally referenced to the RMARKER, sometimes called the receive timestamp.
ToF	time of flight	The time taken for a radio signal to travel between the transmitting antenna and the receiving antenna
TX	transmit or transmitter	Term used to refer to the transmitter section of a transceiver or the operation of transmitting signals
UWB	ultra wide-band	A radio scheme employing channel bandwidths of, or in excess of, 500MHz
WSN	wireless sensor networks	A network of wireless nodes intended to enable the monitoring and control of the physical environment

14 APPENDIX 3: References

[1]	IEEE Std 802.15.4™-2020. IEEE Standard for Low-Rate Wireless Networks. Developed by the LAN/MAN Standards Committee of the IEEE Computer Society. Available from: http://standards.ieee.org/
[2]	IEEE Std 802.15.4z™-2020 (Amendment to IEEE Std 802.15.4™-2020). IEEE Standard for Low-Rate Wireless Networks. Amendment 1: Enhanced Ultra Wideband (UWB) Physical Layers (PHYs) and Associated Ranging Techniques. Developed by the LAN/MAN Standards Committee of the IEEE Computer Society. Available from: http://standards.ieee.org/
[3]	QM33100 APIs and Simple examples release – code available from http://www.qorvo.com/
[4]	IEEE 802.15.8IEEE Standard for Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Peer Aware Communications (PAC). Available from: http://standards.ieee.org/
[5]	QM33100 Datasheet
[6]	APS312 Production tests for DW3000 based products – application note available from http://www.qorvo.com/
[7]	Fira Consortium - https://www.firaconsortium.org/

15 APPENDIX 4: Reference TX_POWER setting tables

The tables below provide reference settings for which PHR byte is reduced relatively to DATA_PWR, SHR_PWR and STS_PWR bytes.
This is required for optimization of output power and compliance with regulations.

It is advised to refer to these tables when selecting a reference value for tx_power register.

The last column of each table indicates an estimate of the power difference between BASE_PWR (DATA_PWR, SHR_PWR and STS_PWR bytes) and PHR_PWR.
The power difference is calculated in tenth of dB and should be as close as possile to 60. For some coarse and fine gain combinations, it is not possible to apply exactly 6dB dial back. This is generally observed for lower power setting.
In such condition, the table would indicate the setting providing the closest.difference to 6dB dial back.

Note: the reference tables are indicative only.
After selecting a reference tx power setting, it is required to verify that the PHR part of the frame is transmitted 6dB lower than other parts of the frame. If the reference setting is not sufficient, then it is recommended to reduce either the coarse or fine gain further.

Tables for channel 9:

Table 50: Reference TxPower setting CHAN9, Coarse 0

Coarse	Fine	TxPower Setting	Power : BASE- PHR 0.1dB steps
0	63	0xFCFC68FC	60
0	62	0xF8F868F8	59
0	61	0xF4F464F4	61
0	60	0xF0F064F0	60
0	59	0xECEC64EC	60
0	58	0xE8E864E8	59
0	57	0xE4E460E4	61
0	56	0xE0E060E0	60
0	55	0xDCDC60DC	59
0	54	0xD8D85CD8	61
0	53	0xD4D45CD4	60
0	52	0xD0D05CD0	59
0	51	0xCCCC5CCC	58
0	50	0xC8C858C8	60
0	49	0xC4C458C4	59
0	48	0xC0C054C0	61
0	47	0xBCBC54BC	60
0	46	0xB8B854B8	58
0	45	0xB4B450B4	61

