MCR20A 2.4 GHz Low-Power Transceiver Reference Manual

Supports: MCR20AVHM

Document Number: MCR20ARM
Rev. 3, July 2016

Contents

Section number **Title** **Page**

Chapter 1
Introduction to MCR20A

Chapter 2
Signal Multiplexing and Signal Descriptions

Chapter 3
System Considerations

Chapter 4
Modem: Modes of Operation

Chapter 5
Modem: Advanced Security Module

Chapter 6
Modem: Interrupts

Chapter 7
Modem: Timer Information

Chapter 8
Modem SPI Interface

Chapter 9
Modem: SPI Register Descriptions

Chapter 1
Introduction to MCR20A

1.1 Introduction

The MCR20A transceiver is a 2.4 GHz Industrial, Scientific and Medical (ISM) and Medical Body Area Network (MBAN) transceiver intended for the IEEE® 802.15.4 Standard. The MCR20A device is a standalone transceiver that is normally combined with a software stack and a Kinetis K series, M series or other microcontroller (MCU) to implement an IEEE 802.15.4 Standard platform solution.

The MCR20A transceiver contains a complete 802.15.4 physical layer (PHY) modem designed for the IEEE® 802.15.4 Standard that operates in the 2.4 GHz ISM frequency band and supports 2.36 to 2.4 GHz Medical Band (MBAN) frequencies. The transceiver includes antenna diversity, 1mW nominal output power, hardware acceleration for dual PAN modes, integrated transmit/receive switch, on-board power supply regulation, and full spread-spectrum encoding and decoding. Additionally, the transceiver includes a PA with internal voltage controlled oscillator (VCO), integrated transmit/receive switch, on-board power supply regulation.

The MCR20A transceiver supports peer-to-peer, star, and mesh networking and when combined with an appropriate MCU, the MCR20A transceiver provides a cost-effective solution for short-range data links and networks. Interface with the MCU is accomplished using a four wire serial peripheral interface (SPI) connection and an interrupt request output that allows for the use of a variety of processors. The software and processor can be scaled to fit applications ranging from simple point-to-point systems through complete mesh networking. The MCR20A transceiver provides the IEEE 802.15.4 Standard PHY/MAC for use with the Kinetis K20 or Cortex M0 family of MCUs.

This table lists the MCR20A device ordering, temperature range, and package information.

Table 1-1. Ordering Information

Device	Operating Temp Range (TA)	Package
MCR20AVHM(R)	-40° to 105° C	MLGA-32 (R: tape and reel)

Target markets include, but are not limited, to the following:

- Smart Energy
  - Meter
  - ESI (Energy Service Interface)
  - IHD (In Home Display)
  - Gateway
  - Appliance
  - PHEV (Plug-in Hybrid Electric Vehicle)
- Building Control and Home Automation
  - Lighting
  - HVAC
  - Security
- Medical / Personal Health Care
  - Patient Monitoring
  - Institutional Care
- Industrial Control (3rd party stacks - (Low PAN, ISA100, Wireless HART)

1.2 Block Diagram

This figure shows a simplified block diagram of the MCR20A transceiver, which is an 802.15.4 Standard compatible transceiver that provides functions required in the physical layer (PHY) and media access control (MAC) specifications.

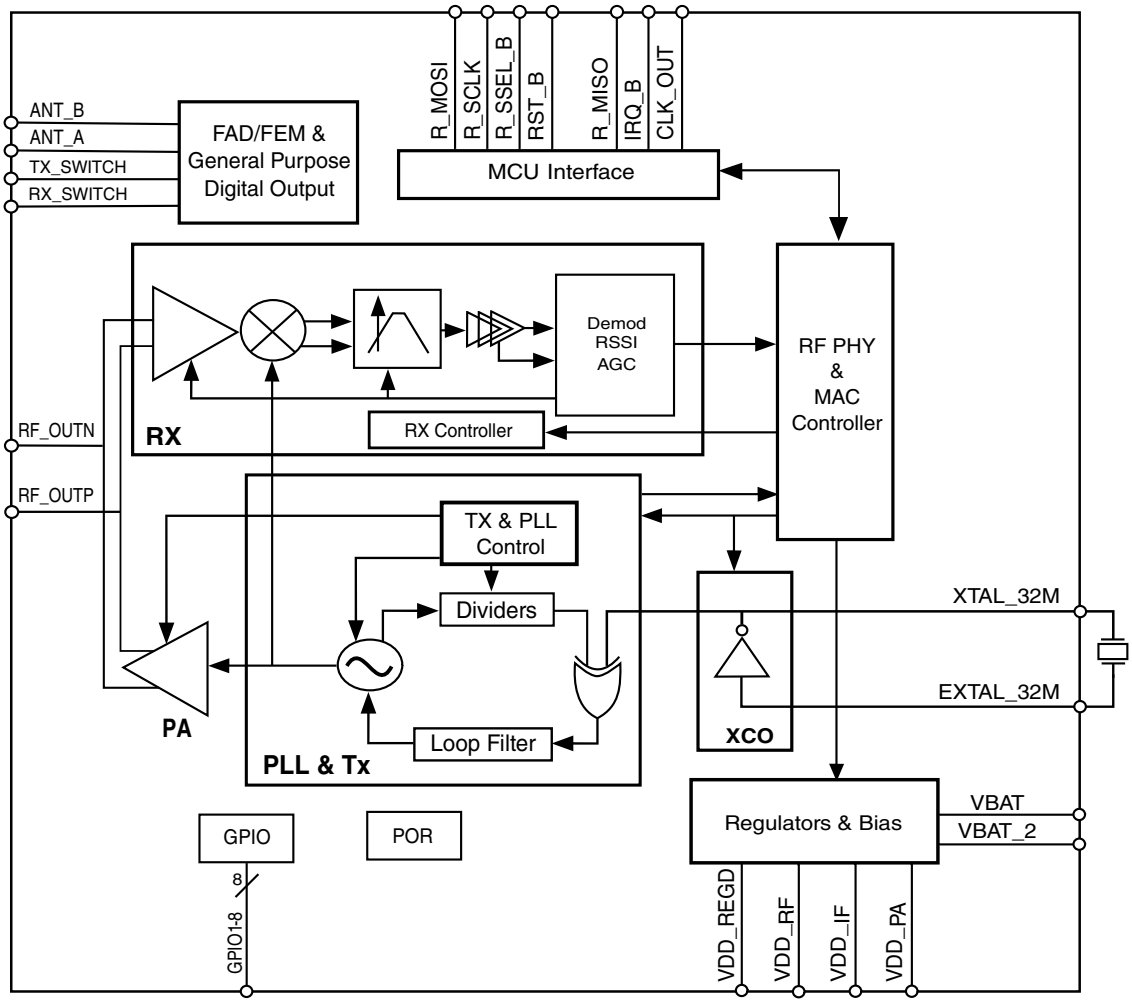


Figure 1-1. Modem Simplified Block Diagram

The modem is used in concert with an MCU. Interface between the devices is accomplished through a 4-wire SPI port and interrupt request line. The media access control (MAC), drivers, and network and application software (as required) reside on the host processor.

1.3 Modem Features Summary

The transceiver has the following features:

- Fully compliant IEEE 802.15.4 Standard 2006 transceiver supports 250 kbps O-QPSK data in 5.0 MHz channels and full spread-spectrum encode and decode, and also extends radio operation to the 2.36 GHz to 2.40 GHz Medical Band (MBAN) frequencies with IEEE 802.15.4j channel, spacing and modulation requirements.

- 2.4GHz frequency band of operation (ISM).
- 250kbps data rate with O-QPSK modulation in 5.0 MHz channels with direct sequence spread spectrum (DSSS) encode and decode.
- Operates on one of 16 selectable ISM channels per IEEE 802.15.4 specification.
- Programmable output power
- Supports 2.36 GHz to 2.40 GHz Medical Band (MBAN) frequencies with IEEE 802.15.4j channel, spacing and modulation requirements.

- Small RF foot print
  - Differential input/output port used with external balun for single port operation.
  - Supports antenna diversity operation with external front end (FE).
  - Low external component count.
- Hardware acceleration for IEEE® 802.15.4 Standard
  - Complete 802.15.4 onboard modem
  - IEEE 802.15.4 Standard 2006 packet processor/sequencer with receiver frame filtering
  - Random number generator
  - Support for dual PAN ID mode
  - Internal event timer block with four comparators to assist sequencer and provide timer capability
- 32 MHz crystal reference oscillator with onboard trim capability to supplement external load capacitors
- Programmable frequency clock output (CLK_OUT) for use by MCU
- SPI Command Channel interface slave port with burst mode operation
- Interrupt request output (IRQ) - provides interrupt request capability to MCU
- 128-byte RAM data buffer to store 802.15.4 packet contents for transceiver sequences
- Eight (8) software programmable GPIOs
- Low power operational modes with single SPI command device wake-up (SPI communication is enabled in LP mode)

- 1.8 V to 3.6 V operating voltage with on chip voltage regulators
- -40C to +105C temperature range
- RoHS compliant, 5 mm x 5 mm, 32-pin, MLGA package

1.4 RF Interface and Usage

The modem RF interface provides a bidirectional, differential port that connects directly to a balun. The balun connects directly to a single-ended antenna and converts that interface to a fully differential, bidirectional, on-chip interface with transmit/receive switch, LNA, and complementary PA outputs. This combination allows for a small footprint and low cost RF solution.

In additional the modem provides dedicated output signals that can be used to control external RF components. These outputs are hardware switched and also support antenna diversity.

1.5 Radio Architecture

The radio structure is built upon the IEEE 802.15.4 Standard packet structure.

1.5.1 Packet Structure

The following figure shows the packet structure.

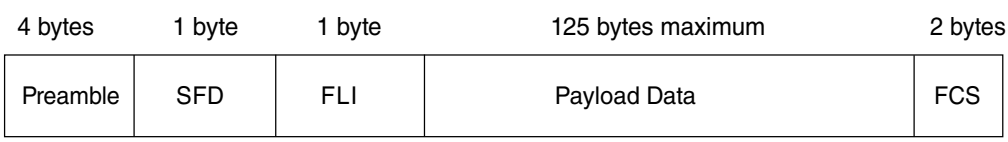


Figure 1-2. MCR20A Transceiver Packet Structure

1.5.2 Receive Path Description

The receive path operates in duplex with the transmit mode having an additional feature to operate in a low power run state that can also be considered as a partial power down mode. Architecture is Near Zero IF (NZIF) having front end amplification, one (1) mixed

signal down conversion to IF that is filtered, demodulated and digitally processed. The RF Front End (FE) is differential and shares the same off chip matching network with the transmit path.

1.5.3 Transmit Path Description

The modem transmits OQPSK modulation having power and channel selection adjustment per user application. After the channel of operation is determined, coarse and fine tuning is executed within the Frac-N PLL to engage signal lock. After signal lock is established, the modulated buffered signal is then routed to a multi-stage amplifier for transmission. The PA differential outputs share the pins with the front end.

1.6 IEEE 802.15.4 Acceleration Hardware

The 802.15.4 transceiver has several hardware features that reduce the software stack size, off-load functions from the CPU, and improve performance:

- Fully supports 2003 & 2006 versions of the IEEE 802.15 Standard.
- Supports slotted and unslotted modes
- Supports beacon enabled and non-beacon enabled networks
- Onboard 128-byte packet data buffering
- Random number generator
- 802.15.4 Sequence support
  - RX (conditionally followed by TXAck)
  - TX
  - CCA (used for CCA and ED cycles)
  - Tx/Rx (Tx followed by unconditional Rx or RCACK)
  - Continuous CCA
- 802.15.4 Receiver Frame filtering.

1.7 Advanced Security Module (ASM) Overview

The ASM engine encrypts using the Advanced Encryption Standard (AES). It can perform "Counter" (CTR), Cipher Block Chaining (CBC) and plain AES mode encryption. The combination of CTR and CBC modes of encryption is known as CCM mode encryption. CCM is short for Counter with CBC-MAC. CCM is a generic authenticate and encrypt block cipher mode. CCM is only defined for use with 128 bit block ciphers, such as AES.

The ASM has the following features:

- CTR encryption in 11 bus clock cycles.
- CBC encryption in 11 bus clock cycles
- AES encryption in 11 bus clock cycles.
- Encrypts 128 bits as a unit.

The ASM is designed to be loaded with data and then started with a self-clearing "start" bit. Sixteen 8 bit registers of a key plus sixteen 8 bit registers of a counter plus sixteen 8 bit registers of text are necessary for "Counter" mode encryption. Cipher Block Chaining (CBC) mode needs only a key field and a text field programmed. Typically, only the text fields and counter fields need to be continuously written since the key field won't change.

The module has a built in self test that must be initiated by the software to make the module usable. Until this test is run and passes the ASM module is disabled.

A simple block diagram of the ASM module is shown in the following figure.

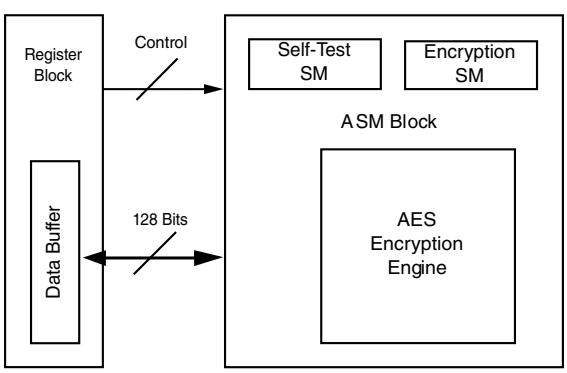


Figure 1-3. ASM Block Diagram

1.8 MCU Interface with SPI Overview

The following figure illustrates the microcontroller interface with the modem for the SiP. The typical required signals are:

- 4-wire SPI port - slave mode only
  - Maximum bitrate is 16 MHz for writes and 9 MHz for reads
  - Clock phase and polarity - CPHA=0 and CPOL=0
  - MSB first shifting
  - Supports Register Access and Packet Buffer accesses in bursts of byte transfers
  - Most registers and Packet Buffer accessible with crystal "on" or "off"
- Interrupt request output - active low
- Device asynchronous hardware reset RST_B - active low

The SPI interface provides communication between the MCU and the modem's register set and Packet Buffer. The modem SPI is a slave-only interface; the MCU must drive R_SSEL_B, R_SCLK and R_MOSI. Write and read access to both Direct and Indirect registers is supported, and transfer length can be single-byte or bursts of unlimited length. Write and read access to the Packet buffer can also be single-byte or a burst mode of unlimited length.

The SPI interface is asynchronous to the rest of the device. No relationship between R_SCLK and the modem's internal oscillator is assumed. All synchronization of the SPI interface to modem takes place inside the SPI module and is done for both register writes and reads.

The SPI is capable of operation in all power modes except reset. Operation in the Hibernate state means radio’s crystal oscillator is disabled; effectively no XTAL connected in hibernate mode. Most radio registers, direct and indirect, can be accessed, both read and write, during Hibernate; and includes the complete Packet Buffer. All GPIO-related registers are accessible in Hibernate. In this state minimal power consumption will be realized especially during the register-initialization phase.

The SPI design features a compact, single-byte control word reducing SPI access latency to a minimum. Most SPI access types require only a single-byte control word with the address embedded in the control word. During control word transfer (the first byte of any

SPI access), the contents of the IRQSTS1 register (highest-priority status register) are are always shifted out so that the MCU gets access to IRQSTS1 with the minimum possible latency on every SPI access.

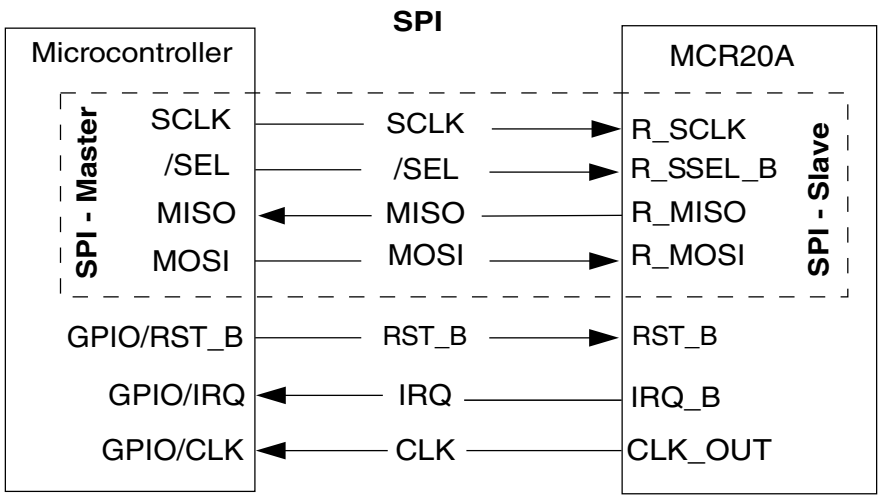


Figure 1-4. Microcontroller to modem interface block diagram

1.8.1 Transceiver Control Overview

The transceiver is controlled by a bank of registers (Direct Register Space) accessed via the SPI interface. The transmit and receive packet data (stored in a 128-byte buffer) is also accessed via the SPI. The onboard registers provide control and status for the entire device.

1.8.1.1 Interrupt Request Overview

The modem has up to 13 individual sources of interrupt request to the MCU. These are all capable of individual control, and are logically OR-combined to drive a single, active low, interrupt request pin (IRQ_B) to the external MCU.

- The IRQ_B pin is configured as actively-driven high by the modem.
- Each interrupt source has its own interrupt status bit in Direct Register space.
- Each interrupt can be individually controlled by an interrupt mask - The IRQ is issued when the mask is cleared to 0.
- There is also a global interrupt mask, TRCV_MSK, which can enable/disable all IRQ_B assertions by programming a single masking bit.

- All status bits use a write-1-to-clear protocol - interrupt status bits are not affected by reads.
- IRQ_B will remain asserted until all active interrupt sources are cleared or masked.

1.8.1.2 Event Timer Overview

The modem features a 24-bit Event Timer that can be used in conjunction with the sequencer to provide protocol control as well as timing interrupts. The Event Timer consists of a continuously running counter and four (4) separate 24-bit comparators:

- The Event Timer counter runs at the 802.15.4 bit rate of 250 kHz (programmable).
- Each comparator has an individual interrupt request capability - the compare status is set when there is a match between the comparator and the timer counter. Each status can be enabled to generate an IRQ.
- In addition, a separate 16-bit T2PRIMECMP comparator is provided, which uses only the *lower 16 bits* of Event Timer, rather than require a full 24-bit compare.

1.9 Clock Output, RF Control, and GPIO Summary

The modem provides a set of I/O pins useful for supplying a system clock to the MCU, controlling external RF LNA/PA or antenna diversity circuitry and GPIO. The following sections discuss these options.

1.9.1 CLK_OUT Reference

The CLK_OUT digital output can be enabled to drive the system clock to the MCU. This provides a highly accurate clock source based on the transceiver reference oscillator. The clock is programmable over a wide range of frequencies divided down from its 32 MHz reference.

1.9.2 RF Control Signals

The modem provides four dedicated signals for control of external RF components. These signals designated as ANT_A, ANT_B, RX_SWITCH, and TX_SWITCH can be enabled to control external amplifiers, antenna switches, and other modules. When enabled they are switched via an internal hardware state machine. Typical uses include:

- Antenna diversity
- External PA
- External LNA
- T/R switching

1.9.3 Antenna Diversity

To improve the reliability of RF connectivity to long range applications, Antenna Diversity feature is supported without using the MCU through use of four dedicated control pins by direct register antenna selection. The digital regulator supplies bias to analog switches for control of external PA/LNA. These switches are programmable to sink and source two levels of current (2-3 mA and 10 mA) or can operate in a high impedance mode.

1.9.4 General Purpose Input Output (GPIO)

In addition eight (8) GPIO are provided for general use. Features for these pins include:

- Programmable output drive strength
- Programmable output slew rate
- Hi-Z mode
- Programmable as outputs or inputs (default)
- No IRQ capability

1.10 Modem Operational Modes

The modem has six operating modes which include:

- Reset / Power-down
- Low Power (LP) / Hibernate
- Doze (low power with reference oscillator active)
- Idle
- Receive
- Transmit

The following table describes these modes:

Table 1-2. Modem Mode Definitions and Transition Times

Mode	Definition	Transition Time To or From Idle
Off	All modem functions Off, Leakage only. $\overline{RST}$ asserted. Digital outputs are tri-stated including IRQ	500 µs
Hibernate	Crystal Reference Oscillator Off. Modem responds to SPI activity.	250 µs
Doze	Crystal reference oscillator ON but CLK_OUT output available only if selected. Digital regulator in Low Power mode.	< 1 µs[1]
Reset	Crystal reference oscillator ON, enable CLK_OUT output at 4 MHz and 32.787 kHz.	376 µs
Receive	Crystal reference oscillator ON. Receiver ON.	144 µs
Transmit	Crystal reference oscillator ON. Transmitter ON.	144 µs

1. At 9 MHz, the doze to idle transistion time will be less than 2 SPI writes (1.8 µs).

1.11 External PA and LNA

The modem supports features to add either an external PA, LNA or RF switch which can extend the range or add antenna diversity to the target application. The following hardware features aid in the configuration of an FEM:

- Four dedicated programmable pins to sink and source 1mA, 2mA, 4mA and 8mA currents to control FEM (Front End Module) features such as a PA, LNA and RF switches for antenna diversity, etc.
- Balun used to optimize performance and provide a differential RX/TX output to single ended feature. Modem RX/TX outputs are differential “I” and “Q” and can be utilized in that format if desired.

Chapter 2
Signal Multiplexing and Signal Descriptions

2.1 Pin assignments

This figure shows the MCR20A transceiver's package pin assignment.

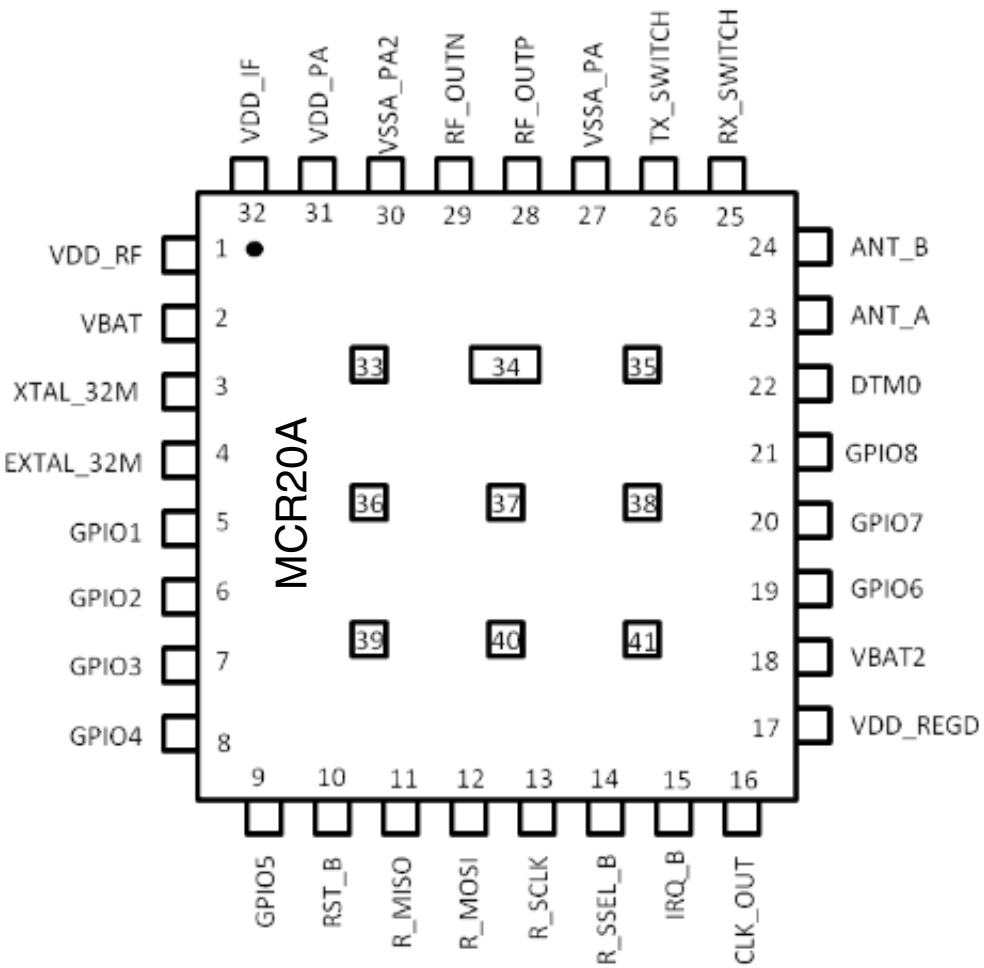


Figure 2-1. Pin assignment

2.2 Pin function table

Table 2-1. Pin function

Pin number	Pin name	Type	Function	Description
1	VDD_RF	Analog Power Output	Analog Voltage	Analog 1.8 Vdc
2	VBAT	Power Input	Battery Voltage	Connect to system VDD supply
3	XTAL_32M	Analog Output	RF	32 MHz reference oscillator output
4	EXTAL_32M	Analog Input	RF	32 MHz references oscillator input
5	GPIO1	Digital Input/Output	General-Purpose IO	GPIO
6	GPIO2	Digital Input/Output	General-Purpose IO	GPIO
7	GPIO3	Digital Input/Output	General-Purpose IO	GPIO
8	GPIO4	Digital Input/Output	General-Purpose IO	GPIO
9	GPIO5	Digital Input/Output	General-Purpose IO	GPIO or CLK_OUT default state select
10	RST_B	Digital Input/Output	Digital	Device asynchronous hardware reset. Active low.
11	R_MISO	Digital Input/Output	Digital	SPI MISO
12	R_MOSI	Digital Input/Output	Digital	SPI MOSI
13	R_SCLK	Digital Input/Output	Digital	SPI clock
14	R_SSEL_B	Digital Input/Output	Digital	SPI slave select
15	IRQ_B	Digital Input/Output	Digital	Interrupt command signal
16	CLK_OUT	Digital Output	RF	Programmable clock source
17	VDD_REGD	Digital Power Ref	Digital Voltage	Digital 1.8 Vdc ref. Decouple to ground.
18	VBAT2	Power Input	Battery Voltage	Connect to system VDD supply.
19	GPIO6	Digital Input/Output	General-Purpose IO	GPIO
20	GPIO7	Digital Input/Output	General-Purpose IO	GPIO
21	GPIO8	Digital Input/Output	General-Purpose IO	GPIO
22	DTM0	—	Factory Test	Do not connect.
23	ANT_A	Digital Input/Output	Antenna Diversity	Programmable sink and source current output with selectable high impedance state.
24	ANT_B	Digital Input/Output	Antenna Diversity	Programmable sink and source current output with selectable high impedance state.
25	RX_SWITCH	Digital Input/Output	Control Switch	Programmable sink and source current output with selectable high impedance state.
26	TX_SWITCH	Digital Input/Output	Control Switch	Programmable sink and source current output with selectable high impedance state.

Table continues on the next page...

Table 2-1. Pin function (continued)

Pin number	Pin name	Type	Function	Description
27	VSSA_PA	—	Gnd	RF ground
28	RF_OUTP	RFInput/Output	RF	Bidirectional RF input/output positive
29	RF_OUTN	RFInput/Output	RF	Bidirectional RF input/output negative
30	VSSA_PA2	—	Gnd	RFground
31	VDD_PA	Analog Power Input	Analog Voltage	Analog 1.8 Vdc input
32	VDD_IF	Analog Power Input	Analog Voltage	Analog 1.8 Vdc input
33	GND_RF	—	—	Connect to RF ground
34	GND_PA	—	—	Connect to RF ground
35	GND_RF	—	—	Connect to RF ground
36	—	Factory test	Reserved	Do not connect
37	—	Factory test	Reserved	Do not connect
38	—	Factory test	Reserved	Do not connect
39	—	Factory test	Reserved	Do not connect
40	—	Factory test	Reserved	Do not connect
41	—	Factory test	Reserved	Do not connect

Chapter 3
System Considerations

3.1 Introduction

Communication to the modem function is through the common SPI bus, the MCU interrupt request, and several MCU GPIO lines. Primary interface with the modem is through the SPI command structure that allows reading/writing modem registers and provides initialization of parameters, reading of status, and control of modem operation. The modem can ask for real time response through the interrupt request structure, and four GPIO signals allow control of the modem reset and monitoring of some real time status.

This chapter presents information addressing application and operation of the node from a system level. The areas considered here are also covered in greater detail in the following sections of the book. The book is organized such that the first three chapters present the top-level view of the MCR20A device and the following chapters present individual functions in detailed descriptions.

3.2 Power Connections

The modem power connections are listed in the following table.

Table 3-1. Power Pin Descriptions

Pin Name	Type	Description	Functionality
VDD_RF	Power Output	Regulator output for RF circuitry	Decouple to ground.
VBAT	Power Input	Main voltage supply	Decouple to ground.
VDD_REGD	Power Output	Regulated output supply for digital circuitry	Decouple to ground
VBAT2	Power Input	Main voltage supply	Decouple to ground.
GND_PA	Power Input	PA supply ground	Connect to ground
GND_RF	Power Input	RF supply ground	Connect to ground
VDD_PA	Power Input	Regulated supply for PA	Decouple to ground

Table continues on the next page...

Table 3-1. Power Pin Descriptions (continued)

Pin Name	Type	Description	Functionality
VDD_IF	Power Input	Regulated supply for IF	Decouple to ground
VSSA_PA	Power Input	Common VSS	Connect to ground
VSSA_PA2	Power Input	Common VSS	Connect to ground

When designing power to the device, the following points need to be considered for the modem:

- There are two modem primary power inputs, which include VBAT for analog power and VBAT2 for digital circuitry.
- For logic level compatibility between the modem and the system MCU, VBAT, and VBAT2 must be connected with the MCU to a common source supply of 1.8–3.6 VDC. It is not recommended to supply the MCU below 1.8V.
- VDD_PA is the supply of the internal power amplifier and is powered by the VDD_RF regulator output via the analog regulator.
- VDD_RF and VDD_IF are provided to allow bypass of the RF and IF circuitry regulated supplies. The decoupling capacitor is in the range of 220 nF and 470 nF as shown in the figure.
- The VDD_REGD supply is decoupled externally as shown in the figure. The external decoupling capacitor shall be in the range of 220 nF and 470 nF.

Power supply connections for the modem are shown in the following figure.

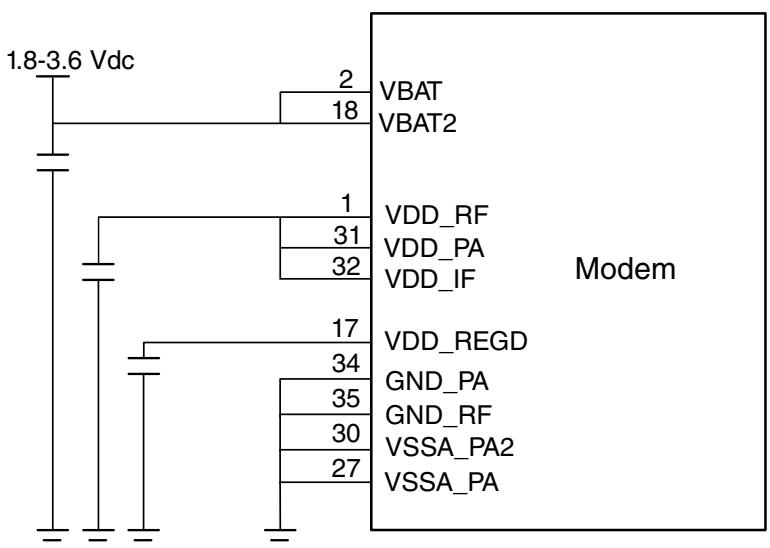


Figure 3-1. Modem Power Supply Connections

NOTE

In addition to the capacitor on the VBATs, there are two separate decoupling capacitors, one on VDD_RF, VDD_IF, and VDD_PA that are tied together, and a standalone on VDD_REGD.

3.3 Modem Reset

The modem active low reset input RST_B is recommended to be driven from an MCU GPIO pin. In the interest of lowest power, there is no external pull-up resistor on input RST_B. An MCU GPIO programmed as an output typically also has a software controlled pull-up resistor. However, it would normally not be used because the modem can be held in hardware reset by the MCU for extended periods of time. The transceiver IRQ pullup can also be disabled, and having no resistor makes for lowest power applications.

From a power-on or “cold start” condition, the MCU GPIO normally initiates as a high-impedance input with its internal pullup disabled and the IRQ pullup is enabled, which holds the modem reset input high. As part of the MCU initialization, GPIO must be programmed as an output and then driven low to reset the modem. The RST_B input is asynchronous and needs to be held low for only a short period.

- The MCU software must configure one of its GPIOs as an output and then drive it low to reset the modem.

During the modem reset procedure,

- The MCU software should also configure the I/O port driving GPIO5 to select the default modem CLK_OUT frequency (GPIO5 high = 32.787 kHz; GPIO5 low = 4 MHz).
- The modem RST_B input is asynchronous and needs to be held low for only a short period.

In the reset condition, the modem is totally powered down and no clocks are available. After RST_B is released, the modem will power up, initialize, and go to its idle condition in less than 1 millisecond, and in turn, this causes an IRQ_B interrupt request and allows CLK_OUT to start toggling. Coming out of reset, the WAKE_IRQ interrupt status bit is set and causes the assertion on the IRQ_B because the wake interrupt source is not masked by default.

Once the interrupt request is seen by the MCU, the MCU can assume the modem is alive and ready for programming via the SPI bus.

3.4 Modem Interrupt Request to MCU

The modem interrupt request IRQ_B is an active low output that is asserted when an interrupt request is pending. The signal is released to high by writing a 1 to all asserted interrupt status bits in the modem status registers via a SPI transaction.

3.5 MCR20A Transceiver Interface to MCU

The modem interacts with the host MCU through its SPI interface, interrupt request, and several status and control signals.

3.5.1 SPI Command Channel

Primary interface with the modem is through the SPI command structure that allows reading/writing modem registers and provides initialization of parameters, reading of status, and control of modem operation. The modem is a slave only and the MCU SPI must be programmed and used as a master only. Further, the SPI performance is limited by the modem constraints of 8 MHz maximum SPI clock frequency, and use of the MCU SPI must be programmed to meet the modem SPI protocol. The SPI bus connections for a Kinetis K20-512 typically are:

- MCU MOSI output drives modem R_MOSI.
- Modem R_MISO output drives MCU MISO.
- MCU SCLK output drives modem R_SCLK.
- MCU /SELoutput drives modem R_SSEL_B.

3.6 System Oscillator and Clock Considerations

3.6.1 Modem Crystal Oscillator

The modem oscillator source must always be present and an external crystal is used to implement the oscillator. The source frequency must be 32 MHz with a total accuracy of ±40 ppm or greater as required by the IEEE 802.15.4 specification.

In this figure, crystal X1 and capacitors C1 and C2 form the modem crystal oscillator circuit. An onboard feedback resistor of approximately 1 MOhm (not shown) between input EXTAL_32M and output XTAL_32M provides DC biasing for the oscillator buffer. An important parameter for the 32 MHz crystal X1 is a load capacitance of less than 9 pF. The oscillator needs to see a balanced load capacitance at each terminal of about 18 pF. As a result, the sum of the stray capacitance of the PCB, device pin (EXTAL or XTAL), and load capacitor (C1 or C2) at each terminal must equal about 18 pF. C1 and C2 are typically values of 6-9 pF. Higher values can load the crystal buffer and cause oscillator start-up problems.

The MCR20A transceiver crystal oscillator frequency can be trimmed by programming modem CLK_OUT_CTRL register to the corresponding wanted frequency and trim the frequency with the XTAL_TRIM register. The trimming procedure varies the frequency by a few Hertz per step, depending on the type of crystal. As xtal_trim[7:0] is increased, the frequency is decreased. This feature is useful for factory calibration of the crystal frequency to set the accuracy for the radio as required by the IEEE 802.15.4 specification.

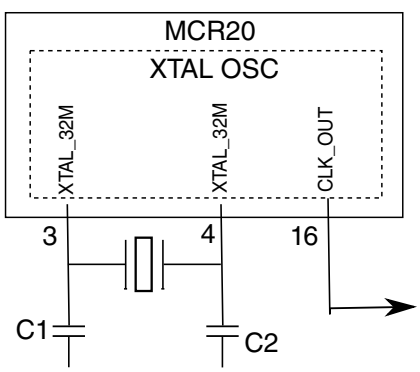


Figure 3-2. MCR20A Oscillator and External Connections

3.6.2 System Clock Configurations

Because of the multiple clock configurations in an MCU and the CLK_OUT output from the modem, there are a number of variations for system clock configurations. Key considerations for any system clock configuration are:

- The modem 32 MHz source (typically the crystal oscillator) must always be present. The crystal has special requirements and the reference frequency must meet IEEE 802.15.4 requirements
- Battery-operated application requirements for low power can impact the choices for MCU clock source

- The system clock configuration can impact system initialization procedures
- Software requirements can impact MCU processor and bus speed. The user must be aware of the performance requirements for the MCU.

3.6.3 Single System Crystal with CLK_OUT driving MCU crystal input

The single crystal (modem crystal) with CLK_OUT driving the MCU external clock input is a common configuration for low cost and excellent frequency accuracy. The CLK_OUT frequency is programmable from 32.786 kHz to 32 MHz and drives the MCU external source.

Note

For this system option to be usable, the system MCU must have an alternative (typically onboard) start-up clock.

In this configuration, clock start-up from a reset condition involves:

- MCU reset is released and MCU starts on an internal clock
- Initialization software must reset and then release reset to the modem (MCU still running on start-up clock)
- Wait for modem start-up interrupt request (less than 1 ms). CLK_OUT default is hardware selectable as either 4 MHz or 32.786 kHz defined by the state of modem GPIO5.
- Program CLK_OUT to a different frequency (if desired) as shown in the table.
- Wait for the CLK_OUT source to lock, and then switch MCU clock to external source

Additional considerations for this mode of operation include:

- If the modem is forced to the Off condition and CLK_OUT is killed, there is a 500 µs wait for the modem CLK_OUT to start from the Off condition after RST is released
- If the MCU puts the modem into Doze mode, keeping the CLK_OUT alive is a higher power, but available, option
- If an accurate period is required for longer time delays (such as a beacon period), keeping CLK alive for very long periods is an option, but would be a higher power option typically than using a separate crystal for the MCU

3.7 Modem GPIO Characteristics

The modem GPIO hardware consists of eight (8) signals total (GPIO1-GPIO8) though only GPIO1-6 are used in the SIP, and only GPIO1-2 are connected directly to SIP package pins. Immediately after reset, each GPIO has either an internal pull-up (GPIO1-5) or pull-down (GPIO6-8) enabled.

NOTE

To avoid extra current drain from floating input pins, the reset initialization routine in the application program should change the direction of unused pins to outputs (programmed low) so the pins do not float. Outputs programmed low is the preferred option for lowest power.

The functionality of the modem GPIO is controlled by programming of the modem SPI registers via the SPI interface. For information about controlling all these pins as general purpose I/O pins, see SPI Register Descriptions chapter.

3.8 MCR20A Digital Signal Properties Summary

This table summarizes digital I/O pin characteristics. These characteristics are determined by the way the common pin interfaces are hard-wired to internal circuits.

Table 3-2. MCR20A Digital Signals Properties

Pin Name	Dir	Drive Strength	Output Slew[1]	Pullup/ Pulldown[2]	Direction
IRQ_B	O	SWC	N	N	Actively Driven output by default. Can be configured as Open-drain (see IRQ_B_OD bit). Drive strength controlled by NON_GPIO_DS bit.
XTAL_32M	I	--	--	N	
EXTAL_32M	O	--	--	N	
RST_B	I	--	--	N	Pin should not be allowed to float. Must be externally actively driven, or externally pulled up to Vbatt via resistor.
CLK_OUT	O	SWC	SWC	N	Drive strength controlled by CLK_OUT_DS bit. Slew Rate controlled by CLK_OUT_SR bit. Tri-state controlled by CLK_OUT_HIZ bit. Frequency controlled by CLK_OUT_DIV[2:0] bits
R_SCLK	I	--	--	SWC	Pullup device controlled by SPI_PUL_EN bit
R_MOSI	I	--	--	SWC	Pullup device controlled by SPI_PUL_EN bit
R_MISO	O	SWC	N	N	Drive strength controlled by NON_GPIO_DS bit
R_SSEL_B	I	--	--	SWC	Pullup device controlled by SPI_PUL_EN bit
GPIO1	I/O	SWC	N	SWC	Drive strength controlled by GPIO_DS[1] bit. Pull Enable/Polarity controlled by GPIO_PUL_EN[1] and GPIO_PUL_SEL[1] bits.

Table continues on the next page...

Table 3-2. MCR20A Digital Signals Properties (continued)

Pin Name	Dir	Drive Strength	Output Slew[1]	Pullup/ Pulldown[2]	Direction
GPIO2	I/O	SWC	N	SWC	Drive strength controlled by GPIO_DS[2] bit. Pull Enable/Polarity controlled by GPIO_PUL_EN[2] and GPIO_PUL_SEL[2] bits.
GPIO3	I/O	SWC	N	SWC	Drive strength controlled by GPIO_DS[3] bit. Pull Enable/Polarity controlled by GPIO_PUL_EN[3] and GPIO_PUL_SEL[3] bits.
GPIO4	I/O	SWC	N	SWC	Drive strength controlled by GPIO_DS[4] bit. Pull Enable/Polarity controlled by GPIO_PUL_EN[4] and GPIO_PUL_SEL[4] bits.
GPIO5	I/O	SWC	N	SWC	Drive strength controlled by GPIO_DS[5] bit. Pull Enable/Polarity controlled by GPIO_PUL_EN[5] and GPIO_PUL_SEL[5] bits.
GPIO6	I/O	SWC	N	SWC	Drive strength controlled by GPIO_DS[6] bit. Pull Enable/Polarity controlled by GPIO_PUL_EN[6] and GPIO_PUL_SEL[6] bits.
GPIO7	I/O	SWC	N	SWC	Drive strength controlled by GPIO_DS[7] bit. Pull Enable/Polarity controlled by GPIO_PUL_EN[7] and GPIO_PUL_SEL[7] bits.
GPIO8	I/O	SWC	N	SWC	Drive strength controlled by GPIO_DS[8] bit. Pull Enable/Polarity controlled by GPIO_PUL_EN[8] and GPIO_PUL_SEL[8] bits.

1. Slew Rate Controlled: N=No; SWC=software-controlled
2. Internal pullup/pulldown device: N=No; SWC=software-controlled

3.9 Transceiver RF Configurations and External Connections

The MCR20A transceiver radio has features that enable a flexible and low cost RF interface:

- Programmable output power from –35 dBm to +8 dBm. This allows the user to set the power based on their specific applications minimizing current consumption while maximizing battery life.
- –102 dBm (typical) receive sensitivity — At 1% PER, 20-byte packet (well above IEEE 802.15.4 specification of –85 dBm).
- Integrated transmit/receive (T/R) switch for low cost operation — With internal PAs and LNA, the internal T/R switch allows a minimal part count radio interface using only a single balun to interface to a single-ended antenna.

3.9.1 RF Interface Pins

The following figure shows the RF interface pins and the associated analog blocks. The MCR20A radio is a differential transceiver architecture that has two (2) I/Os that support both receive and transmit functionality. Specifically, these I/Os are opposite in phase and will require off-chip matching for optimal performance. Because RF_OUTP and RF_OUTN ports are shared and duplexed, matching networks will need to accommodate both modes of operation.

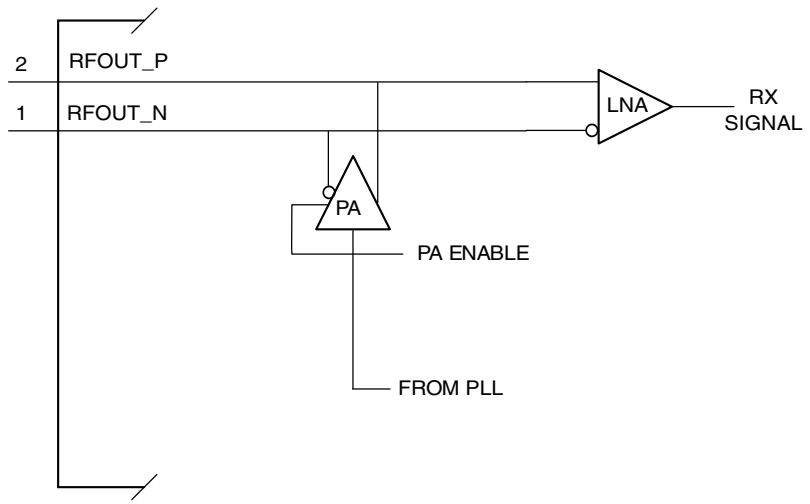


Figure 3-3. RF Interface Pins

3.9.1.1 Antenna Diversity

Fast Antenna diversity (FAD) is supported in hardware through use of the four (4) dedicated control pins by register setup and control selection. They are ANT_A, ANT_B, RX_SWITCH, and TX_SWITCH. These switches, powered by VBAT, are programmable to sink and source two (2) levels of current (2 - 3 mA and 10 mA) or can operate in a high impedance mode. Each I/O will have a buffered feature with adjustable slew rate capability. Unless overridden by software, the last-good-antenna-selection (based on a successful preamble detect), is locked in place for TX sequences, as well as CCA and Energy Detect sequences. The hardware-selected antenna will not change again until the next RX preamble detection.