0	44	0xB0B050B0	59
0	43	0xACAC50AC	58
0	42	0xA8A84CA8	60
0	41	0xA4A44CA4	58
0	40	0xA0A048A0	61
0	39	0x9C9C489C	59
0	38	0x98984498	61
0	37	0x94944494	60
0	36	0x90904490	58
0	35	0x8C8C408C	61
0	34	0x88884088	59
0	33	0x84843C84	62
0	32	0x80803C80	60
0	31	0x7C7C3C7C	57
0	30	0x78783878	60
0	29	0x74743874	58
0	28	0x70703470	60
0	27	0x6C6C346C	57
0	26	0x68683068	61
0	25	0x64643064	58
0	24	0x60602C60	61
0	23	0x5C5C2C5C	58
0	22	0x58582858	62
0	21	0x54542854	59
0	20	0x50502450	62
0	19	0x4C4C244C	58
0	18	0x48482048	62
0	17	0x44442044	58
0	16	0x40401C40	62
0	15	0x3C3C1C3C	57
0	14	0x38381838	63
0	13	0x34341834	58
0	12	0x30301430	61
0	11	0x2C2C142C	55
0	10	0x28281028	62
0	9	0x24241024	55
0	8	0x20200C20	61
0	7	0x1C1C0C1C	52
0	6	0x18180818	58
0	5	0x14140414	62
0	4	0x10100010	58
0	3	0xC0C000C	43
0	2	0x8080008	25
0	1	0x4040004	11

0	0	0x0000000	0

Table 51: Reference TxPower setting CHAN9, Coarse 1

Coarse	Fine	TxPower Setting	Power : BASE- PHR 0.1dB steps
1	63	0xFDFDA0FD	60
1	62	0xF9F9A0F9	59
1	61	0xF5F59CF5	60
1	60	0xF1F19CF1	59
1	59	0xEDED9CED	59
1	58	0xE9E998E9	60
1	57	0xE5E594E5	60
1	56	0xE1E194E1	59
1	55	0xDDDD90DD	60
1	54	0xD9D990D9	59
1	53	0xD5D58CD5	60
1	52	0xD1D18CD1	59
1	51	0xCDCD88CD	60
1	50	0xC9C984C9	60
1	49	0xC5C584C5	59
1	48	0xC1C180C1	60
1	47	0xBDBD80BD	59
1	46	0xB9B97CB9	60
1	45	0xB5B57CB5	59
1	44	0xB1B178B1	60
1	43	0xADAD78AD	59
1	42	0xA9A974A9	59
1	41	0xA5A570A5	60
1	40	0xA1A16CA1	61
1	39	0x9D9D6C9D	59
1	38	0x99996899	60
1	37	0x95956895	59
1	36	0x91916491	60
1	35	0x8D8D608D	61
1	34	0x89896089	59
1	33	0x85855C85	60
1	32	0x81815C81	58
1	31	0x7D7D587D	59
1	30	0x79795479	59
1	29	0x75755075	61
1	28	0x71715071	58
1	27	0x6D6D4C6D	59

1	26	0x69694869	61
1	25	0x65654865	58
1	24	0x61614461	59
1	23	0x5D5D405D	61
1	22	0x59593C59	62
1	21	0x55553C55	59
1	20	0x51513851	61
1	19	0x4D4D344D	62
1	18	0x49493449	57
1	17	0x45453045	60
1	16	0x41412C41	61
1	15	0x3D3D2C3D	56
1	14	0x39392839	58
1	13	0x35352435	60
1	12	0x31312031	62
1	11	0x2D2D202D	56
1	10	0x29291C29	57
1	9	0x25251825	62
1	8	0x21211421	63
1	7	0x1D1D141D	54
1	6	0x19191019	57
1	5	0x15150C15	62
1	4	0x11110811	65
1	3	0xD0D040D	64
1	2	0x9090009	57
1	1	0x5050005	43
1	0	0x1010001	32

Table 52: Reference TxPower setting CHAN9, Coarse 2

Coarse	Fine	TxPower Setting	Power : BASE- PHR 0.1dB steps
2	63	0xFEFEC0FE	60
2	62	0xFAFABCFA	60
2	61	0xF6F6BCF6	59
2	60	0xF2F2B8F2	60
2	59	0xEEEEB8EE	60
2	58	0xEAEAB4EA	60
2	57	0xE6E6B4E6	59
2	56	0xE2E2B0E2	60
2	55	0xDEDEACDE	60
2	54	0xDADAACDA	59
2	53	0xD6D6A8D6	60

2	52	0xD2D2A8D2	59
2	51	0xCECEA4CE	60
2	50	0xCACAA0CA	60
2	49	0xC6C6A0C6	59
2	48	0xC2C29CC2	60
2	47	0xBEBE9CBE	59
2	46	0xBABA94BA	60
2	45	0xB6B694B6	59
2	44	0xB2B290B2	59
2	43	0xAEAE8CAE	60
2	42	0xAAAA88AA	60
2	41	0xA6A684A6	60
2	40	0xA2A280A2	60
2	39	0x9E9E7C9E	61
2	38	0x9A9A7C9A	59
2	37	0x96967896	61
2	36	0x92927892	59
2	35	0x8E8E748E	59
2	34	0x8A8A708A	60
2	33	0x86866C86	61
2	32	0x82826C82	59
2	31	0x7E7E687E	59
2	30	0x7A7A647A	59
2	29	0x76766076	60
2	28	0x72725C72	60
2	27	0x6E6E586E	61
2	26	0x6A6A546A	61
2	25	0x66665466	58
2	24	0x62625062	59
2	23	0x5E5E4C5E	60
2	22	0x5A5A485A	61
2	21	0x56564856	58
2	20	0x52524452	58
2	19	0x4E4E404E	59
2	18	0x4A4A3C4A	59
2	17	0x46463846	61
2	16	0x42423442	61
2	15	0x3E3E303E	63
2	14	0x3A3A303A	57
2	13	0x36362C36	58
2	12	0x32322832	59
2	11	0x2E2E242E	60
2	10	0x2A2A202A	61
2	9	0x26261C26	63

2	8	0x22221C22	54
2	7	0x1E1E181E	57
2	6	0x1A1A141A	55
2	5	0x16161016	60
2	4	0x12120C12	60
2	3	0xE0E080E	63
2	2	0xA0A040A	59
2	1	0x6060006	56
2	0	0x2020002	45

Note: Coarse setting 3 should not be used when configuring the device in channel 9.

Tables for channel 5:

Table 53: Reference TxPower setting CHAN5, Coarse 0

Coarse	Fine	TxPower Setting	Power : BASE- PHR 0.1dB steps
0	63	0xFCFC5CFC	61
0	62	0xF8F85CF8	60
0	61	0xF4F45CF4	59
0	60	0xF0F058F0	61
0	59	0xECEC58EC	60
0	58	0xE8E858E8	59
0	57	0xE4E454E4	61
0	56	0xE0E054E0	60
0	55	0xDCDC54DC	59
0	54	0xD8D854D8	58
0	53	0xD4D450D4	61
0	52	0xD0D050D0	60
0	51	0xCCCC50CC	59
0	50	0xC8C850C8	58
0	49	0xC4C44CC4	61
0	48	0xC0C04CC0	60
0	47	0xBCBC4CBC	59
0	46	0xB8B84CB8	58
0	45	0xB4B448B4	61
0	44	0xB0B048B0	60
0	43	0xACAC48AC	59
0	42	0xA8A848A8	58
0	41	0xA4A444A4	61

0	40	0xA0A044A0	59
0	39	0x9C9C449C	58
0	38	0x98984098	61
0	37	0x94944094	60
0	36	0x90904090	58
0	35	0x8C8C3C8C	62
0	34	0x88883C88	60
0	33	0x84843C84	58
0	32	0x80803880	62
0	31	0x7C7C387C	59
0	30	0x78783478	61
0	29	0x74743474	59
0	28	0x70703070	63
0	27	0x6C6C306C	61
0	26	0x68683068	58
0	25	0x64642C64	62
0	24	0x60602C60	59
0	23	0x5C5C2C5C	56
0	22	0x58582858	61
0	21	0x54542854	58
0	20	0x50502450	61
0	19	0x4C4C244C	57
0	18	0x48482048	63
0	17	0x44442044	59
0	16	0x40401C40	64
0	15	0x3C3C1C3C	59
0	14	0x38381838	66
0	13	0x34341834	61
0	12	0x30301830	54
0	11	0x2C2C142C	60
0	10	0x28281428	52
0	9	0x24241024	63
0	8	0x20201020	53
0	7	0x1C1C0C1C	63
0	6	0x18180C18	50
0	5	0x14140814	66
0	4	0x10100810	48
0	3	0xC0C040C	57
0	2	0x8080008	61
0	1	0x4040004	32
0	0	0x0000000	0