3.9.1.2 Fast Antenna Diversity (FAD)

The following system factors can be considered when configuring FAD for end applications:

- With Fast Antenna Diversity, sensitivity performance increases the low-end limit of the dynamic range of the system.
- There is little cost of FAD which is essentially enabling the functionality and adding the circuitry for situations where the application runs the possibility of operating at the limits of RX sensitivity.

The next figure shows a simplified block diagram of the two port antenna control topology.

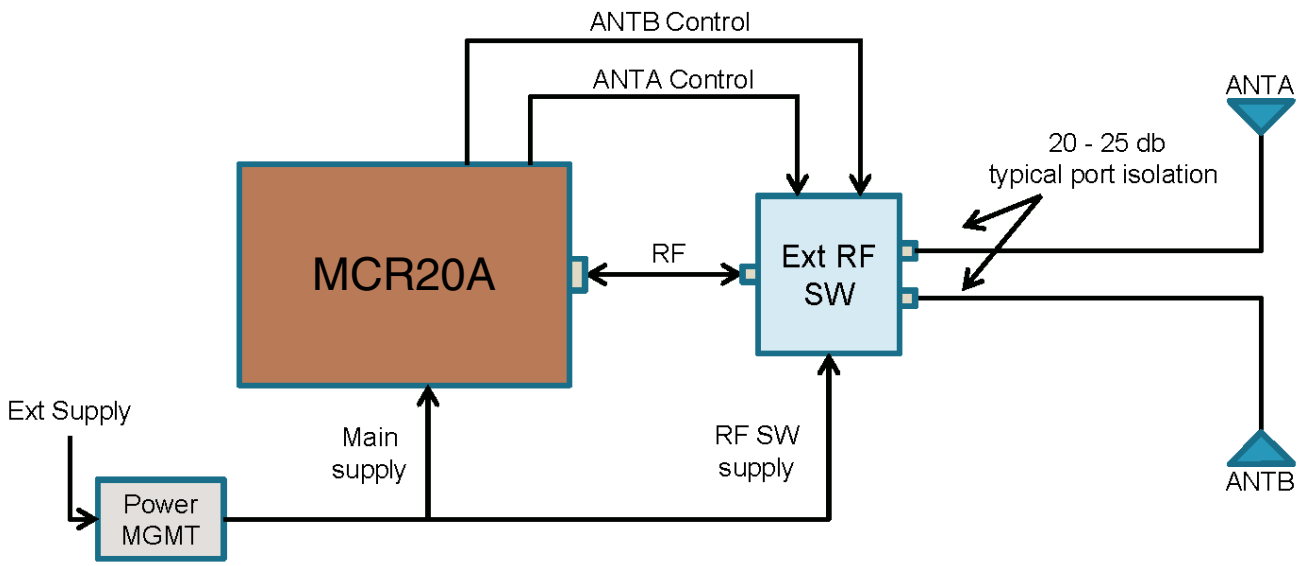


Figure 3-4. FAD control simplified block diagram

FAD Control Register Setup

The following registers allow setup, control and enable of the Fast Antenna Diversity (FAD) feature:

Antenna PAD Control (Indirect Modem_ANT_PAD_CTRL, 0x30): This register sets the action on the pads ANTA, ANTB, RX_SWITCH and TX_SWITCH.

Set bit field ANTX_EN [1:0] to proper selection for end application (default is 0).

- ANTA and ANTB switch signal during the preamble phase of the receive cycle.
- RX_SWITCH and TX_SWITCH switch during the receive and transmit cycle respectively.

Miscellaneous Pad Control (Indirect Modem_MISC_PAD_CTRL, 0x31): This register sets the pad drive strength for ANTA, ANTB, RX_SWITCH and TX_SWITCH.

Bit[0] = 0 (default) for normal drive strength.

Antenna AGC and FAD Control (Indirect Modem_ANT_AGC_CTRL , 0x51): This register controls the FAD feature.

Enable bit field [0] to 1 - Turns FAD on (default is 0).

The following figure shows the flow and timing diagram from the idle state when the receiver takes either the FAD RX flow or the standard RX flow:

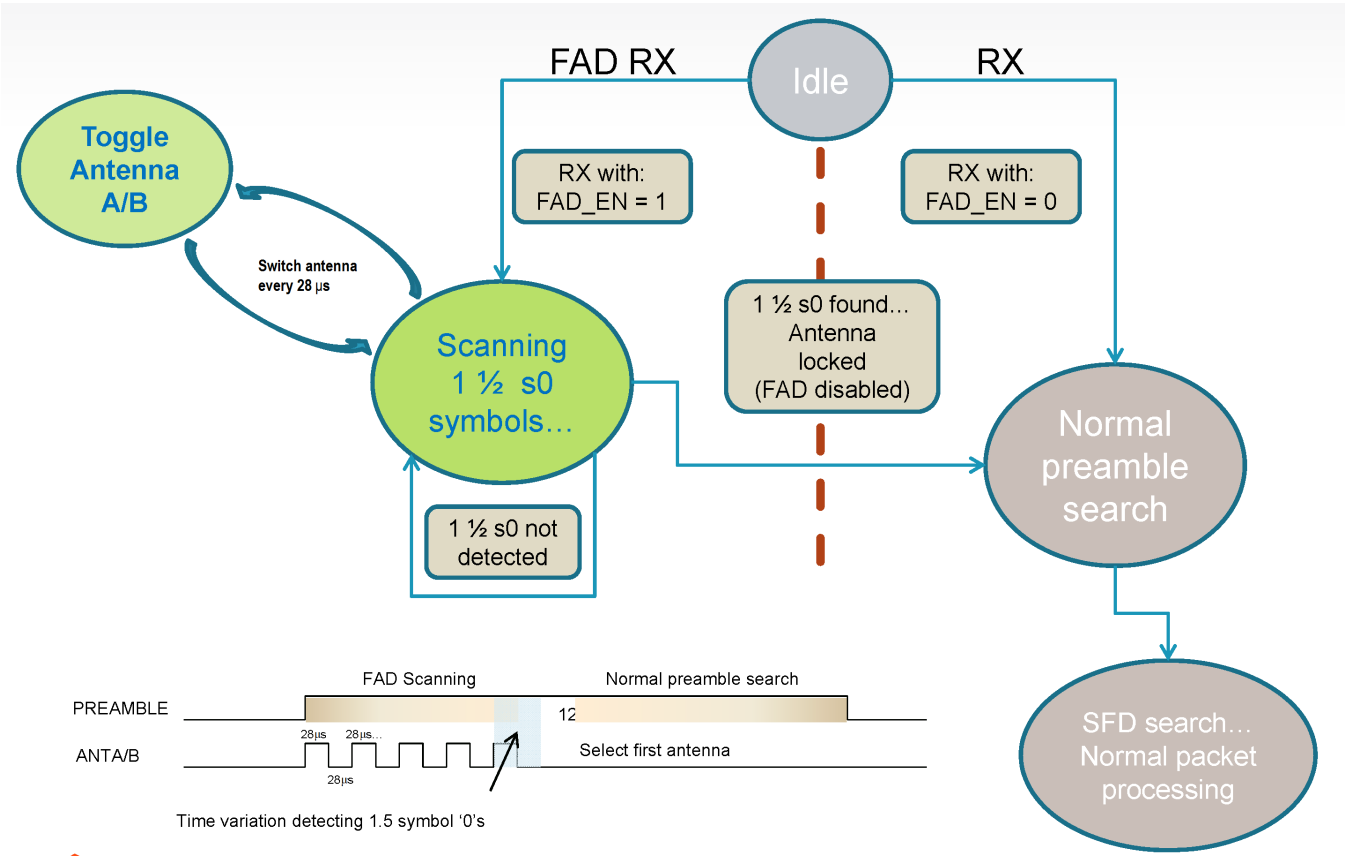


Figure 3-5. FAD flow and timing diagram

FAD and Packet Timing

Packet timing with FAD enabled with two external antenna's A and B.

During the time period t1 – t0, the antenna's are switching at a rate of 28 µs.

- The circular region shows the timing variation while detecting 1.5 symbol '0's and is a function of system sensitivity level.
- At t1, the 'first' antenna meeting the 1.5 symbol '0' criteria is selected … 3 more symbol '0's are then detected to complete the preamble search.

Normal packet processing follows a good preamble match

- The preamble period t2 – t0 is fixed at 128 µs of the total packet cycle time – if a valid preamble is not found during this period, the receiver will start a new RX cycle.

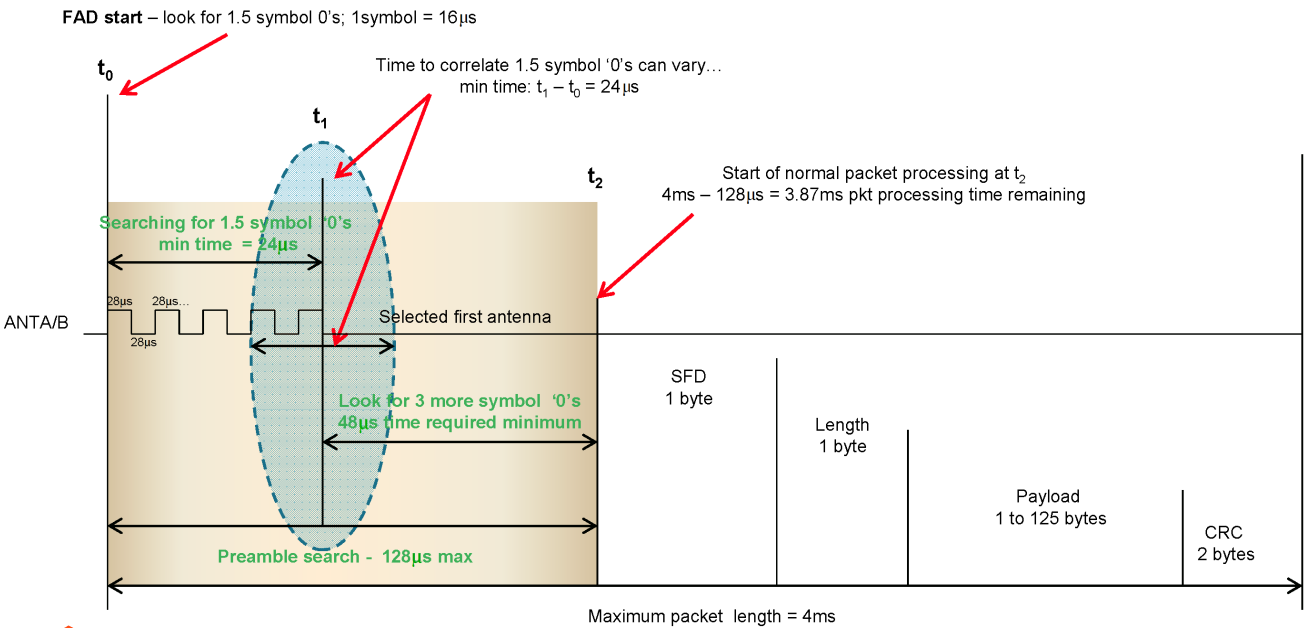


Figure 3-6. FAD preamble and packet timing

The next figure shows a typical FAD control pin design topology

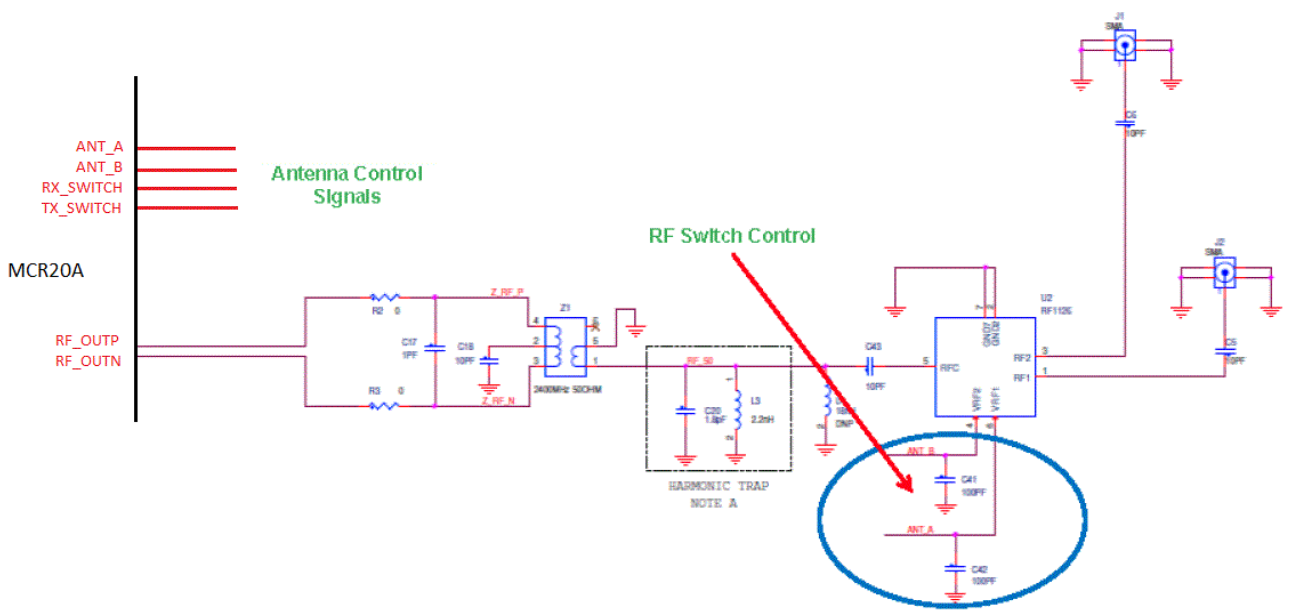


Figure 3-7. Typical circuitry for a two-antenna external switch design

The following figure shows a simulated out of range antenna scenario when using the isolation through an RF switch to offset the RF signal by 21 dBm. The following sensitivity performance summary explains non-FAD and FAD Enabled Sensitivity Improvement.

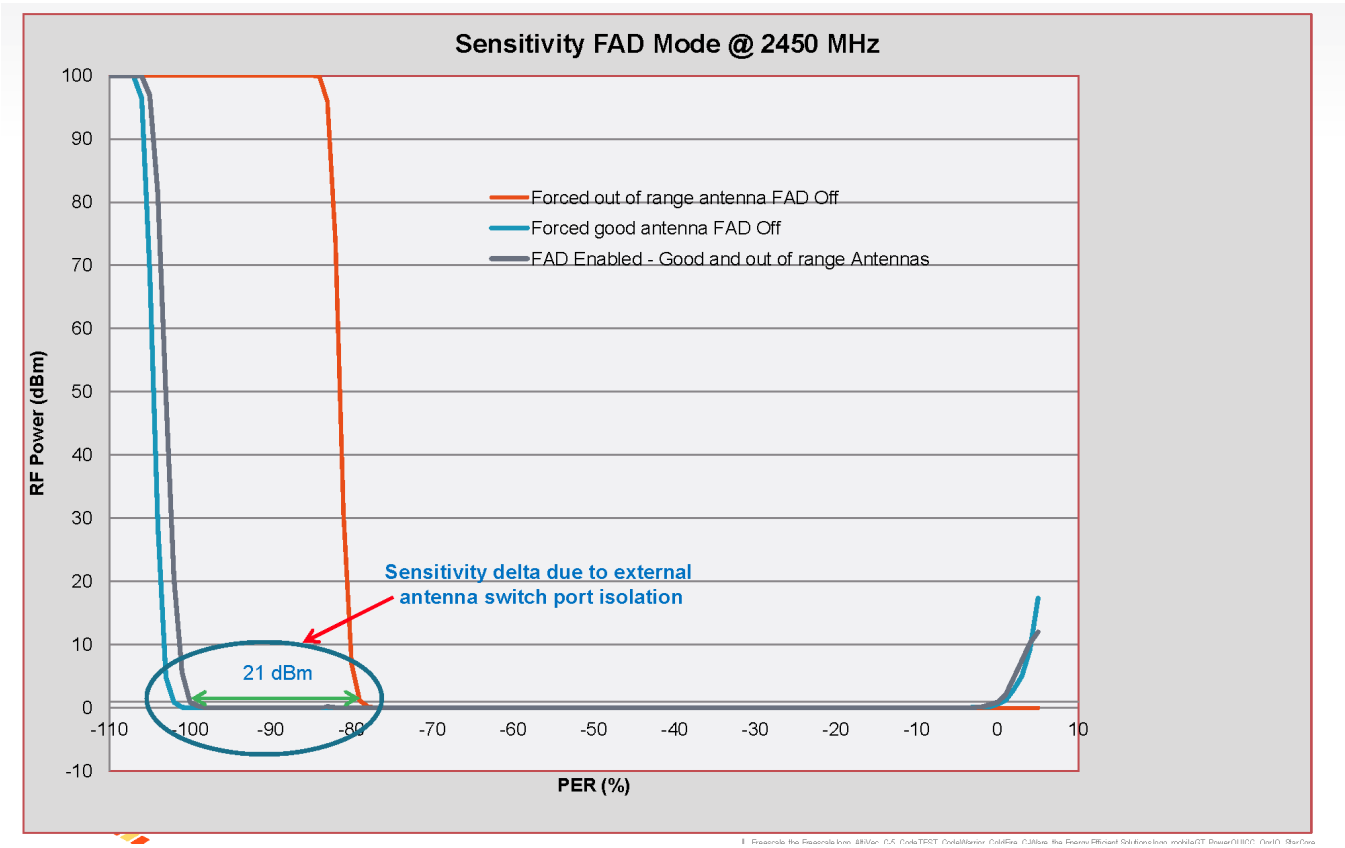


Figure 3-8. Sensitivity FAD mode at 2450 MHz

Sensitivity Performance Summary

Non-FAD mode

- By purposely choosing the bad antenna (out of range RF signal) it can be seen (orange curve) that the sensitivity is shifted to –78 dBm.
- By purposely choosing the good antenna (RF signal in range) it can be seen (blue curve) that the sensitivity is normal.
- The sensitivity delta is due to the isolation characteristics of the external RF switch component. This is the basic limitation of the system. The isolation of the RF switch ports in most cases is between 20-25 dB.

With FAD enabled (purple curve): With one antenna out of range and the other antenna good, the sensitivity is NOW in the normal range (about -99 dBm or better).

- Sensitivity performance is impacted by 1-2 dB in FAD mode vs. an ideally placed antenna but overall PER performance is significantly increased vs. having a single antenna with poor placement.
- Within the dynamic range of the receiver, sensitivity in FAD mode does not impact performance and therefore is not discernible to the user.

- Outside the dynamic range of the receiver is where FAD will help system performance and will have noticeable improvements to the system.
- Unless overridden by software, the last-good-antenna-selection (based on a successful preamble detect), is locked in place for TX sequences as well as CCA and Energy Detect sequences. The hardware-selected antenna will not change again until the next RX sequence.

3.9.2 RF Output Power Distribution

The following figure shows the linear region of the output and the typical power distribution of the radio as a function of PA_PWR [4:0] range. The PA_PWR [4:0] is the lower 5 bits of the PA_PWR 0x23 direct register and has a usable range of 3 to 31 decimal.

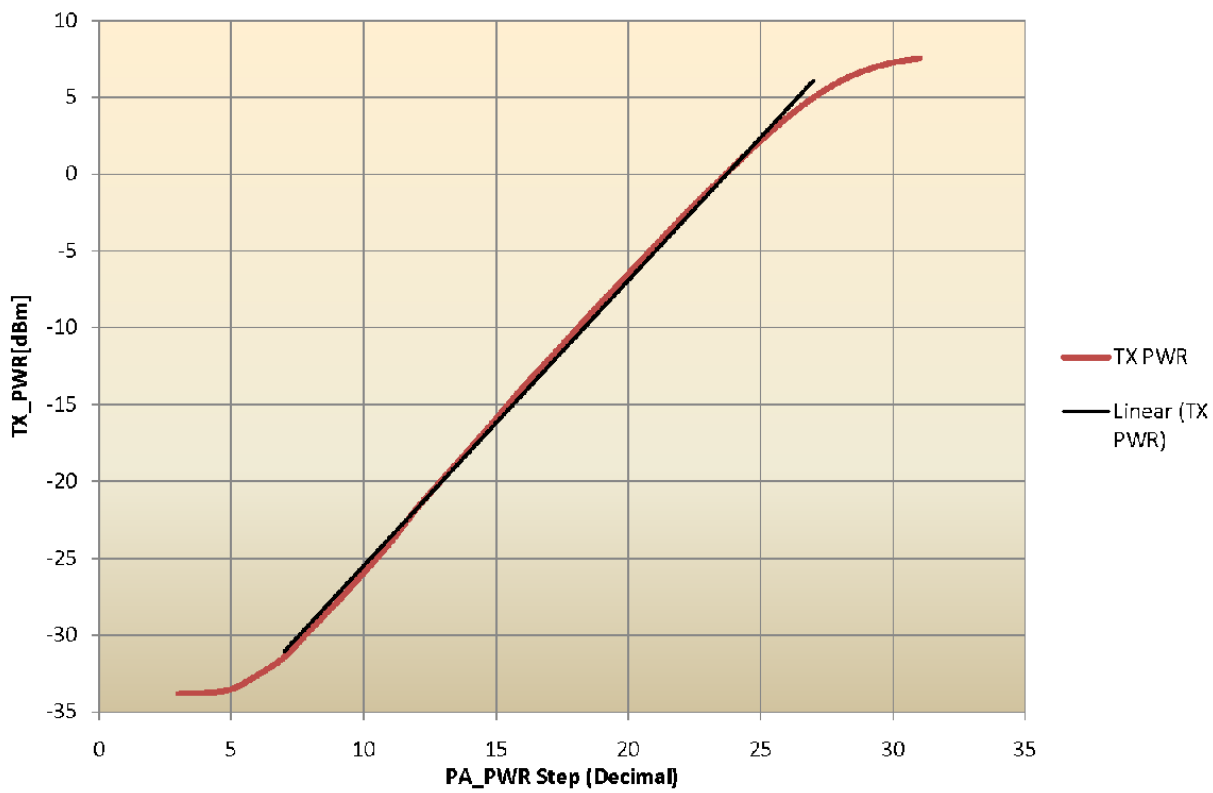


Figure 3-9. MCR20A transmit power vs. PA_PWR step

3.9.3 LQI ED RSSI

The following figure shows the devices reported Energy Detect (ED) as a function of input power. The figure also shows the Link Quality Indication (LQI)/RSSI_CONT (continuous) and the RSSI (Received Signal Strength Indication) as a function of a unitless numeric register reading. LQI is available 64 µs after the preamble is detected. RSSI_CONT can be used to read LQI if a continuous value averaged over the entire received packet is desired. This value may be read as LQI if continuous update of LQI is desired during packet reception (both scaled the same but averaged differently). The curves are measured using the default offset compensation value and can be changed to center the curve if desired.

The following sequeces are performed in test software to create curves in the figure otherwise similar sequences are incorporated into our software stacks:

- For ED, device is configured to perform an ED via write to the register 0x7 with 0x00
    - ED/CCA sequence is started via write to register 0x3 with 0x3
    - Register 0 is polled for bit 3 to be set
    - RSSI_CONT_EN bit is disabled in Indirect 0x25 (CCA_CTRL) all other bits are reset/overwrite values
    - direct register 0x0B is read (CCA Final)
    - direct register 0x25 is read (LQI)
    - indirect register 0x5B is read (RSSI)
    - direct register 0x26 is read (RSSI_CONT) [not enabled in ED case so ignored]
- For LQI, device is configured to perform an LQI via write to the register 0x7 with 0x00
    - Device is configured to perform an LQI via write to the register 0x7 with 0x00
    - RX sequence is started via write to register 0x3 with 0x1
    - Register 0 is polled for bit 2 to be set
    - RSSI_CONT_EN bit is Enabled in Indirect 0x25
    - RSSI_CONT_EN bit is disabled in Indirect 0x25 (CCA_CTRL) all other bits are reset/overwrite values
    - Direct register 0x0B is read (CCA Final)

- Direct register 0x25 is read (LQI)
- Indirect register 0x5B is read (RSSI)
- Direct register 0x26 is read (RSSI_CONT) [Valid for LQI]

For both LQI and ED the input power is swept with a modulated input signal from -100 dBm to -20 dBm in steps of 1 dBm and the above sequences are called for each step and results recorded.

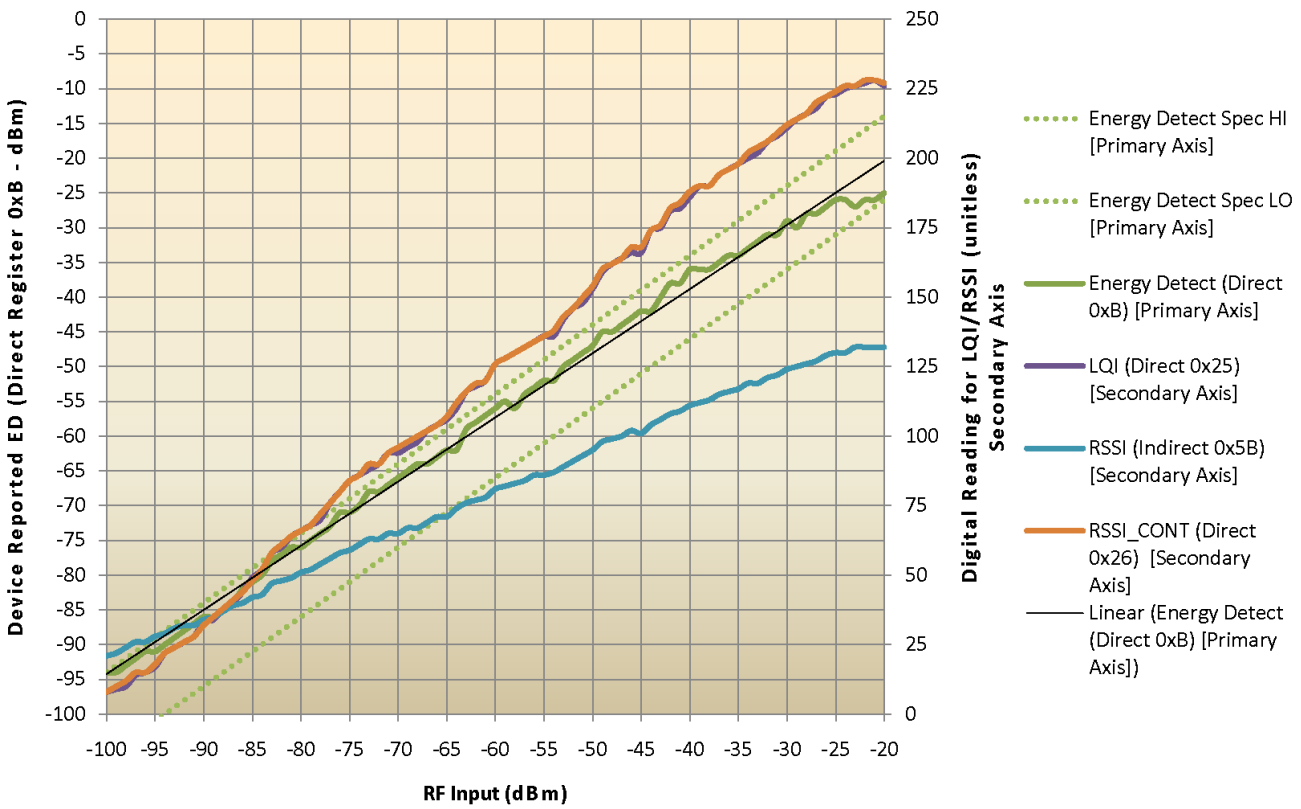


Figure 3-10. ED/LQI relation to RSSI/RSSI_CONT

From the figure above, an equation can be derived as follows. Using Y(LQI) and X(RF), the minimum and maximum values from the figures secondary axis are determined as:

Table 3-3. Minimum and Maximum Values

LQI	Y(LQI)	X(RF)
Maximum	225	-25
Minimum	12	-100
Point near middle Dynamic Range	125	-60

Therefore the slope = 2.84 and B = 295.4. So to convert the 8-bit unitless number read from direct register 0x25 to an ideal RF equivalent, use the following equation:

RF = (LQI – 295.4) / 2.84

3.10 Timer Resources

When writing software applications, the user should be aware of the set of timer resources available.

3.10.1 Modem Event Timer

The MCR20A transceiver contains an internal Event Timer block whose clock is derived from a programmable prescaler which is driven by the 32 MHz crystal source. The Event Timer consists of the prescaler and a 24-bit counter which increment whenever the modem crystal clock is operating. The Event Timer provides the following functions:

- The 24-bit counter generates “current” system time - The counter runs continuously when the modem is not in a low power mode and provides a local “current” time. The present value can be read from the EVENT_TMR_USB/MSB/LSB registers. A counter frequency of 250 kHz is used. The counter value is used to generate a time stamp for received frames and is also used for comparison to four different comparators that allow up to four different time delays
- Interrupt generation at pre-determined system times - There are four comparator functions that can generate an interrupt to the MCU when the value of the comparator matches the system counter. These function as timers to generate time delays or trigger events and can augment the MCU timers for generating time delays required within the 802.15.4 application software
- Exit from Doze Mode at pre-determined system time - Timer comparator Tmr_Cmp2 can be used to wake-up the modem from low power Doze Mode
- Latches “timestamp” value during packet reception - With every received frame, there is a timestamp generated and stored in a modem SPI register. The timestamp is taken from the counter when the frame length indicator (FLI) is received
- Initiates timer-triggered sequences - The modem has three basic active modes of RX, TX, and CCA. Each of these sequences can be timer-triggered to become active after a set delay based on timer comparator Tmr_Cmp2. This feature can be useful for implementing 802.15.4 application sequences

3.11 Low Power Considerations

Many IEEE 802.15.4 applications such as sensor End Devices are required to be battery operated. As expected, long battery life is highly desirable and is very dependent on application parameters. Over-the-air operation uses RX, TX, and CCA modes, where power is highest. As a result, the time between radio operations should be kept at the longest possible period that the application will allow.

When designing for low power operation consider:

- The modem and the MCU each have several low power options
- The modem is entirely controlled by the MCU; the low power options/combinations will be determined by MCU programming
- The power down control of the modem must be maintained by the MCU in the MCU's power down configuration
- All unused port GPIO must be configured to a known state (preferably output low) to prevent unwanted leakage current.

Lowest power in a system is more than just putting the modem and/or the MCU in a low power mode. The relationship between the functions, the timing between them, and clock management must all be considered. The duty cycle between active operations is also very important as it can impact whether sleep operation or active operation will have the biggest impact over an extended time period.

3.11.1 Low-Power Preamble Search (LPPS)

The Low Power Preamble Search (LPPS) feature of the radio allows the user to effectively save receiver current while maintaining adequate system performance. Attributes of the the LPPS feature are described as follows:

- When enabling the LPPS feature the demodulator and other sub-sections of the receiver can be independently duty cycled at 50%.
- The receiver current consumption is reduced approximately 4 mA.
- Sensitivity performance decreases slightly due to time delta in the correlator while searching for an additional s0 before entering normal preamble search.
- There is no added cost of using the LPPS feature since no extra components are required. In fact, enabling LPPS mode all the time has little system design impact and the current savings is 20%.

LPPS Control Register Setup

There is one register that allows setup, control and enabling of the Low-Power Preamble Search (LPPS) feature:

- LPPS Control (Indirect Modem_LPPS_CTRL, 0x56) This is the control register and has this bit field definition: LPPS_EN [bit 0] (Master Enable for Low-Power Preamble Search (LPPS) mode) the Buffer, LIM, RSSI and LNA can be controlled independently. See section LPPS_CTRL (Indirect_Modem_LPPS_CTRL)
- 1: Enable Low-Power Preamble Search mode
- 0: Normal operation

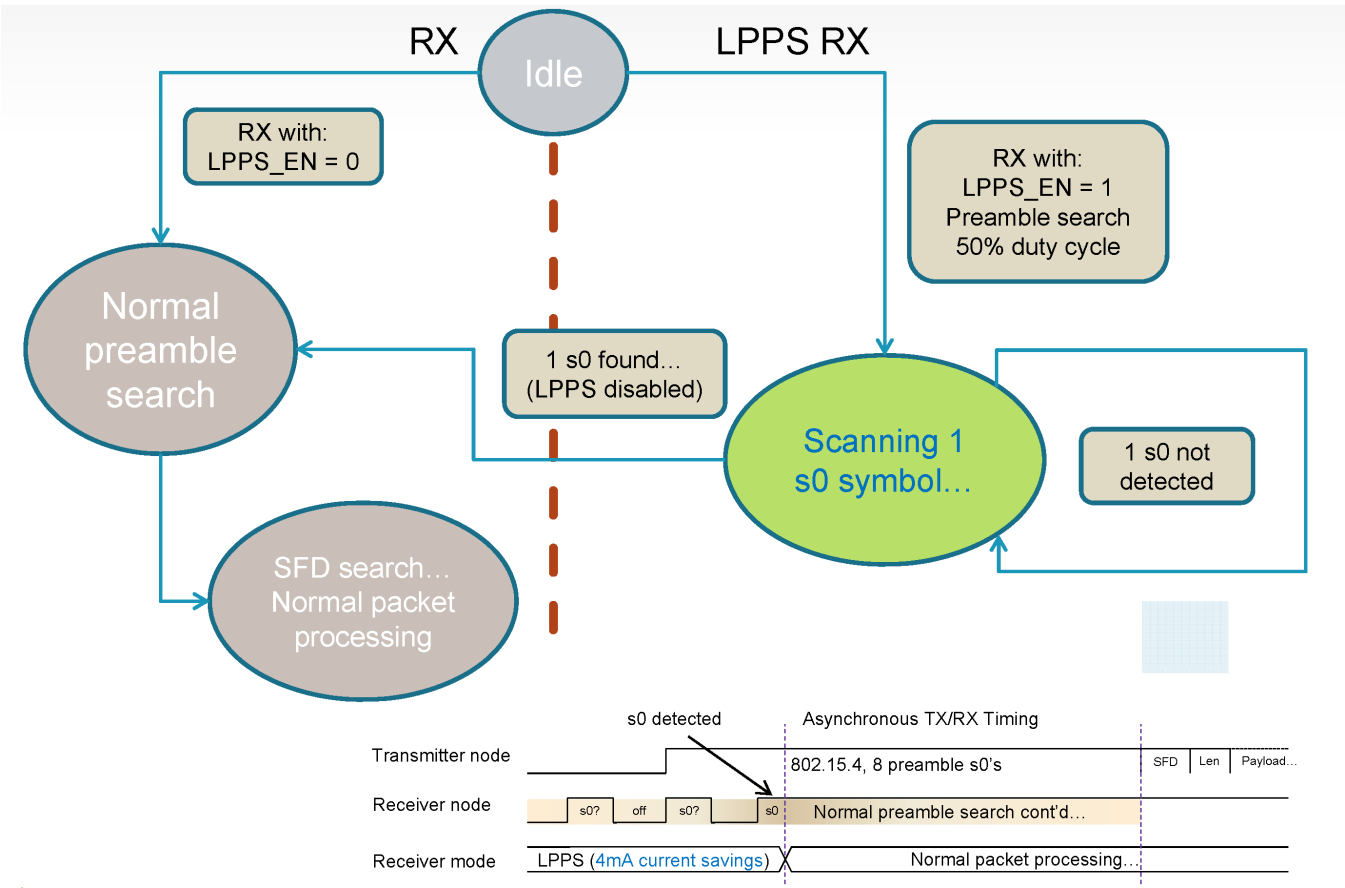


Figure 3-11. LPPS flow diagram

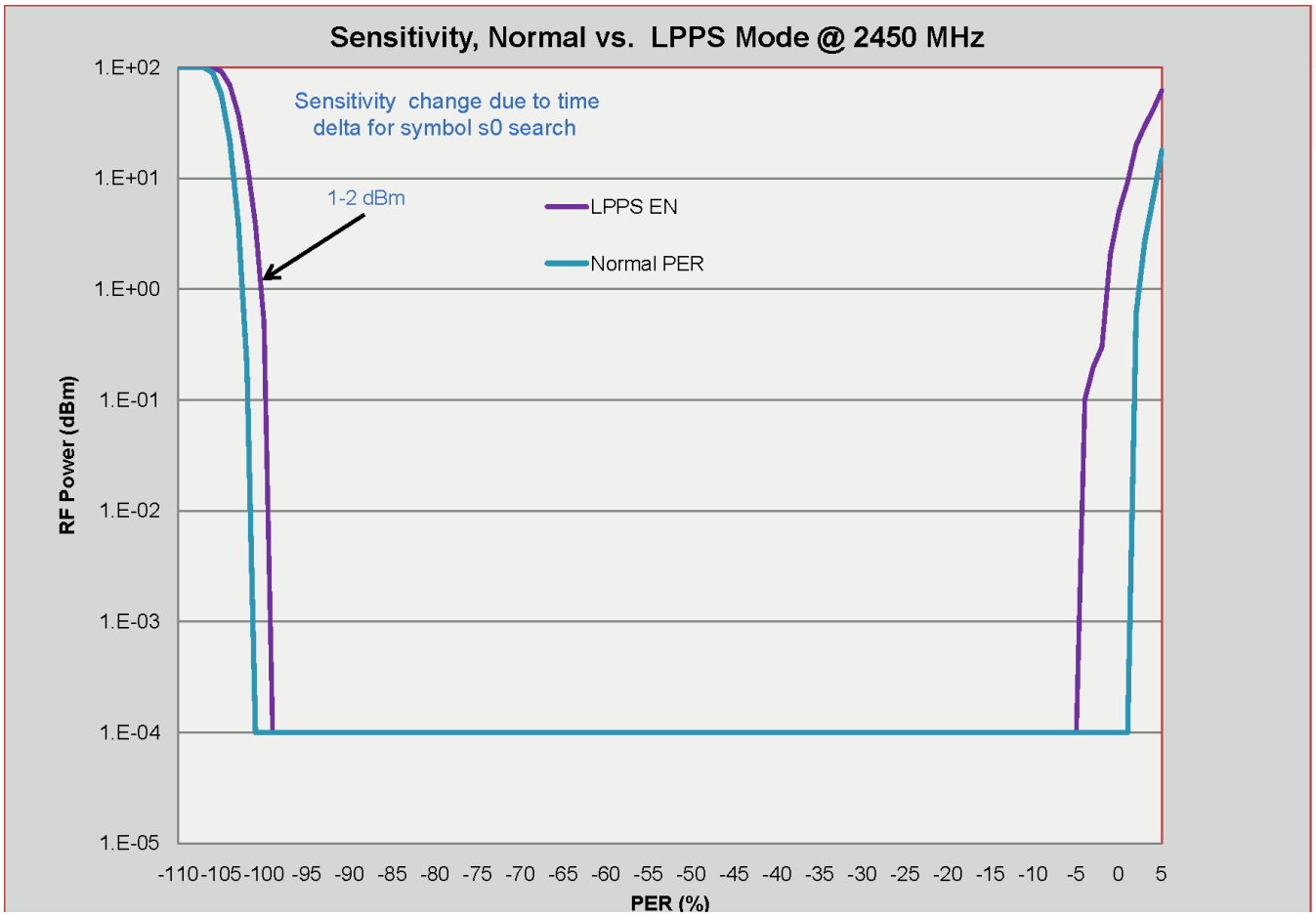


Figure 3-12. LPPS enabled sensitivity performance

Sensitivity Performance Summary

Normal (non-LPPS) PER mode

- Sensitivity in normal PER mode is -102 dBm as shown by the blue curve

LPPS enabled (purple curve)

- Sensitivity is slightly reduced by 1 - 2 dBm due to time delta the receiver requires to search for the s0 symbol defined by the mode
- The current is reduce by 4mA in LPPS mode with no additional system cost
- There is no discernible disadvantage of enabling LPPS all the time and taking advantage of the current savings

3.11.2 Recovery Times from Low Power Modes

This section discusses use cases and summarizes the modes of operation of the transceiver.

3.11.3 Run Time Current

As previously stated, the modem is fully under control of the MCU. For lowest power, the modem should be kept in a low power mode as much as possible. However, it is also important to understand the current profile of the modem under active operation. In a similar sense, the MCU can use varying levels of current depending primarily on the clock sources and frequencies used.

3.11.3.1 Modem Active Currents

In normal operational mode the modem's rest state in the Idle Mode. All active sequences originate from the Idle or Doze Mode and return to the Idle or Doze Mode. The three active sequences are Clear Channel Assessment (CCA), RX, and TX, and each has a separate current profile. The following table lists the typical currents while in the listed modes, but does not show the transition profiles when moving between modes.

Table 3-4. Modem Active State Currents

Mode	Current (typ @ 2.7V)
Idle (No CLK_OUT)	700 µA
Hibernate / Doze (No CLK_OUT)	<1 µA / 500 µA
CCA/ED	19 mA
RX / RX LPPS Mode	19 mA, 15 mA LPPS
TX (0 dBm nominal output power)	17 mA

A normal sequence of events may include a 802.15.4 node performing first a CCA to see if the channel is clear, second transmitting a frame (assuming the channel is clear), and finally after the TX, going into to Receive Mode to look for an acknowledge. The modem has built-in complex autosequences to manage different operation modes.

Chapter 4
Modem: Modes of Operation

4.1 Power Management Overview

The power management structure provides the biasing and the framework to power manage the MCR20A.

The power management includes:

- A 1.8V digital regulator with POR
- A 1.8V analog regulator
- Biasing block

The supply strategy block diagram is provided in the following figure.

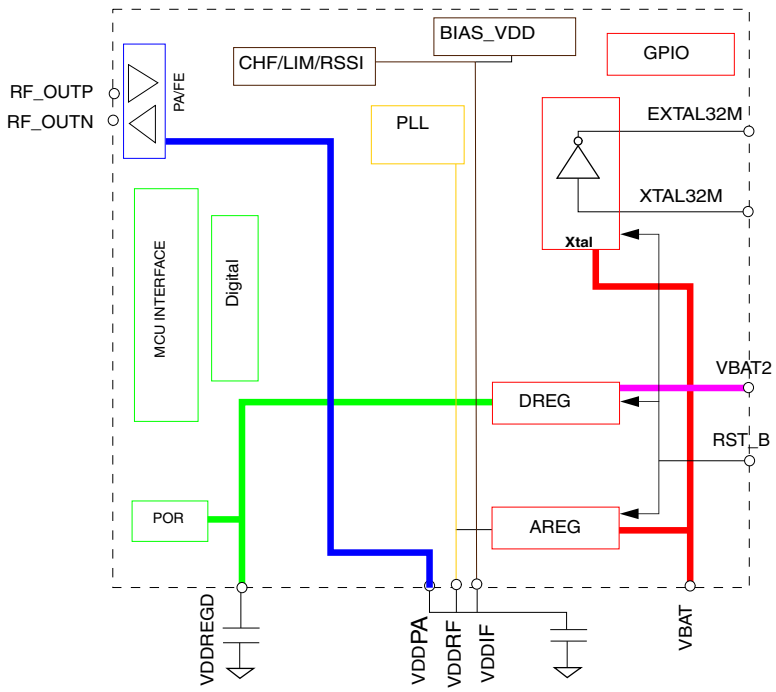


Figure 4-1. Supply strategy block diagram

4.1.1 Features

The modems power management structure has the following features:

- 1.8V voltage regulation for internal analog circuitry.
- Power on Reset (POR) for global digital reset.
- 1.8V voltage regulation for internal digital circuitry
- Centralized biasing

4.1.2 Power and Regulation Topology

The following blocks are sourced by the battery supply:

- XTAL
- Digital regulator 1.8 V
- Analog regulator 1.8 V
- The RF blocks are supplied by dedicated supplies:
- VDDPA for the PA that source high current at the Frontend (PA is not used at the same time)

- VDDRF for the PLL block (mainly VCO), clean supply regulated by the LDO directly
- VDDIF for the other analog blocks and also used potentially by the XOR.

VDDPA and VDDIF are connected externally such that decoupling between supplies is easy

4.1.3 Digital Regulator and POR

The digital regulator provides the 1.8 V supply to the digital logic. It is always turned on when the RST_B pin has been released. For VDDREGD, the external decoupling capacitor is in the range of 220 nF and 470 nF. As the figure below shows, the VDDREGD supply is decoupled externally. The digital regulator includes three modes of operation:

- Off mode - controlled by the active low RST_B pin, and active when reset is active
- On mode with maximum current capability - required in RUN mode and during startup
- Low power mode (in LP/HB and DOZE) - The current capability in this mode is limited but retains register contents and SPI functionality. The current consumption is max 300 μA.

POR block provides reset if VBAT is low (</=1.4 V). POR comes out of reset if VBAT is high (>/=1.65 V). The POR is used to hold the digital logic in reset and then release the xtal oscillator when the power is going up.