Table 54: Reference TxPower setting CHAN5, Coarse 1

Coarse	Fine	TxPower Setting	Power : BASE- PHR 0.1dB steps
1	63	0xFDFD94FD	60
1	62	0xF9F994F9	59
1	61	0xF5F590F5	60
1	60	0xF1F18CF1	60
1	59	0xEDED8CED	59
1	58	0xE9E988E9	60
1	57	0xE5E588E5	59
1	56	0xE1E184E1	60
1	55	0xDDDD84DD	59
1	54	0xD9D980D9	60
1	53	0xD5D580D5	59
1	52	0xD1D17CD1	61
1	51	0xCDCD7CCD	60
1	50	0xC9C97CC9	59
1	49	0xC5C578C5	61
1	48	0xC1C178C1	60
1	47	0xBDBD78BD	59
1	46	0xB9B974B9	60
1	45	0xB5B574B5	59
1	44	0xB1B170B1	61
1	43	0xADAD70AD	60
1	42	0xA9A970A9	59
1	41	0xA5A56CA5	60
1	40	0xA1A168A1	61
1	39	0x9D9D689D	60
1	38	0x99996499	60
1	37	0x95956495	59
1	36	0x91916091	60
1	35	0x8D8D608D	59
1	34	0x89895C89	60
1	33	0x85855885	61
1	32	0x81815881	59
1	31	0x7D7D547D	59
1	30	0x79795079	60
1	29	0x75755075	58
1	28	0x71714C71	59
1	27	0x6D6D486D	61
1	26	0x69694869	58
1	25	0x65654465	60

1	24	0x61614061	62
1	23	0x5D5D405D	59
1	22	0x59593C59	61
1	21	0x55553C55	58
1	20	0x51513851	60
1	19	0x4D4D344D	61
1	18	0x49493449	57
1	17	0x45453045	60
1	16	0x41412C41	61
1	15	0x3D3D2C3D	56
1	14	0x39392839	58
1	13	0x35352435	60
1	12	0x31312031	63
1	11	0x2D2D202D	57
1	10	0x29291C29	59
1	9	0x25251825	65
1	8	0x21211821	55
1	7	0x1D1D141D	57
1	6	0x19191019	62
1	5	0x15151015	50
1	4	0x11110C11	52
1	3	0xD0D080D	60
1	2	0x9090409	61
1	1	0x5050005	64
1	0	0x1010001	32

Table 55: Reference TxPower setting CHAN5, Coarse 2

Coarse	Fine	TxPower Setting	Power : BASE- PHR 0.1dB steps
2	63	0xFEFEC0FE	60
2	62	0xFAFABCFA	60
2	61	0xF6F6B8F6	60
2	60	0xF2F2B4F2	60
2	59	0xEEEEB0EE	60
2	58	0xEAEAACEA	60
2	57	0xE6E6A8E6	60
2	56	0xE2E2A4E2	60
2	55	0xDEDEA4DE	59
2	54	0xDADAA0DA	60
2	53	0xD6D69CD6	60
2	52	0xD2D29CD2	59