Note

- The POR is disabled during loose mode. (See "Modes and start up process" Table).
- There is no battery monitoring

4.1.4 Analog Regulator (AREG)

The analog regulator is a low drop linear regulator that regulates to 1.8 V typical from the battery voltage 3.6 V-1.8 V. When the battery is low the regulated supply VDDRF follows the battery voltage (max 150 mV drop). This LDO provides the supply to the RF

(including PA) and analog circuitry of the modem. The decoupling capacitor is in the range of 220 nF and 470 nF. As this figure shows, the three VDDA supplies (VDDRF, VDDIF and VDDPA) are decoupled externally.

4.2 Modem Operational Modes Summary

The modem has a number of passive operational modes that allow for low-current operation as well as modes where the transceiver is active. These modes are summarized in the following table.

Table 4-1. Modem Mode Definitions

Mode	Definition
Off	$\overline{\text{RST}}$ asserted. All IC functions Off, Leakage only.
Hibernate	Crystal Reference Oscillator Off.
Doze	Crystal Reference Oscillator On but CLK_OUT is available only if enabled
Idle	Crystal Reference Oscillator On with CLK_OUT output available.
Receive	Crystal Reference Oscillator On. Receiver On.
Transmit	Crystal Reference Oscillator On. Transmitter On.
CCA / Energy Detect	Crystal Reference Oscillator On. Receiver On.

4.2.1 Power modes

The RESET/power down mode is the mode used to turn everything off in the circuit. All the blocks supplied by VBAT are controlled directly by the RST_B pin for the power consumption of the circuit to be minimal.

Note

The GPIO pins must not remain floating.

When RST_B is released, the digital regulator will automatically startup. When the digital supply is sufficiently high the POR signal is released. During startup, the RSTB high must be ensured for proper startup (pullup)

When POR is released, it starts automatically where the XTAL and a wake up interrupt is generated. The circuit is then set in IDLE mode.

All the other modes can be accessed through SPI generation by programming the blocks independently. For example, the RUN mode (TX or RX) is activated through the xcvseq setting. Care must be taken that before going to this mode the XTAL is enabled and the digital regulator is in full mode. The analog regulator will be automatically turned "on" then by the state machine.

The LP/hibernate mode is a specific mode where the XTAL is disabled. Going out of this mode via the SPI means that the XTAL needs to be started up again, the digital regulator forced to high power mode and a new wake up interrupt is generated.

The following figure provides the power state flow diagram for the modem and the "Modes and start up process" Table, summarizes power mode and start up processes.

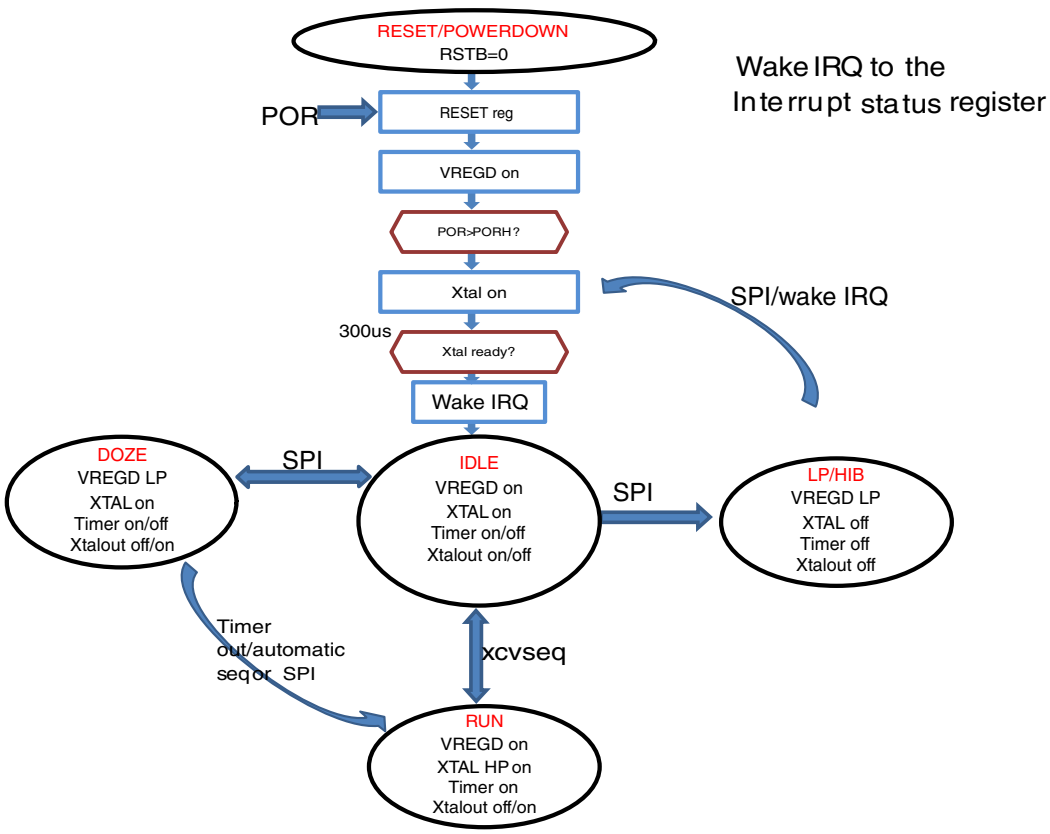


Figure 4-2. Power State Diagram

Table 4-2. Modes and start up process

Mode[1]	VREGD/POR	XTAL	VREGA	XTAL_out	Timer	Registers	GPIO	Current consumption
IDLE	ON	ON	OFF	Programmable	ON	Normal	Known	700 µA, typical (no clockout)
DOZE	Loose mode	ON	OFF	Programmable	ON/OFF	Retained	Known	500 µA, typical

Table continues on the next page...

Table 4-2. Modes and start up process (continued)

Mode[1]	VREGD/POR	XTAL	VREGA	XTAL_out	Timer	Registers	GPIO	Current consumption
								(no clockout)
Low power / Hibernate	Loose mode	OFF	OFF	OFF	OFF	Retained	Known	<1 µA[2]
Reset / Powerdown	OFF	OFF	OFF	OFF	OFF	Reseted	Known	<30 nA
RUN[3]	ON	ON	ON	Programmable	ON	Normal	Known	15 mA

1. Values identified in this table reflect nominal voltage and temperature.
2. Value does not include SPI activity in LPPS mode.
3. Current of 15 mA refers to radio current.

Note

Providing a clock output signal on CLK_OUT will add some power consumption depending on the capacitive load and frequency.

4.3 Sequence Manager

The Sequence Manager (also known as the RF digital manager), is a hardware state machine that controls the timing of all transmit, receive, and CCA operations in the modem. The Sequence Manager interfaces directly with 3 lower level managers (RX digital manager, TX digital manager, and PLL digital manager). For each digital and analog element in the modem's transceiver, these 4 state machines operate collaboratively, to perform some or all of the following functions:

- Activation
- Deactivation
- Initialization
- Mode transitions
- Calibration routines
- Clock enabling and disabling
- Interrupt triggering and status generation

Designed into the sequence manager are a variety of complex sequences (called autosequences), which are sequential concatenations of simpler, building-block sequences. The hardware autosequences provide the following advantages:

- Smaller S/W memory footprint: H/W automates operations previously done by S/W

- Increases MCU sleep times, reducing current consumption
- Handles timing-critical sequences (rx-to-tx), e.g., data-polling, not possible in S/W

Sequences can be initiated by software directly, or alternatively, can be initiated automatically at the expiration of a hardware timer. The general parameters of the sequences can be programmed by software prior to initiating the sequences. Such parameters allow software to select:

- Whether CCA is required ahead of a transmit sequence
- Whether an automatic Ack frame should be transmitted after a receive sequence
- Slotted vs. unslotted mode
- Type of CCA (e.g., Energy Detect, CCA mode 2)
- Whether the device is a PAN Coordinator or not
- Whether the device resides on 2 networks simultaneously (Dual PAN mode)

The modem's Sequence Manager also includes support for Dual PAN mode, and Active Promiscuous mode.

4.3.1 Modem Sequence Manager Operation

The Sequence Manager controls, coordinates, and automates all transceiver sequences within the modem. Software selects the desired sequence, sets the general parameters, and (optionally) configures a hardware timer to trigger the sequence at a precise time in the future. Subsequently, the MCU can enter a low-power state, or tend to other activities, while the sequence manager state machine conducts the programmed transceiver operations. When the sequence completes, an interrupt alerts the MCU. At this point the MCU can check the completion status of the sequence, download received data from the packet buffer, and then prepare the next sequence.

Sequences can be simple transceiver operations. For example, transmit one frame, or, perform a CCA on a channel. Alternatively, the sequence manager supports complex sequences, which use simple operations as building blocks, strung together in a back-to-back fashion, with precise timing intervals between operations. For example, a “TR” sequence calls for a transmit frame followed by a receive frame. For complex sequences, interrupts can optionally be generated at the completion of each stage of the sequence, providing greater visibility for the MCU into the timeline of the sequence. Software can determine the precise state of the sequence manager state machine at any time. Software can opt to abort a sequence at any time; when this happens, the sequence manager will return to its IDLE state in an orderly fashion.

The sequence manager is responsible for the implementation of the Wireless Physical Layer (PHY) specifications described in the IEEE 802.15.4 standard, in the modem. This is accomplished by activating and deactivating the key digital and analog elements in the

transmit or receive chain, at precisely the right times, and for precisely the right durations. Most of the low-level timing parameters are hardwired within the state machine, and thus are not programmable. The time resolution of the sequence manager is 2 µs (1/8 of a transmitted symbol), so this is the granularity of the individual steps. The higher-level timing parameters, such as the interval between operations in a complex sequence, are largely programmable, except for those defined as constants in the IEEE 802.15.4 standard.

4.3.2 Functional Description

This section describes the supported sequences in more detail. All sequences can be initiated by software directly (instantaneously) by writing the desired sequence to the XCVSEQ register, with the TMRTRIGEN bit deasserted. Alternatively, software can schedule the initiation of a sequence at a precise time in the future, by programming the desired start time into the TC2 timer compare register, and setting the TMRTRIGEN bit. The XCVSEQ register field is writable and readable, so software can read this register to determine which sequence it had programmed earlier. After the sequence completes (SEQIRQ interrupt), software must set the XCVSEQ to IDLE (0x0), before initiating a new sequence. This prevents the undesired, inadvertent re-triggering of the sequence manager on the next TC2 match. If the sequence-complete interrupt (SEQIRQ) is masked, software can determine when the sequence is complete by polling the SEQ_STATE register, until it returns to idle (0x0). While a sequence is ongoing, software can abort the sequence at any time by writing IDLE (0x0) to XCVSEQ; the sequence manager state machine will respond with an orderly return to the SEQ_IDLE state, and a SEQIRQ will be generated. While a sequence is underway, software should not attempt to change XCVSEQ (other than aborting the sequence as previously described); the sequence manager will ignore all mid-sequence attempts to change XCVSEQ. As mentioned previously, software must write the IDLE value to XCVSEQ before changing sequences.

The supported sequences are shown in the following table.

Table 4-3. Transceiver supported sequences

XCVSEQ	Sequence	Description
0	I	Idle
1	R	Receive Sequence - conditionally followed by a TxAck
2	T	Transmit Sequence
3	C	Standalone CCA

Table continues on the next page...

Table 4-3. Transceiver supported sequences (continued)

XCVSEQ	Sequence	Description
4	TR	Transmit /Receive Sequence - transmit unconditionally followed by either an R or RxAck
5	CCCA	Continuous CCA - sequence completes when channel is idle
6	Reserved	
7	Reserved	

All sequences can be aborted by a PLL unlock detection. In the modem, this is referred to as PLL unlock. The unlock detection is qualified by the sequence manager; in other words, an unlock detection during the early stages of the PLL warmup is expected, and is not allowed to abort the sequence at that point. The primary rationale for the PLL unlock abort, is to prevent transmission on an incorrect, or unstable, frequency during a transmit operation, or to increase battery life by not prolonging a receive or CCA operation that is bound to fail. Software can disable the PLL unlock auto-abort altogether by asserting the PLL_ABORT_OVERRIDE bit.

4.3.2.1 Supported Sequences

4.3.2.1.1 Sequence (Idle)

When the IDLE value is written to XCVSEQ, the sequence manager goes to its SEQ_IDLE state. If not already in SEQ_IDLE, the sequence manager executes an orderly warmdown to return to SEQ_IDLE. A SEQIRQ interrupt is then issued. Writing IDLE value to XCVSEQ is the proper way to abort a sequence.

4.3.2.1.2 Sequence R (Receive)

Sequence R is the basic receive sequence. Sequence R can be used to receive all types of 802.15.4 PHY- and MAC-compliant frames, including reserved frame types. However, reception of Acknowledge frames is not recommended using Sequence R. This is because reception of an Acknowledge frame, usually follows a transmitted frame, with a designated Sequence Number, so that the received Acknowledge frame can be verified for the matching Sequence Number. In a standalone Sequence R, there is no Sequence Number to verify against, so any Acknowledge frame is merely transferred to the Packet Buffer. (Note: the appropriate way to receive Acknowledge frames is with Sequence TR, with the RXACKRQD bit asserted). Using Sequence R, all frames that pass frame-filtering rules and CRC check, are transferred to the Packet Buffer.

The basic execution of a successful Sequence R is as follows:

- MCU writes TC2 compare value to the desired initiation time, and sets TMRTRIGEN (optional)
- MCU writes XCVSEQ=R
- Wait for TC2 match (if TMRTRIGEN=1)
- SEQ_MGR executes RxWarmup
- Preamble Detected
- SFD Detected
- FrameLength octet received, transferred to Packet Buffer
- N additional octets received (N = FrameLength), transferred to Packet Buffer. Note if (FrameLength > 127), then N = 127.
- If frame-filtering rules and CRC check pass
    - RXIRQ interrupt issued
    - SEQ_MGR executes RxWarmdown
    - If received Frame Control Field indicates AckRequest=1
        - SEQ_MGR executes TxWarmup
        - an Ack frame is transmitted using the received Sequence Number
        - SEQ_MGR executes TxWarmdown
- SEQIRQ interrupt issued

When Sequence R is initiated, the sequence manager warms up the receiver (analog and digital elements). This process takes 144 µs. From this point forward, timer TMR3 can be enabled as a “bracketing” timer (software option). If a TMR3 timer match occurs, and software has asserted the TC3TMOUT bit, the sequence manager will warm down the receiver and return to SEQ_IDLE state. Assuming no TMR3 timeout, reception proceeds in preamble-search mode. After a preamble is detected, reception continues in SFD-search mode. At this point, the sequence manager’s slot timer is activated. This is because a slotted “transmit acknowledge” frame (TxAck) may be required later in this sequence. The slot timer is loaded with the following value (in µs):

Slot Timer Preload = 160 + TXWARMUPTIME + (2*ACKDELAY).

The 160 term arises because, at the point of SFD detect, we are exactly 10 symbols into the backoff slot, and the symbol period is 16 µs. (The slot timer timebase is actually 2 µs, half the time-resolution of the equation shown above). The slot timer will count up and eventually rollover at 319, to 0. (There are 320 µs in a backoff slot). The receiver receives the next octet (FrameLength). Starting with this octet, and continuing through the remainder of the frame, each octet that is received is transferred to Packet Buffer. After each octet is transferred to Packet Buffer, the Packet Buffer address is incremented by 1. The next 2 octets (Frame Control Field) are then received, which are parsed to obtain the AckRequest field, and the FrameVersion field. The next octet is then received (SequenceNumber), and stored. The remainder of the frame, determined by FrameLength, is then received. As the octets are received, the packet data is parsed and subjected to packet-filtering rules. If any of the packet-filtering rules fail, or if the CRC

check on the FCS field fails, the sequence manager will return the receiver to preamble-search mode, after the last octet has been received and transferred to Packet Buffer. Assuming the packet-filtering rules and CRC pass, an interrupt (RXIRQ) is issued to the MCU. At this point, the sequence manager must determine whether a TxAck is required. A TxAck is required if the following 3 conditions are all met:

- AUTOACK=1 (register bit)
- PROMISCUOUS=0 (register bit)
- The received FrameControlField has AckRequest=1

If these conditions are met, and SLOTTED mode is not in effect, the sequence manager loads its TxAck timer with the following value (in μs):

TxAck Timer Preload = 192 - TXWARMUPTIME - 2*ACKDELAY + 16*ACKPOSTPONE. Note: ACKPOSTPONE is not supported in the modem and is hardwired to 0.

The 192 μs is the non-slotted RX-to-TX turnaround requirement in the 802.15.4 standard. The TXWARMUPTIME takes into account the fact that the TxAck timer must expire early for the Tx warmup to be complete at exactly the 192 μs point. The ACKDELAY is a signed, fine-tune adjustment to the TxAck transmission time (± 62 μs), and the ACKPOSTPONE allows the TxAck to be delayed an additional 255 symbols, if necessary.

If SLOTTED mode is in effect, the slot timer was previously loaded (at SFD detect), but the sequence manager needs to determine how much time is left in the current backoff slot. If less than 192 μs remain, the sequence manager must wait an additional backoff slot before initiating the Ack transmission. So it sets a flag indicating it must allow the slot counter to rollover an additional iteration.

Finally, at TxAck timer expiration (in non-SLOTTED mode), or at the qualified rollover of the slot timer (in SLOTTED mode), a Tx warmup is initiated. The conclusion of the Tx warmup will coincide precisely with a backoff slot boundary if SLOTTED mode is in effect. The sequence manager will then initiate the transmission of the Ack frame. The entire Ack frame will be built up by hardware; the Packet Buffer is not used for the auto-TxAck feature. The hardware will insert the following parameters into the Ack frame's Frame Control Field:

Table 4-4. Frame Control Field for Hardware-generated Auto-TxAck Frame

FCF Field	Value
FrameType	Acknowledge
SecurityEnabled	0

Table continues on the next page...

Table 4-4. Frame Control Field for Hardware-generated Auto-TxAck Frame (continued)

FCF Field	Value
FramePending	If SRCADDR_EN=1, FramePending gets the value of FramePendingFlag from Source Address Matching accelerator hardware. If SRCADDR_EN=0, FramePending gets the value of ACK_FRM_PND.
AckRequest	0
PanIDCompression	0
Reserved (bits 7-9)	000
DstAddrMode	0
FrameVersion	copy FrameVersion from the previously-received frame
SrcAddrMode	0

The Sequence Number for the Ack packet, will be the Sequence Number obtained from the previously received frame.

After the Ack frame has been transmitted, the sequence manager will issue a TXIRQ interrupt, and then perform a Tx warmdown. Finally, a SEQIRQ interrupt will be issued, and the sequence manager returns to SEQ_IDLE state.

4.3.2.1.3 Sequence T (Transmit)

Sequence T is the basic, standalone transmit sequence. Sequence T can be used to transmit all types of 802.15.4 PHY- and MAC-compliant frames. However, transmission of Acknowledge frames is not recommended using Sequence T. This is because transmission of an Acknowledge frame, usually follows a received frame, with a designated Sequence Number. The sequence manager automates the transmission of an Ack frame that follows a received frame, using Sequence R with the AUTOACK bit asserted. This is the recommended method for transmitting an Ack frame. This is especially beneficial, because the transmitted Ack frame octets are generated by hardware, and so no setup of a transmit packet in Packet Buffer, is required. However, sequence manager hardware does not prohibit Ack frames using Sequence T.

Sequence T allows for the insertion of 1 or 2 CCA (clear channel assessment) measurements prior to transmission, to ensure that the selected channel is idle. The CCA-before-TX feature is governed by two register bits, CCABFRTX and SLOTTED. All CCA measurements must indicate channel-idle, in order for the sequence manager to proceed to transmission; if the channel is determined by CCA to be busy, the sequence manager will terminate the sequence without a transmission. The number of CCA measurements attempted by the sequence manager, and the timing of the measurements, is shown in the following table.

CCABFRTX	SLOTTED	Number of CCA measurements	Timing of Initiation of CCA measurement #1	Timing of Initiation of CCA measurement #2	Timing of First Bit of Transmitted Frame (assumes channel idle)
0	X	0	—	—	Immediately follows TxWarmup
1	0	1	Immediately follows RxWarmup	—	Immediately follows RxPartialWarmdown and TxPartialWarmup
1	1	2	Immediately follows RxWarmup	320 µs after Initiation of CCA measurement #2	320 µs after Initiation of CCA measurement #2

Any CCA mode can be used for Sequence T and Sequence TR (CCA modes 1, 2, and 3 are available). Software must configure the CCATYPE bits prior to initiating the autosequence, to select the CCA mode.

The basic execution of a successful Sequence T is as follows:

- MCU writes TC2 compare value to the desired initiation time, and sets TMRTRIGEN (optional)
- MCU sets CCABFRTX if one or more CCA's are required prior to transmi
- MCU sets SLOTTED if in slotted mode (2 CCA's will be attempted)
- MCU writes XCVSEQ=T
- Wait for TC2 match (if TMRTRIGEN=1)
- If CCABFRTX=1 :
  - SEQ_MGR executes RxWarmup
  - SEQ_MGR initiates CCA measurement (takes 128 µs)
  - if CCA indicates channel busy, sequence terminates, CCAIRQ & SEQIRQ issued
  - otherwise, if CCA indicates channel idle:
    - if SLOTTED=0, SEQ_MGR executes a RxPartialWarmdown
    - if SLOTTED=1, SEQ_MGR initiates a 2nd CCA, 320 µs after initiating 1st CCA.
    - if SLOTTED=1, and channel busy, sequence terminates, CCAIRQ & SEQIRQ issued
    - if SLOTTED=1, and channel idle, SEQ_MGR executes a RxPartialWarmdown
- If SLOTTED=1, SEQ_MGR waits until 320 µs elapse after 2nd CCA initiation
- SEQ_MGR executes TxPartialWarmup
- Preamble transmitted
- SFD transmitted
- FrameLength octet is obtained from Packet Buffer Address 0, and transmitted

- CRC engine is reset
- N additional octets are obtained from Packet Buffer and transmitted (N = FrameLength - 2)
- Each octet that is transmitted, is shifted through the CRC engine
- After Nth octet transmitted, FCS is available from CRC engine.
- FCS lower octet transmitted
- FCS upper octet transmitted
- SEQ_MGR issues TXIRQ
- SEQ_MGR executes TxWarmdown
- SEQ_MGR issues SEQIRQ

When Sequence T is initiated, the sequence manager first must determine whether one or more CCA measurements are required. If CCABFRTX bit is asserted, the sequence manager warms up the receiver (analog and digital elements). This process takes 144 µs. Immediately after the conclusion of the warmup, a CCA measurement is initiated. The CCA measurement takes 128 ms. If CCA indicates the channel is busy, the sequence manager warms down the receiver, issues CCAIRQ and SEQIRQ interrupts, and terminates the sequence. If, however, CCA indicates the channel is idle, the sequence manager inspects the SLOTTED bit to determine what to do next. If the SLOTTED bit is clear, the sequence manager issues a CCAIRQ interrupt, and then executes a partial warmdown of the receiver. If, however, the SLOTTED bit is set, the sequence manager does not issue a CCAIRQ interrupt; instead, the sequence manager initiates a second CCA measurement, precisely 320 µs after the initiation of the first CCA. This 320 µs interval is timed by the slot timer, which had been preloaded with 320 µs at the instant that Rx warmup was complete. At the conclusion of the second CCA, the sequence manager issues a CCAIRQ. If the second CCA indicates the channel is busy, the sequence manager warms down the receiver, issues a SEQIRQ interrupt, and terminates the sequence. If, however, CCA indicates the channel is idle, the sequence manager executes a warmdown of the receiver. Prior to executing the warmdown, the sequence manager reloads the slot timer, in order to precisely schedule the ensuing transmission. This time, the slot timer is loaded with a smaller value, in order to account for the fact that the timer must expire early in order to trigger the Tx warmup, which must complete at precisely the next backoff slot boundary. Thus, this time the slot timer is preloaded with:

Slot Timer Preload = 320 - TXWARMUPTIME + (2*TXDELAY).

The 320 µs is the duration of the backoff slot. The TXWARMUPTIME takes into account the fact that the TxAck timer must expire early for the Tx warmup to be complete at exactly the 320 µs point. The TXDELAY is a signed, fine-tune adjustment to the TxAck transmission time (± 62 µs).

If the SLOTTED bit is asserted, after the slot timer expires, the sequence manager executes a Tx warmup. The conclusion of this warmup coincides with the backoff slot boundary. If the SLOTTED bit is deasserted, the sequence manager does not wait for the slot timer, and instead executes the partial Tx warmup immediately after the partial Rx warmdown.

At this point, the sequence manager begins the transmit operation. The preamble and SFD are transmitted first. The preamble and SFD octets are generated by hardware, and are not included in the Packet Buffer. Then, the FrameLength octet is obtained from the Packet Buffer (address 0), and the buffer address pointer is incremented by one. The FrameLength octet is transmitted. The CRC engine is reset at this point. The FrameLength octet is used to determine how many more octets remain in the Packet Buffer. The number of remaining octets is:

Number of octets remaining in Tx buffer = FrameLength - 2

This is because the final 2 octets of the transmit frame constitute the FCS (Frame Check Sequence), and this value is hardware-computed. Thus, FCS is not part of the Tx buffer. Each of the octets in the Tx buffer is transmitted, and simultaneously shifted through CRC, to generate the FCS field. After the Tx buffer is drained and all of its octets transmitted, the 2 FCS octets are transmitted, starting with the least significant octet. At this point, the transmit operation is complete. The sequence manager issues a TXIRQ interrupt. The sequence manager then executes a complete Tx warmdown, and follows up with a SEQIRQ interrupt. The sequence manager returns to SEQ_IDLE state.

4.3.2.1.4 Sequence C (Standalone CCA)

Sequence C is the standalone CCA sequence. During Sequence C, the sequence manager executes a Clear Channel Assessment (CCA). The result of the CCA measurement is reported back to software in the CCA bit of the IRQSTS2 Register. The CCA bit indicates either a busy channel (1), or an idle channel (0).

The basic execution of a Sequence C is as follows:

- MCU writes TC2 compare value to the desired initiation time, and sets TMRTRIGEN (optional)
- MCU writes XCVSEQ=C
- Wait for TC2 match (if TMRTRIGEN=1)
- SEQ_MGR asserts one of {cca1_en, cca2_en, cca3_en, ed_en} to CCA_DIG
- SEQ_MGR executes RxWarmup
- SEQ_MGR waits an additional 26 µs for RSSI/AGC settling
- SEQ_MGR initiates CCA measurement by asserting rx_cca_en to CCA_DIG
- SEQ_MGR waits for CCA to complete
- CCA bit is set to the CCA status (1=busy 0=idle)

- CCAIRQ interrupt issued
- SEQ_MGR executes RxWarmdown
- SEQIRQ interrupt issued

When Sequence C is initiated, the sequence manager asserts one of 4 control signals to CCA_DIG, based on which CCA or ED mode is programmed into CCATYPE. The mapping of CCATYPE to control signal is shown in the following table:

CCATYPE	CCA Control Signal
00	ed_en
01	cca1_en
10	cca2_en
11	cca3_en

The asserted control signal will remain asserted for the duration of the Sequence C. The sequence manager warms up the receiver (analog and digital elements). This process takes 144 μs. The TC3 “bracketing” timer has no effect on the sequence manager during Sequence C, so it cannot abort the sequence. At the completion of the warmup, the sequence manager initiates the CCA by asserting the rx_cca_en signal to CCA_DIG. While CCA is enabled, the CCA_DIG measures and averages the energy or signal on the channel. At the completion of the CCA process, the CCA bit is set to the status of the channel (idle or busy). The CCAIRQ is issued, the receiver is warmed down, the SEQIRQ is issued, and the sequence manager returns to SEQ_IDLE state.

Any CCATYPE setting may be used for Sequence C.

4.3.2.1.5 Sequence TR (Transmit/Receive)

Sequence TR is a combination Transmit/Receive sequence. The sequence is executed as a concatenation of 1 transmit operation followed by 1 receive operation, with a minimum TX-to-RX turnaround time in between. There are 2 permutations of Sequence TR, depending on the RXACKRQD bit. For both permutations, the Sequence T that constitutes the first half of a Sequence TR, is identical to the standalone Sequence T (XCVSEQ=2). This means that the transmit operation can be slotted or unslotted, and can be preceded by 1 or 2 CCA measurements, depending on the state of the CCABFRTX and SLOTTED bits.

If RXACKRQD=0, then Sequence TR is executed as a Sequence T followed by a Sequence R, with a minimum TX-to-RX turnaround time in between. The Sequence R, which constitutes the second half of the Sequence TR, is identical to the standalone Sequence R (XCVSEQ=1). This means that the receive operation can be followed by an

automatic, hardware-generated transmit Acknowledge frame, if all the necessary conditions are met. As with basic Sequence R, data for a successful receive operation is always transferred to Packet Buffer.

If RXACKRQD=1, then Sequence TR is executed as a Sequence T followed by a Receive-Acknowledge-Only frame. This type of receive operation is special, and not the same as a standalone Sequence R. The Ack-only receive operation filters all incoming frames, looking only for an Acknowledge frame whose Sequence Number matches the Sequence Number that was transmitted in the Sequence T portion of the sequence. All non-matching frames are discarded, and after each non-matching frame, the sequence manager will return the receiver to preamble-detect mode. This receive operation will continue until the matching Ack frame is received. Because this is a Receive-Acknowledge-Only operation, and not a Sequence R, there will be no receive octets transferred to Packet Buffer (an exception is made if ACTIVE_PROMISCUOUS=1; if so, all receive octets are transferred to the Packet Buffer, even for the Receive-Acknowledge-Only frame). The sequence will end with a RXIRQ interrupt, which indicates that the matching Ack frame was successfully received.

Needless to say, if during the transmit operation of a Sequence TR, an Acknowledge frame is being requested of the receiving end device (Frame Control Field: AckRequest=1), then Sequence TR with RXACKRQD=1 is the appropriate procedure. (Using Sequence TR with RXACKREQ=0 is not recommended for this scenario, because there is no Sequence Number matching.)

Conversely, if during the transmit operation of a Sequence TR, an Acknowledge frame is not being requested of the receiving end device (Frame Control Field: AckRequest=0), then Sequence TR with RXACKRQD=1 should never be used, because a receive acknowledge frame will not be forthcoming. Instead, use Sequence TR with RXACKRQD=0, or simply Sequence T, if no followup receive frame is expected.

4.3.2.1.6 Sequence CCCA (Continuous CCA)

Sequence CCCA is the Continuous CCA sequence. This sequence is designed to accommodate situations where channel availability may be infrequent, or low-duty-cycle, and a device needs to be able to transmit at the earliest available opportunity. During Sequence CCCA, the sequence manager repeats CCA measurements continuously until a channel-idle condition is found. The interval between the end of one CCA measurement, and the start of the next, is 126 µs. The actual interval between consecutive CCA measurements (i.e., from measurement-start to measurement-start), in the CCCA sequence, is 126 + 128 = 254 µs. After each CCA iteration, the CCA bit (IRQSTS2 Register) is set to the result of the measurement. As long as the CCA bit is high (busy) at the end of the iteration, the sequence manager will initiate a new measurement. Unlike Sequence C (standalone CCA), there is no CCAIRQ interrupt generated at the end of

each CCA measurement, until the channel-idle condition is detected, at which point CCAIRQ is issued. The SLOTTED bit has no effect on this sequence. A Sequence CCCA will be aborted by the Sequence Manager if a TMR3 match occurs, and TC3TMOUT=1. CCA modes 1, 2, or 3, can be used for Continuous CCA.

The basic execution of a Sequence CCCA is as follows:

- MCU writes TC2 compare value to the desired initiation time, and sets TMRTRIGEN (optional)
- MCU writes XCVSEQ=CCCA
- Wait for TC2 match (if TMRTRIGEN=1)
- SEQ_MGR asserts one of {cca1_en, cca2_en, cca3_en} to CCA_DIG
- SEQ_MGR executes RxWarmup
- SEQ_MGR waits for 26 µs for RSSI/AGC settling
- SEQ_MGR initiates CCA measurement by asserting rx_cca_en to CCA_DIG
- SEQ_MGR waits for CCA to complete
- CCA bit is set to the CCA status (1=busy 0=idle)
- If CCA=1, initiate a new CCA measurement immediately and repeat the previous 4 steps
- If CCA=0, CCAIRQ is issued
- SEQ_MGR executes RxWarmdown
- SEQIRQ interrupt issued

The CCA bit (IRQSTS2 register) may change dynamically during the CCA measurement, and are considered valid (and sampled by the Sequence Manager), only at the end of the CCA measurement; therefore the user should not continuously poll CCA during CCCA and be expecting continuously-valid results.

The user can continuously monitor progress of the Sequence CCCA by reading the CCCA_BUSY_CNT register.

4.3.2.2 Sequence Manager Warmup Timing

The Sequence Manager not only controls autosequence timing, but also controls the warmup and warmdown timing of the digital and analog transceiver elements. The Sequence Managers executes warmups, in concert with the 3 lower-level managers:

- RX Digital Manager, or RSM
- TX Digital Manager, or TSM
- PLL Digital Manager, or PSM

For each low-level manager, the Sequence Manager has a single enable. The 3 manager-enables are:

- RXON (activates the RSM)
- TXON (activates the TSM)
- PLLON (activates the PSM)

After activation, the lower level managers take responsibility for carrying out the enabling and calibration of the individual transceiver blocks, both digital and analog. The lower level managers remain in their active states for as long as the Sequence Manager holds the corresponding manager-enable high. Normally, the Sequence Manager will hold the signals high for the duration of an operation (i.e., a packet reception, or a CCA measurement as part of a larger autosequence). After the Sequence Manager deasserts a manager enable, the lower-level manager shall return to its “off” or idle state, within 2 µs. In the case of an abnormal termination (i.e., a software abort, PLL unlock abort, or a TMR3 timeout), the Sequence Manager will deassert all manager enables, and all 4 state machines will return to idle. For aborts that occur during TX sequences, the Sequence Manager will perform a staged warmdown, to ensure the PLL and other analog elements remain stable while the PA gradually powers off. Aside from enabling and disabling of the lower-level managers by the top-level Sequence Manager, the 3 lower-level managers run autonomously of the top-level manager; there is no feedback, or other information, conveyed from the satellite managers to the Sequence Manager. All 4 managers run synchronously, using the same 2MHz clock. There are no clock domain crossings amongst them.

4.3.2.3 Dual PAN Mode Sequencing

The Sequence Manager supports Dual PAN mode, which allows the modem to reside on 2 networks simultaneously, switching back and forth between the 2 networks under software, or hardware control.

In Automatic Dual PAN mode, the modem must switch from one network to another under hardware control, at a software-programmed rate. Because the 2 networks often occupy different channels (frequencies), the sequence manager supports an “on-the-fly channel change”, whereby, during a Sequence R, in preamble-search state (RX_PRE), the sequence manager will toggle PLLON and switch to a new set of programmed channel-select registers, wait out a PLL settling time, and then resume preamble search on the new channel. The entire on-the-fly channel change consumes 68 µs, and 2 sequence manager states. The first state (RX_PAN1) deasserts PLLON and switches the PLL_INT and PLL_FRAC outputs to PHY_DIG to select the new PAN’s channel. The RX_PAN1 state also deasserts pll_lock_en so that unlock events that would naturally occur during the channel-change process, will not trigger a PLL_UNLOCK_IRQ or abort a sequence. The duration of RX_PAN1 is 4 µs. The Sequence Manager then transitions to RX_PAN2, which re-asserts PLLON. This initiates the PLL on the new channel, and PLL calibration

and locking begins (controlled by PSM). The pll_lock_en remains deasserted during RX_PAN2 so that unlock events continue to be ignored. The duration of RX_PAN2 is 52 µs. After RX_PAN2, the Sequence Manager returns to preamble-search (RX_PRE). Note: the RSM takes a few microseconds beyond this, to return to its preamble-search state, bringing the total “on-the-fly channel change” duration to 68 µs. This 68 µs represents a short “blind spot” during which preamble detection cannot occur, while the modem switches from one network to another. The following diagram depicts Dual PAN “on-the-fly channel change”.

4.4 Dual PAN ID introduction

The modem represents a high performance platform/radio that includes hardware support for a device to reside in two networks simultaneously. In optional Dual PAN mode, the device will alternate between the two (2) PANs under hardware or software control. Hardware support for Dual PAN operation consists of two (2) sets of PAN and IEEE addresses for the device, two (2) different channels (one for each PAN), a programmable timer to automatically switch PANs (including on the fly channel changing) without software intervention. There are control bits to configure and enable Dual PAN mode and read only bits to monitor status in Dual PAN mode. A device can be configured to be a PAN coordinator on either network, both networks or neither.

For the purpose of defining "PAN" in the content of Dual PAN mode, two (2) sets of network parameters are maintained. In this chapter, "PAN0" and "PAN1" will be used to refer to the two (2) PANs. Each parameter set uniquely identifies a PAN for Dual PAN mode. These parameters are described in the following table.

Table 4-5. PAN0 and PAN1 descriptions

PAN0	PAN1
Channel0 (PHY_INT0, PHY_FRAC0)	Channel1 (PHY_INT1, PHY_FRAC1)
MacPANID0 (16-bit register)	MacPANID1 (16-bit register)
MacShortAddrs0 (16-bit register)	MacShortAddrs1 (16-bit register)
MacLongAddrs0 (64-bit registers)	MacLongAddrs1 (64-bit registers)
PANCORDNTR0 (1-bit register)	PANCORDNTR1 (1-bit register)

During device initialization if Dual PAN mode is used, software will program both parameter sets to configure the hardware for operation on two(2) networks.

4.4.1 Manual and Automatic Modes

Two (2) modes are available for Dual PAN operation:

- Manual
- Automatic

4.4.1.1 Manual Dual PAN Mode

In manual Dual PAN mode software controls which PAN is selected for transmission and reception at all times. Software populates both PAN network parameter-set-registers listed in Table 4-5 where software selects a PAN to begin transmission of reception. A single control bit ACTIVE_NETWORK selects the PAN. To select PAN0, ACTIVE_NETWORK should be set to 0; to select PAN1, ACTIVE_NETWORK should be set to 1. Software then initiates an auto sequence by writing the appropriate value to XCVSEQ. At any point software can elect to switch to the alternate PAN. To switch, software aborts the sequence (if one is active) by writing XCVSEQ = IDLE. Software then toggles ACTIVE_NETWORK and restarts a new sequence with XCVSEQ. Software is responsible for aborting and initiating sequences. Hardware is responsible for warming up the transceiver on the selected channel (CHANNEL0 or CHANNEL1) and performing address-filtering on the selected network's address parameters. (PANCORDNTR, MacPanID, Short Address and/or IEEE address).Software overhead is significantly reduced in this mode because there is no need to re-program all of the PAN-select registers every time the PAN is toggled in Dual PAN mode where only a single bit (ACTIVE_NETWORK) needs to be written. For manual Dual PAN operation, DUAL_PAN_AUTO should be set to 0.

The diagram shown in the next figure depicts Dual PAN operation in manual mode. In this example, software populates both PAN parameter sets during initialization. Also, software elects to initiate packet reception on PAN1, so ACTIVE_NETWORK is set to 1. Software initiates the Sequence R by writing to XCVSEQ. Hardware warms up the receiver on CHANNEL1 and enters preamble-search. In this example no packet is received but a data request from higher level software requires a switch to PAN0. Software responds by aborting the reception by writing Sequence 1 to XCVSEQ. Software then sets ACTIVE_NETWORK to 0 to select PAN0. In the example, software initiates a Sequence TR. The hardware warms up the receiver for CCA on CHANNEL0, performs CCA (channel is idle), warms down the receiver and warms up the transmitter on CHANNEL0. Then, transmits the requested packet, warms down the transmitter and warms up the receiver again (as per the auto sequence) on CHANNEL0 in order to receive the acknowledge packet. The acknowledge packet is received and the auto

sequence completes successfully with the RXIRQ and SEQIRQ. Software elects to resume packet reception on PAN1) which was in progress before the data request). Software toggles ACTIVE_NETWORK back to 1 and re-initiates Sequence R.

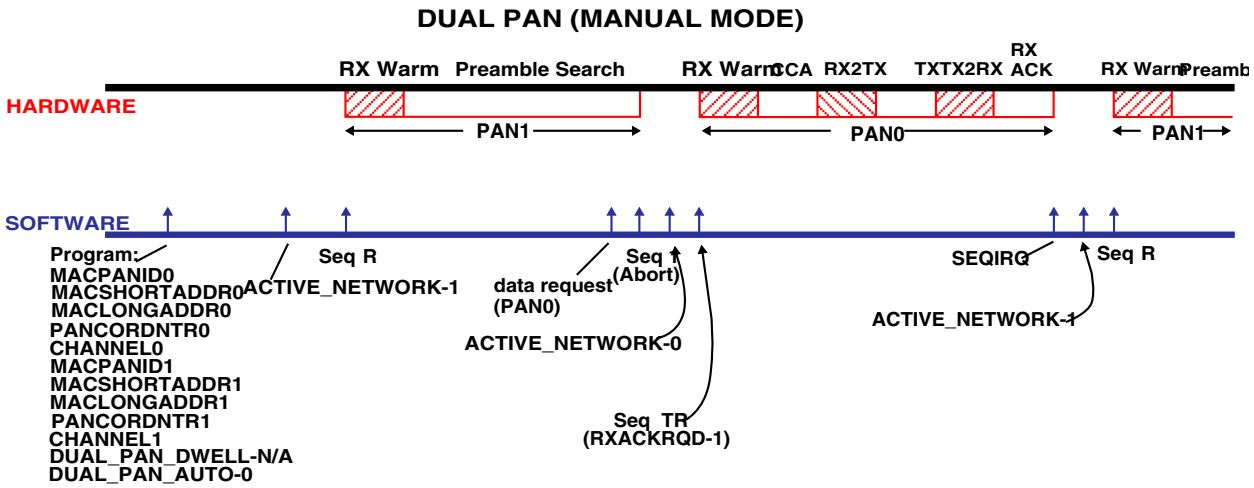


Figure 4-3. MCR20A Dual PAN (Manual Mode) hardware / software diagram

4.4.1.2 Auto Dual PAN

In automatic Dual PAN mode hardware is responsible for alternating the selected PAN at a pre-programmed interval. Software programs a "PAN dwell time", which determines the amount of time during which the hardware will receive on a given PAN. This is done by programming the DUAL_PAN_DWELL register. Software sets the DUAL_PAN_AUTO bit to "1" for automatic Dual PAN operation and sets ACTIVE_NETWORK to correspond to the PAN on which it desires packet reception to begin. Software then initiates a Sequence R. Hardware warms up the receiver on the selected PAN and begins preamble search. When the dwell time expires, assuming preamble and SFD have not been detected, hardware will switch to the alternate PAN including on-the-fly channel change. After it is switched to the alternate PAN, hardware will use the address filtering parameters (PANCORDNTR, MacPanID, Short Address and/or IEEE) programmed for the alternate PAN. After the dwell time expires on the alternate PAN, assuming preamble and SFD have not been detected, hardware will switch back to the original PAN. This process will repeat indefinitely until one of the following occurs:

1. A packet is successfully received (address match and CRC valid).
2. Software aborts the Sequence R.
3. A TMR3 RX timeout which automatically aborts the Sequence R.

4. A PLL unlock automatically aborts the Sequence R.

If the hardware PAN dwell timer expires during a packet reception (i.e., preamble and SFD have been detected during a Sequence R), the PAN-switch will be delayed until the packet is completely received. If the packet was received successfully (address match and CRC valid), the receive sequence will terminate with RXIRQ and SEQIRQ (if an auto-Acknowledge was requested and enabled, the sequence will complete after the Acknowledge packet was transmitted, and TXIRQ will also be set, in addition to RXIRQ and SEQIRQ); the PAN switch will occur at this point, after the Sequence has returned to IDLE. If the received packet fails FCS (CRC check), or, if the packet was not intended for the device (address mismatch), the hardware will toggle the PAN, and return to preamble-search on the new PAN.

If the hardware PAN dwell timer expires during a state *other* than RX preamble search during Sequence R (i.e., if expiration occurs during TX, CCA, ED, Sequence TR, or any warm-up or transitional state), PAN-switch will be delayed until either:

1. Preamble Search state is reached during a Sequence R (*not* Sequence TR).
2. Sequence returns to IDLE.

At any point, software can elect to interrupt the automatic Dual PAN operation, by aborting the sequence with a XCVSEQ=IDLE, and the setting the DUAL_PAN_AUTO bit to 0. In response, hardware will freeze the current state of the dwell timer, and capture the currently-selected PAN, so that hardware-controlled PAN alternation can resume where it left off, later. Software can then tend to that tasks that caused the Dual PAN interruption. When software wants to resume Dual PAN operation, it then sets DUAL_PAN_AUTO=1, and re-initiates Sequence R. Hardware will resume the dwell timer from the point it was frozen earlier, on the same PAN, and will warm-up the receiver to receive on the PAN that was being monitored before the interruption. Hardware will dwell on that PAN until the timer expires, and then switch PAN's (assuming no preamble and SFD detected, subject to the conditions listed above). Software can always determine the selected PAN at any time, in any mode, by reading CURRENT_NETWORK bit. If DUAL_PAN_AUTO=0, then CURRENT_NETWORK = ACTIVE_NETWORK. If DUAL_PAN_AUTO=1, then CURRENT_NETWORK is controlled by hardware.