2	51	0xCECE98CE	60
2	50	0xCACA94CA	60
2	49	0xC6C694C6	59
2	48	0xC2C290C2	60
2	47	0xBEBE8CBE	60
2	46	0xBABA8CBA	59
2	45	0xB6B688B6	60
2	44	0xB2B288B2	59
2	43	0xAEAE84AE	60
2	42	0xAAAA84AA	59
2	41	0xA6A680A6	60
2	40	0xA2A27CA2	61
2	39	0x9E9E7C9E	60
2	38	0x9A9A789A	61
2	37	0x96967896	60
2	36	0x92927492	60
2	35	0x8E8E748E	59
2	34	0x8A8A708A	60
2	33	0x86866C86	60
2	32	0x82826882	61
2	31	0x7E7E647E	60
2	30	0x7A7A607A	60
2	29	0x76765C76	61
2	28	0x72725872	61
2	27	0x6E6E586E	59
2	26	0x6A6A546A	59
2	25	0x66665066	61
2	24	0x62625062	58
2	23	0x5E5E4C5E	59
2	22	0x5A5A485A	60
2	21	0x56564456	61
2	20	0x52524052	62
2	19	0x4E4E404E	58
2	18	0x4A4A3C4A	59
2	17	0x46463846	61
2	16	0x42423442	61
2	15	0x3E3E303E	63
2	14	0x3A3A303A	57
2	13	0x36362C36	58
2	12	0x32322832	59
2	11	0x2E2E242E	60
2	10	0x2A2A202A	62
2	9	0x26262026	55
2	8	0x22221C22	55

2	7	0x1E1E181E	58
2	6	0x1A1A141A	57
2	5	0x16161016	63
2	4	0x12120C12	65
2	3	0xE0E080E	73
2	2	0xA0A040A	74
2	1	0x6060406	45
2	0	0x2020002	45

Table 56: Reference TxPower setting CHAN5, Coarse 3

Coarse	Fine	TxPower Setting	Power : BASE- PHR 0.1dB steps
3	63	0xFFFFD4FF	60
3	62	0xFBFBD0FB	60
3	61	0xF7F7CCF7	60
3	60	0xF3F3C8F3	60
3	59	0xEFEFC4EF	60
3	58	0xEBEBC0EB	60
3	57	0xE7E7BCE7	60
3	56	0xE3E3B8E3	60
3	55	0xDFDFB4DF	60
3	54	0xDBDBB0DB	60
3	53	0xD7D7ACD7	60
3	52	0xD3D3A8D3	60
3	51	0xCFCFA4CF	60
3	50	0xCBCBA4CB	59
3	49	0xC7C7A0C7	60
3	48	0xC3C39CC3	60
3	47	0xBFBF9CBF	59
3	46	0xBBBB98BB	60
3	45	0xB7B794B7	60
3	44	0xB3B394B3	59
3	43	0xAFAF90AF	60
3	42	0xABAB8CAB	60
3	41	0xA7A78CA7	59
3	40	0xA3A388A3	59
3	39	0x9F9F849F	60
3	38	0x9B9B809B	60
3	37	0x97978097	59

3	36	0x93937C93	60
3	35	0x8F8F7C8F	59
3	34	0x8B8B788B	60
3	33	0x87877487	60
3	32	0x83837083	61
3	31	0x7F7F6C7F	60
3	30	0x7B7B687B	60
3	29	0x77776477	60
3	28	0x73736073	60
3	27	0x6F6F5C6F	61
3	26	0x6B6B586B	61
3	25	0x67675867	59
3	24	0x63635463	59
3	23	0x5F5F505F	60
3	22	0x5B5B4C5B	61
3	21	0x57574C57	58
3	20	0x53534853	58
3	19	0x4F4F444F	58
3	18	0x4B4B404B	59
3	17	0x47473C47	60
3	16	0x43433843	61
3	15	0x3F3F343F	61
3	14	0x3B3B303B	62
3	13	0x37373037	57
3	12	0x33332C33	56
3	11	0x2F2F282F	58
3	10	0x2B2B242B	57
3	9	0x27272027	60
3	8	0x23231C23	60
3	7	0x1F1F181F	63
3	6	0x1B1B141B	62
3	5	0x17171017	68
3	4	0x13131013	50
3	3	0xF0F0C0F	50
3	2	0xB0B080B	50
3	1	0x7070407	50
3	0	0x3030003	50

16 Document history

Table 57: Document history

Revision	Date	Description
A	August 2022	First Release

17 Further Information

Qorvo develops semiconductors solutions, software, modules, reference designs - that enable real-time, ultra-accurate, ultra-reliable local area micro-location services. Qorvo's technology enables an entirely new class of easy to implement, highly secure, intelligent location functionality and services for IoT and smart consumer products and applications.

For further information on this or any other Qorvo product, please refer to app note on Qorvo website.