To initiate auto Dual PAN operation for the first time, or to reset the dwell timer to its programmed value and allow software to select the initial PAN for auto Dual PAN operation, software should program the desired initial PAN to ACTIVE_NETWORK (0 for PAN0, 1 for PAN1), and then program the desired dwell time into the DUAL_PAN_DWELL register. Hardware always responds to a write to DUAL_PAN_DWELL, by resetting the dwell timer to its programmed value, and

selecting the initial PAN for auto Dual PAN operation, based on the state of ACTIVE_NETWORK. A write to DUAL_PAN_DWELL, always re-initializes the DWELL TIMER to the programmed value. If a write to DUAL_PAN_DWELL occurs during an auto sequence, the DWELL TIMER will begin counting down immediately. If a write to DUAL_PAN_DWELL occurs when there is no auto sequence underway, the DWELL TIMER will not begin counting until the next auto sequence begins; it will begin counting at the start of the auto sequence warm-up. Under all circumstances, DUAL_PAN_AUTO must be asserted to allow the DWELL TIMER to count. If DUAL_PAN_AUTO=0, the DWELL TIMER will freeze and hold its state, until DUAL_PAN_AUTO=1. However, a write to DUAL_PAN_DWELL will still initialize the DWELL TIMER to its programmed value, regardless of the state of DUAL_PAN_AUTO.

At any point during auto Dual PAN mode, software can ascertain the PAN currently being serviced by hardware by reading the CURRENT_NETWORK bit. Software can also determine the time remaining in the current dwell by reading the DUAL_PAN_REMAIN register.

In the auto Dual PAN mode, software retains responsibility for initiating and aborting sequences. There is no provision for hardware initiated sequences. TMR2 may be used to initiate an auto sequence at a future time. TMR3 may be used as an RX hardware abort mechanism.

The following figure depicts Dual PAN operation in automatic mode. In this example software populates both PAN parameter sets during initialization. Software wants to initiate packet reception in auto Dual PAN mode on PAN1, so ACTIVE_NETWORK is set to "1". For auto Dual PAN operation software programs the desired PAN-switching interval into DUAL_PAN_DWELL register, in this case 4 ms. Software initiates the Sequence R by writing to XCVSEQ. The dwell timer begins counting as soon as the receiver warm-up begins. Hardware warms up the receiver on CHANNEL1 where PAN was selected as the initial PAN shown in Figure 4-4 and begins preamble search. If a packet is received during this interval, hardware applies PAN1 address filtering. No preamble is detected at the point where the dwell timer expires so hardware executes a PAN-switch to PAN0 that includes an on-the-fly-change. On-the-fly changes require approximately 68 µs to execute and no preamble detection is possible during this "blind spot". After the PAN switch, hardware assumes preamble search on CHANNEL0 and applies PAN0 address filtering if a packet is received. No preamble is detected at the point which the dwell timer expires, therefore hardware performs a PAN-switch back to PAN1 and so on and so forth.

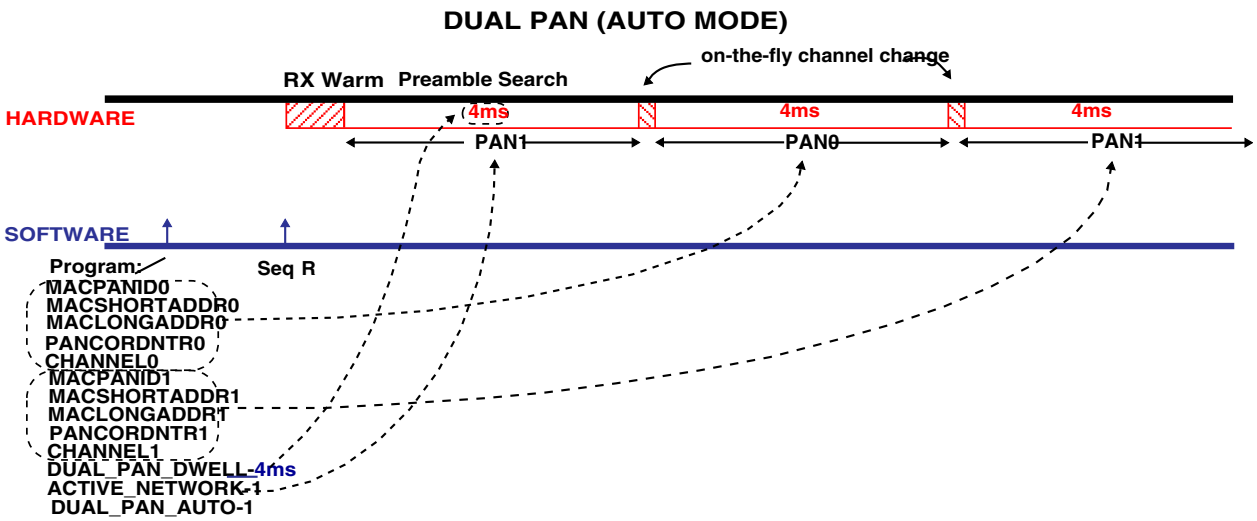


Figure 4-4. MCR20A transceiver dual PAN auto mode diagram

4.4.1.3 Two-PAN-one-channel

A special case occurs if a device is operating in Dual PAN mode (either Auto or Manual mode), and both PAN's occupy the same channel; in other words, CHANNEL0=CHANNEL1. In this case, hardware will detect this condition, and apply both sets of address-filtering parameters simultaneously. Hardware will provide Data Indication if the received packet matches either the PAN0 address parameter set {MacPanID0, MacShortAddrs0, MacLongAddrs0, PANCORDNTR0}, or the PAN1 address parameter set {MacPanID1, MacShortAddrs1, MacLongAddrs1, PANCORDNTR1}. At Data Indication, two status bits can be queried by software to quickly determine on which PAN the packet was received, RECD_ON_PAN0 or RECD_ON_PAN1. In the "Two-PAN-one-channel" scenario, it is possible for a packet to satisfy both sets of addressing parameters (PAN0 and PAN1). When this happens, both RECD_ON_PAN0 and RECD_ON_PAN1 will be set. In the "Two-PAN-one-channel" scenario, the CURRENT_NETWORK bit should be ignored, because both networks are simultaneously active.

If Dual PAN mode is not to be used, ACTIVE_NETWORK should be maintained at 0 (default) to select the PAN0 parameters for transceiving. In single PAN operation, the PAN1 parameter registers need not be programmed, and can be used as scratchpad. (PLL_INT1 and PLL_FRAC1 should not be programmed, and should instead be left in their default state, if Dual PAN mode is not in use).

4.4.2 Source Address Matching

This section describes interaction between Dual PAN mode and the modem packet Processors Source Address Matching function. The Source Address Matching Table is two (2) entries deep and 16 bits wide. In Dual PAN mode, entries from both networks will need to be stored in the table. It is highly undesirable to add a 17th bit to each table entry merely to identify the network on which the packet containing the Source Address was received. In order to keep the Source Address table width at 16 bits, the table will be split into two (2) sections:

- Lower section shall hold "PAN0" source address checksums
- Upper section shall hold "PAN1" source address checksums

In Dual PAN mode when software populates the table with a new Source Address Checksum, it will deposit the new checksum in the appropriate section of the table based upon whether the packet was received on PAN0 or PAN1. When the Packet Processor interrogates the table for a matching checksum, it will only search the section of the table corresponding to the PAN on which the current packet is being received. The dividing-line between the PAN0 and PAN1 sections in the table shall be programmability with a 4-bit register DUAL_PAN_SAM_LVL (SAM is an acronym for "Source Address Matching"). The register identifies the first PAN1 entry in the table (indices below this level apply to PAN0). For single-PAN operation, software should set this register to "12" or above which dedicates the entire table to PAN0. The hardware shall reset-default this register to "12". Figure 4-5 shows the effect of setting the PAN dividing line to "8":

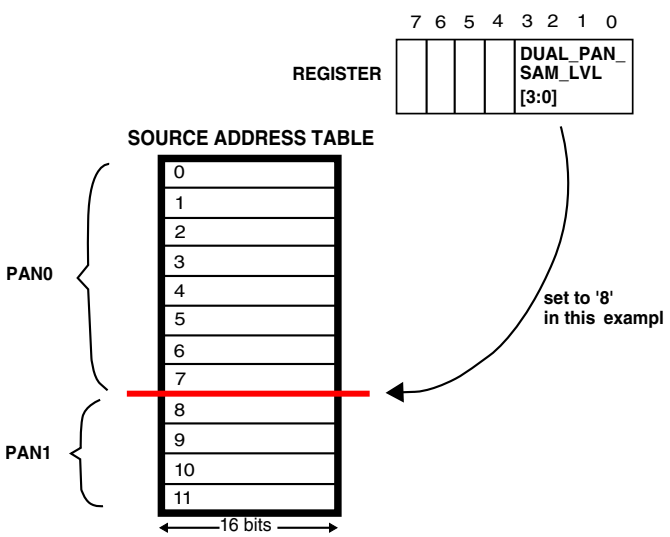


Figure 4-5. Source Address table set to "8"

4.4.3 Programming interface

This section describes registers associated with the Dual PAN ID. All writable registers read back what was previously written by software. Register addresses within the MCR20A tranceiver's memory map and bit assignments are shown below.

Table 4-6. MCR20A memory map and bit assignments

Register Name	Access Mode	Width	Description
DUAL_PAN_AUTO	rw	1	Activates automatic Dual PAN operating mode. In this mode, PAN-switching is controlled by hardware at a pre-programmed rate, determined by DUAL_PAN_DWELL. 1: Auto Dual PAN Mode 0: Manual Dual PAN mode (or Single PAN mode). Default. Whenever DUAL_PAN_AUTO=0, CURRENT_NETWORK=ACTIVE_NETWORK at all times. In other words, software directly controls which PAN is selected. Whenever DUAL_PAN_AUTO=1, CURRENT_NETWORK is controlled by HW
ACTIVE_NETWORK	rw	1	Selects the PAN on which to transceiver, by activating a PAN parameter set (PAN0 or PAN1). In Manual Dual PAN mode (or Single PAN mode), this bit selects the active PAN parameter set (channel and addressing parameters) that governs all auto sequences. In Auto Dual PAN mode, this bit selects the PAN on which to begin transceiving, latched at the point at which DUAL_PAN_DWELL register is written. 0: Select PAN0 (Default)
CURRENT_NETWORK	r	1	This read only bit indicates which PAN is currently selected by hardware in automatic Dual PAN mode. 1: PAN1 is selected 0: PAN0 is selected
DUAL_PAN_DWELL	rw	8	Channel Frequency Dwell Time. In Auto Dual PAN mode, hardware will toggle the PAN, after dwelling on the current PAN for the interval described below (assuming Preamble/SFD not detected). A write to DUAL_PAN_DWELL, always re-initializes the DWELL TIMER to the programmed value. If a write to DUAL_PAN_DWELL occurs during an auto sequence, the DWELL TIMER will begin counting down immediately. If a write to DUAL_PAN_DWELL occurs when there is no auto sequence underway, the DWELL TIMER will not begin counting until the next auto sequence begins; it will begin counting at the start of the sequence warm-up. **PRESCALER bits [1:0] (2 bits)** / **TIMEBASE** / **RANGE (min) - (max) (6 bits)** 00 — 0.5 ms — 0.5 - 32 ms 01 — 2.5 ms — 2.5 - 160 ms 10 — 10 ms — 10 - 640 ms 11 — 50 ms — 50 - 3.2 seconds

Table continues on the next page...

Table 4-6. MCR20A memory map and bit assignments (continued)

Register Name	Access Mode	Width	Description
			A write to DUAL_PAN_DWELL also causes the value of ACTIVE_NETWORK to get latched into the hardware. This latched value will be the starting point for the automatic dual-pan mode (i.e., start on PAN0 or on PAN1). The starting value takes effect immediately (if sequence is underway and DUAL_PAN_AUTO=1), or is otherwise delayed until sequence starts and DUAL_PAN_AUTO=1.
DUAL_PAN_REMAIN	r	6	This read only register indicates time remaining before next PAN switch in auto Dual PAN mode. The units for this register depend on the PRESCALAR setting in the DUAL_PAN_DWELL register according to the following table **DUAL_PAN_DWELL PRESCALAR** — **DUAL_PAN_REMAIN UNITS** 00 — 0.5 ms 01 — 2.5 ms 10 — 10 ms 11 — 50 ms The readback value indicates that between N-1 and N timebase units remain until the next PAN switch. For example, a DUAL_PAN_REMAIN readback value of 3, with a DUAL_PAN_DWELL PRESCALER setting of 2 (10ms), indicates that between 20ms (2*10ms) and 30ms (3*10ms), remain until the next automatic PAN switch.
RECD_ON_PAN0	r	1	Indicates the packet which was just received was received on PAN0
RECD_ON_PAN1	r	1	Indicates the packet which was just received was received on PAN1
DUAL_PAN_SAM_LVL	rw	4	For the Source Address Matching Table, sets the dividing line between PAN0 and PAN1 sections of the table. Table indices at or above this level belong to PAN1. Table entries below this level belong to PAN0. Default = 12 (entire table is PAN0).
CHANNEL0	rw	4	Channel selector for PAN0
MACPANID0	rw	16	MAC PAN ID for PAN0
MACSHORTADDRS0	rw	16	MAC Short Address fro PAN0
MACLONGADDRS0	rw	64	MAC Long Address fro PAN0
PANCORDNTR0	rw	1	1: Device is a PAN Coordinator on PAN0 0: Device is not a PAN Coordinator on PAN0
CHANNEL1	rw	4	Channel selector for PAN1
MACPANID1	rw	16	MAC PAN ID for PAN1
MACSHORTADDRS1	rw	16	MAC Short Address fro PAN1
MACLONGADDRS1	rw	64	MAC Long Address fro PAN1
PANCORDNTR1	rw	1	1: Device is a PAN Coordinator on PAN1 0: Device is not a PAN Coordinator on PAN1

4.5 Active Promiscuous Mode

This section describes the Active Promiscuous mode for the MCR20A transceiver high performance 2.4 GHz IEEE 802.15.4 radio.

4.5.1 Functional Description

The 802.15.4 Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low-Rate Wireless Personal Area Networks (WPAN's) prescribes the following behavior for Promiscuous mode operation.

In promiscuous mode, the MAC sublayer passes all frames received after the first filter directly to the upper layers without applying and more filtering or processing.

Where the first level is described as:

For the first level of filtering, the MAC sublayer shall disregard all received frames that do not contain a correct value in their FCS field in the MFR (see section 7.2.1.9 in the standard specification).

Promiscuous mode allows a device to receive and process all 802.15.4 frames that pass FCS check (CRC), regardless of whether or not they are addressed to the device. This allows a device to receive and process all valid traffic on the network. In Promiscuous mode, the modem packet processor will provide Data Indication on all packets. This capability allows a simple network sniffer to be constructed, which can display comprehensive packet data for all packets transmitted on a network within receiving range of the device.

Because a device operating in Promiscuous mode is designed to do so discretely, automatic acknowledgement of all received packets is inhibited regardless of the state of the ACK request in the Frame Control Field of the received packet. Otherwise, the sniffing device would be acknowledging received frames for which it was not the address.

In Active Promiscuous mode the packet processor will process packets according to the same rules that it applies to Promiscuous mode, except that it will automatically transmit an acknowledgement packet for frames that were addressed to the device. In other words, in Active Promiscuous mode if the received frame meets all the address and packet filtering requirements that would have been applied in non-promiscuous mode an acknowledgment packet will be transmitted; a Data Indication will still be provided for all packets (those that pass FCS check).

The packet processor can also be programmed to provide Data Indication for packets that fail FCS check. Received packets that fail FCS check are never acknowledged under any circumstances.

4.5.2 Special Handling for Broadcast Packets

Special rules apply to received packets that contain a broadcast PAN identifier (0xffff) and/or a broadcast short address (0xffff) according to the 802.15.4 standard.

Note

For valid frames that are not broadcast, if the Frame Type subfield indicates a data or MAC command frame and the Acknowledgement Request subfield of the Frame Control Field is set to one, the MAC sublayer shall send an acknowledgement frame.

An exception is made for received packets thatcontain a broadcast PAN identifier and a long address that matches the device's IEEE address. Such packets are acknowledged because the IEEE address uniquely identifies the device regardless of the PAN. This special case will also apply in "Active Promiscuous" mode where such packets will be acknowledged. The following table describes the "acknowledgement decision" for all permutations of broadcast indicators in the received packets addressing fields.

Table 4-7. Acknowledge-on-Broadcast (all permutations)

Destination Address Mode	PAN ID = |BROADCAST DSTADDR = |BROADCAST	PAN ID = |BROADCAST DSTADDR = BROADCAST	PAN ID = BROADCAST DSTADDR = BROADCAST	PAN ID = |BROADCAST DSTADDR = |BROADCAST
Short	ACK	NO ACK	NO ACK	NO ACK
Long	ACK	–	–	ACK

4.5.3 Special Handling of Rx Acknowledgement Frames

The modem features an auto sequence to automatically receive and verify Acknowledgment Frames after a Transmit Frame has been sent. This accomplished by programming a "Sequence TR with RXACKRQD=1". During normal operation the receive Acknowledgement Frames are not stored in the packet buffer. They are merely checked for matching Sequence Number and Frame Version with a Data Indication issued for a valid match on both. In Active Promiscuous mode, receive

Acknowledgement Frames will be stored into the Packet Buffer just like any other receive frame. This saves software from having to re-construct the contents of the Acknowledgement packet from the contents of the packet that was originally transmitted.

4.5.4 Programming Interface

This section describes registers associated with Active Promiscuous Mode. The definition and functionality of the PROMISCUOUS bit, which will be maintained for the modem is identical to existing FSL 802.15.4 transceivers. The ACTIVE_PROMISCUOUS bit is new for the modem as shown in the following table.

Table 4-8. ACTIVE_PROMISCUOUS register

Register Name	Access Mode	Width	Description
ACTIVE_PROMISCUOUS	rw	1	1: Provide Data Indication on all received packets under the same rules that apply in PROMISCUOUS mode, however acknowledge those packets under rules that apply in non-PROMISCUOUS mode 0: normal operation (Default)

4.6 Clock System

4.6.1 32 MHz Crystal Oscillator

The 32 MHz crystal oscillator (XTAL32m) is composed of the following features:

- Amplifier — it associates with an external crystal and tuning capacitors creating the main oscillator loop.
- Hysteresis comparator — it creates a logic-level signal from the sine wave developed across the crystal terminals.
- Clock gate timer — it waits for a delay before passing the clock signal.
- Clock divider that contains a divider by two and divide by eight circuitry.

The XTAL32m frequency is 32 MHz. It can be trimmed with some internal capacitances connected on both pads. The XTAL32m is supplied by the VBAT. An external clock generator can be connected to XTAL32Mout. In this case, the clock is directly connected to the output in bypass mode. A 1.8 V square clock can be connected as required.

The crystal oscillator block can work in 2 modes:

- Low power mode /RFoff mode where the comparator works with lower current. The clock is used only by the digital manager and provided to the clock output pin. This is the mode by default
- High performance mode/RFon mode where clocks are also sent to the RF part.

4.6.2 Clock output feature overview

The modem has the capability to output various clock frequencies as shown in this table by register selection. This feature allows the user to make adjustments per their application requirements.

- XTAL domain to minimize dynamic current consumption based on power mode selected.
- XTAL domain can be completely gated off (HIBERNATE mode)
- SPI communication allowed in HIBERNATE

Table 4-9. CLOCK_OUT Frequencies

CLK_OUT_DIV [2:0]	CLK_Out frequency	Comments
0	32 MHz	
1	16 MHz	
2	8 MHz	
3	4 MHz	DEFAULT if GPIO5=0
4	2 MHz	
5	1 MHz	
6	62.5 kHz	
7	32.786 kHz	DEFAULT if GPIO5=1

There will be an enable and disable bit for CLK_OUT. When disabling, the clock output will optionally continue to run for 128 clock cycles after disablement. There will also be one(1) bit available to adjust the CLK_OUT I/O pad drive strength. Bits described in the table above will reside in the 8 bit CLK_OUT_CTRL register. Default setting is 32.787kHz with a strap option to select a default of 4MHz.

Chapter 5
Modem: Advanced Security Module

5.1 Advanced Security Module

5.2 Introduction

The ASM engine encrypts using the Advanced Encryption Standard (AES). It can perform Counter (CTR), Cipher Block Chaining (CBC), and plain AES mode encryption. The combination of CTR and CBC modes of encryption is known as CCM mode encryption. CCM is short for Counter with CBC-MAC. CCM is a generic authenticate and encrypt block cipher mode. CCM is defined for use only with 128 bit block ciphers, such as AES.

5.2.1 Features

The ASM has the following features:

- CTR encryption.
- CBC encryption.
- AES encryption.
- Encrypts 128 bits as a unit.

5.2.2 Modes of Operation

The ASM is designed to be loaded with data and then started with a self-clearing START bit. Sixteen 8 bit registers of a key plus sixteen 8 bit registers of a counter plus sixteen 8 bit registers of text are necessary for Counter mode encryption. Cipher Block Chaining (CBC) mode needs only a key field and a text field programmed. Typically, only the text fields and counter fields need to be continuously written since the key field won't change.

The module has a built in self test. This test must be initiated by the software to make the module usable. Until this test is run and passes, the ASM module is disabled. Typical usage would require that this test be run once when the module is powered up. The software can re-run this test at any time it becomes necessary to verify the operation of the encryption engine.

If the test passes, the bit TSTPAS, in the ASM_CTRL2 register will be high. If the test fails the bit will be low and the module will disable itself from being used. This self test can be initiated by the software by writing a "1" to the SELFTST bit to set the mode to self test. Then write a "1" to the START bit. If self test passes then the TSTPAS bit will be set to a 1. If self test fails then the TSTPAS bit will be cleared to a 0 and the ASM module will be disabled and all outputs will be held to constant 0. Be sure and write a "0" to the SELFTST bit to get out of self test mode.

5.2.3 CTR mode block diagram

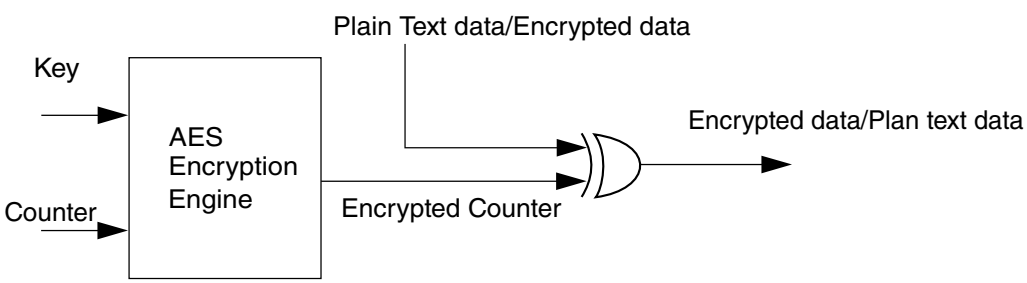


Figure 5-1. CTR Block

Counter mode encryption (CTR) is done by using the AES engine as shown in the block diagram above. The 128 bit key, text and counter value is written to the data register by configuring the ASM_CTRL2[7:5]. The START bit is set and the encrypted data can be read from the data register 11 clocks after the START bit is set by configuring the ASM_CTRL2[7:5]. The IRQSTS2 has an ASM_IRQ bit that the software can read to determine when the result is ready to be read. Decryption still uses the same AES encryption engine. The data to be decrypted is written to the data register. The decrypted result can be read from the data register after configuring the ASM_CTRL2 [7:5] .

5.2.4 CBC mode block diagram

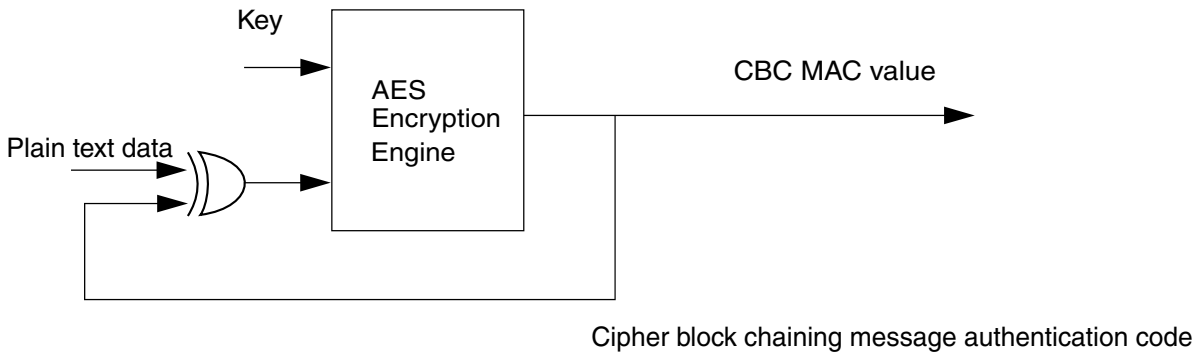


Figure 5-2. CBC Block

The CBC-MAC generation is done by using the AES engine as shown in the block diagram above. The 128 bit key and data is written to data register by configuring the ASM_CTRL2[7:5] . The start bit is set and the MAC value can be read from the data register 11 clocks after the start bit is set after configuring the ASM_CTRL2[7:5]. The ASM_IRQ bit can be read from the IRQSTS2 to determine when the result is complete.

5.2.5 CCM mode

CCM mode is the combination of using CTR mode for protecting the privacy of data, and using CBC mode to generate a MAC to protect the data from unauthorized modifications. In this mode both CTR and CBC bits in the ASM_CTRL1 are set and the START bit is set. The ASM first performs a CTR encryption followed by a CBC conversion. The results of CTR and CBC conversion are stored in separate registers which can be read by configuring the DATA_REG_TYPE_SELECT bits in ASM_CTRL2.

5.2.6 AES mode

AES mode is the most simple method of encryption in which there is no feedback of result to input in any form. The user simply writes the inputs required to the data registers and reads the result from the result register. For this mode, the user must clear both the CTR and CBC bits in the ASM_CTRL1 and set the AES bit followed by START bit to begin the conversion process. After 11 clock cycles the result is available in the result register, which can be read by configuring the DATA_REG_TYPE_SELECT bits from the ASM_CTRL2.

Note

The ASM block cannot do simple AES decryption.

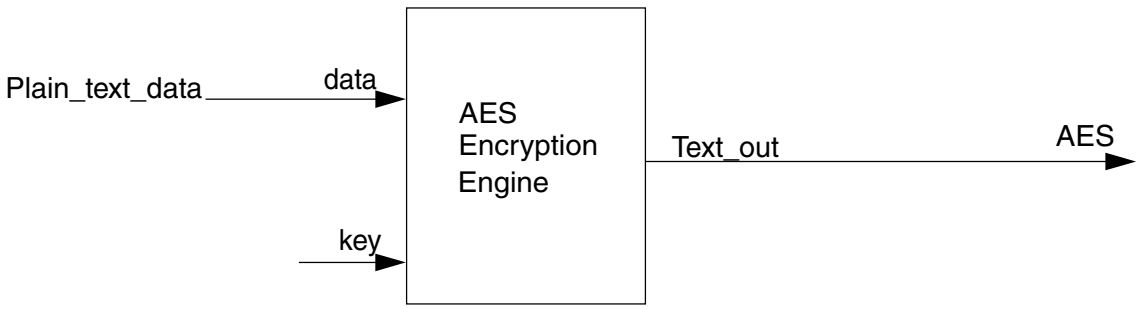


Figure 5-3. Plain AES Block

5.3 ASM module block diagram

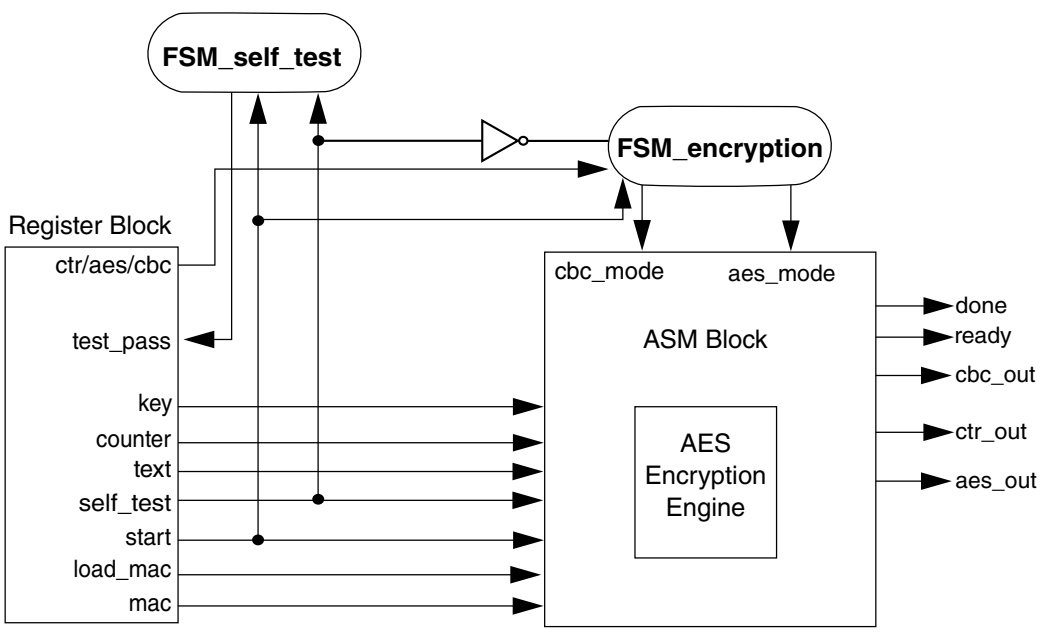


Figure 5-4. ASM Block Diagram

5.3.1 ASM Register Interface

The ASM register interface is used for writing and reading of the Control registers and Result registers of the module. There are separate result registers for the CTR and CBC results but use the same CBC result register for storing the result of plain AES mode of

encryption, therefore it is important that the AES result should be read as soon as the conversion is completed else it is overwritten if another CBC mode encryption is initiated before reading it.

5.3.2 AES Encryption Engine

The encryption engine does not have any knowledge of CBC, CTR, or CCM mode - it only performs simple AES encrytion on the text provided to it using the key value. There is a wrapper around the engine which enables the conversion of the plain cipher data to CTR, CBC, and so on. When a start pulse is received, the engine begins the conversion after exactly 11 clock cycles, then gives a data_ready pulse, followed by a level-signal complete to indicate that the conversion is finished. It is at this point that the data from the asm_engine should be latched to flip-flops because the data changes in subsequent clock cycles.

5.3.3 ASM Logic

There is a wrapper around the AES encryption engine which performs the conversion of plain AES cipher text output from the engine into CTR and CBC data according to the control signals. The scheme of the wrapper is shown logically in the MCR20A asm_logic block implementation diagram. The explanation for various modes is given below:

AES mode: The text written by the user is directly fed into the AES engine, and after conversion, the data is written directly to the result register. The CBC result is used to store the AES result and also to save 128 flip-flops area.

CTR mode: The data input to the engine is the counter value entered by the user, the output by the engine is XOR'ed with the text written by the user - this is then stored in the CTR result register.

CBC mode: The input plain text data is XOR'ed with the previous state output data. There is also an option of preloading the feedback data with some value using LOAD_MAC bit in the ASM_CTRL1 register. The new output data is directly stored in the CBC result register.

Self test mode: In this mode the CBC output is muxed back to the key input value.

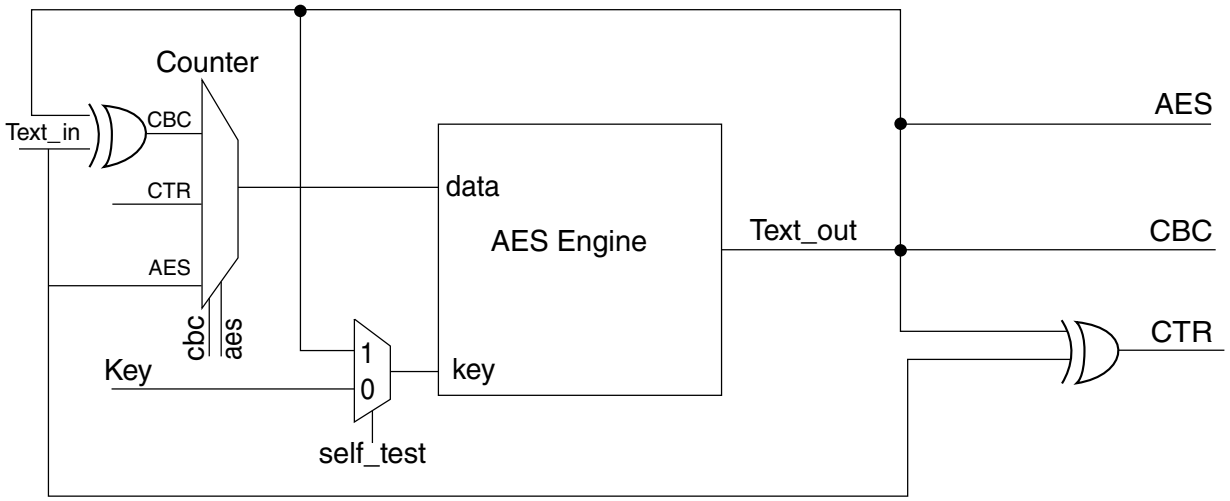


Figure 5-5. MCR20A asm_logic block implementation

5.3.4 AES encryption engine algorithm

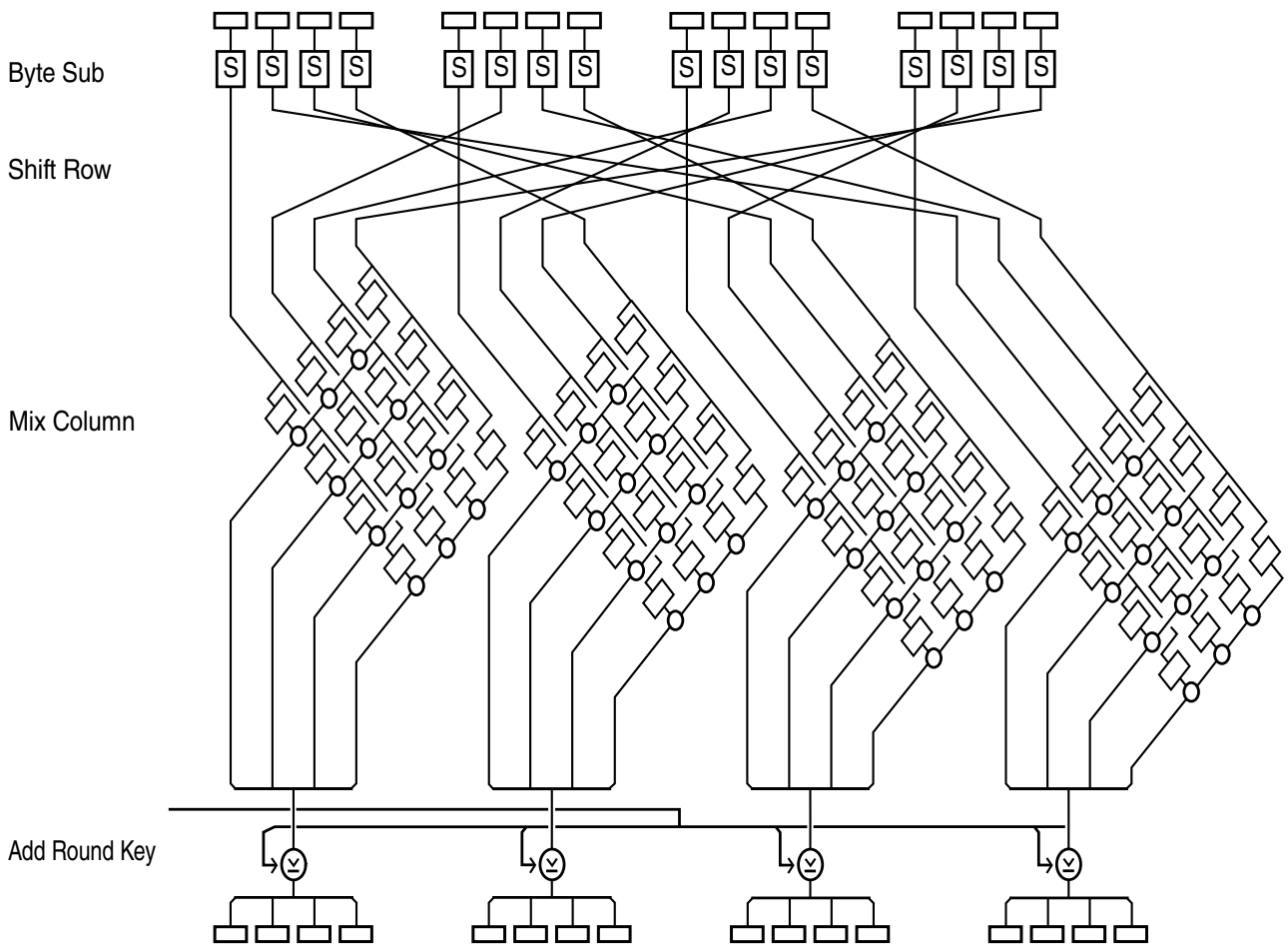


Figure 5-6. AES Encryption

The above diagram documents the encryption process for a single round. Each 128 bit data block is iterated though this structure 10 times. Each time though the round key is updated with a new value.

5.4 Counter mode encryption

11 clocks to complete

- Program ASM_CTRL2[7:5] to 3'b000. Write the key value in the data registers.
- Set the bit "CTR" in ASM_CTRL1. The "CTR" bit is static.
- Program ASM_CTRL2 [7:5] to 3'b001. Write the text value in data registers.
- Program ASM_CTRL2 [7:5] to 3'b010. Write the counter value in data registers.
- Set the START bit in ASM_CTRL1. The START bit is a self clearing bit.
- Poll for the ASM_IRQ bit to be set to a "1", or wait for the ASM_IRQ interrupt signal.
- Program ASM_CTRL2 [7:5] to 3'b011. Read the encrypted text from data registers.
- Repeat steps 3-7 for each 128 bits block.

5.5 AES mode encryption

11 clocks to complete

- Program ASM_CTRL2[7:5] to 3'b000. Write the key value in the data registers.
- Clear both CTR and CBC bit in ASM_CTRL1 and set AES bit.
- Program ASM_CTRL2 [7:5] to 3'b001. Write the text value in data registers.
- Program ASM_CTRL2 [7:5] to 3'b010. Write the counter value in data registers.
- Set the START bit in ASM_CTRL1. The START bit is a self clearing bit.
- Poll for the ASM_IRQ bit to be set to a "1", or wait for the ASM_IRQ interrupt signal.
- Program ASM_CTRL2[7:5] to 3'b110. Read the encrypted text from data registers.
- Repeat steps 3-7 for each 128 bits block.

5.6 Message Authentication Code generation (MAC)

11 clocks to complete

- Program ASM_CTRL2 [7:5] to 3'b000. Program the key value in registers
- Set the bit CBC in ASM_CTRL1.
- Program ASM_CTRL2 [7:5] to 3'b001.Program the text value in registers
- Set the START bit in ASM_CTRL1. The START bit is a self clearing bit.
- Poll for the ASM_IRQ bit to be set to a "1", or wait for the ASM_IRQ interrupt signal.
- Repeat steps 3-6 for each 128 bits block.
- When all blocks have been processed, program ASM_CTRL2 [7:5] to 3'b100 and read the final MAC (authentication code) from data registers.
- The MAC accumulator should be cleared to prepare for the next payload. Writing a 1 to the clear bit will set the MAC accumulator back to all 0. The clear bit is self clearing. If for some reason the software needs to continue a MAC calculation from a previously saved value, the MAC accumulator can be pre-loaded with a initial value. Program ASM_CTRL2 [7:5] to 3'b101 now the value you wish to load into the accumulator is written to data registers and then a 1 is written to the LOAD_MAC bit. The LOAD_MAC bit is self clearing.
- After self-test the CBC result register will have some initial value, so if the user wants to start a fresh transaction then the CBC result register has to be cleared by setting the CLEAR bit in the ASM_CTRL1 register. It will clear all the contents of the CTR and CBC result register.

Chapter 6
Modem: Interrupts

6.1 Introduction

Interrupts provide a way for the radio to inform the host microcontroller (MCU) of onboard events without requiring the MCU to query for status. The following sections describe the interrupt features and sources of the modem.

- 13 interrupt sources (interrupt status bits)
- Each interrupt source individually maskable
- Each interrupt source is individually write-1-to-clear.
- Interrupts remain asserted until cleared.

6.2 Modem Interrupt Sources

The radio has a single, active low, interrupt pin (IRQ_B) provided to the MCU. The interrupt pin is configured as actively-driven by the radio. Internally, the radio has 13 interrupt sources as shown in this table:

Table 6-1. Modem Interrupt Sources

Item	Interrupt Sources Status Bit	Mask Bit	Description	Interrupt Clear Mechanism
1	PLL_UNLOCK_IRQ	PLL_UNLOCK_MSK	PLL Unlock Interrupt. A '1' indicates an unlock event has occurred in the PLL.	write-1-to-clear
2	FILTERFAIL_IRQ	FILTERFAIL_MSK	RX Packet has Failed Filtering. A '1' indicates the most-recently received packet has been rejected due to elements within the packet. In Dual PAN mode, FILTERFAIL_IRQ applies to either or both networks, as follows:	write-1-to-clear

Table continues on the next page...

Table 6-1. Modem Interrupt Sources (continued)

Item	Interrupt Sources Status Bit	Mask Bit	Description	Interrupt Clear Mechanism
			A: If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=0, FILTERFAIL_IRQ applies to PAN0. B: If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=1, FILTERFAIL_IRQ applies to PAN1. C: If PAN0 and PAN1 occupy the same channel, FILTERFAIL_IRQ is the logical 'AND' of the individual PANs' Filter Fail status.	
3	RXWTRMRKIRQ	RX_WMRK_MSK	RX Byte Count Watermark Interrupt. A '1' indicates the number of bytes specified in the RX_WTR_MARK register has been reached.	write-1-to-clear
4	CCAIRQ	CCAMSK	CCA Interrupt. A '1' indicates the completion of a CCA operation.	write-1-to-clear
5	RXIRQ	RXMSK	RX Interrupt. A '1' indicates the completion of a receive operation.	write-1-to-clear
6	TXIRQ	TXMSK	TX Interrupt. A '1' indicates the completion of a transmit operation.	write-1-to-clear
7	SEQIRQ	SEQMSK	Sequence Complete Interrupt. A '1' indicates the completion of an autosequence. This interrupt will assert whenever the Sequence Manager transitions from non-idle to idle state, for any reason.	write-1-to-clear
8	PB_ERR_IRQ	PB_ERR_MSK	Packet Buffer Error Interrupt. 1: Packet Buffer Underrun Error has occurred since the last time this bit was cleared by software 0: Packet Buffer Underrun Error has not occurred since the last time this bit was cleared by software. **Note**: PB_ERR_IRQ will occur only when the SPI Packet Buffer access is BURST mode (not BYTE mode)	write-1-to-clear
9	WAKE_IRQ	WAKE_MSK	Wake Interrupt. Indicates either a Wake-from-POR or Wake-from-HIBERNATE event has occurred. The WAKE_FROM_HIB/TMRSTATUS bit indicates which type of wake event occurred, as long as WAKE_IRQ=1.	write-1-to-clear
10	TMR1IRQ	TMR1MSK	TMR1 Interrupt. Indicates T1CMP comparator value matched event timer counter.	write-1-to-clear

Table continues on the next page...

Table 6-1. Modem Interrupt Sources (continued)

Item	Interrupt Sources Status Bit	Mask Bit	Description	Interrupt Clear Mechanism
11	TMR2IRQ	TMR2MSK	TMR2 Interrupt. Indicates comparator value matched event timer counter. This flag is shared between the T2CMP (24-bit) and T2PRIMECMP (16-bit) compare registers.	write-1-to-clear
12	TMR3IRQ	TMR3MSK	TMR3 Interrupt. Indicates T3CMP comparator value matched event timer counter.	write-1-to-clear
13	TMR4IRQ	TMR4MSK	TMR4 Interrupt. Indicates T4CMP comparator value matched event timer counter.	write-1-to-clear

Each interrupt can be controlled by an interrupt mask. All interrupts are OR- combined with the external pin IRQ_B. The IRQ is issued when IRQ_B is set to 0.

Any or all of the interrupt sources, can be enabled to cause an assertion on IRQ_B. Each interrupt source has its own interrupt status bit in modems Direct Register space. Each interrupt source is individually maskable (each has its own MASK bit) so that each interrupt source can be individual enabled to trigger an assertion on IRQ_B.

There is also a global interrupt mask, TRCV_MSK (PHY_CTRL4 register), which can enable/disable all IRQ_B assertions by programming a single masking bit. All 13 interrupt status bits use a write-1-to-clear protocol. Interrupt status bits are not affected by reads. The IRQ_B pin is a level-sensitive interrupt indicator to the MCU; on an interrupt, IRQ_B will remain asserted until all active interrupt sources are cleared or masked.

6.3 Additional Interrupt Mask and Source Descriptions

Numerous register bits are provided to control interrupt behavior within the modem as shown in the table below. These are described in more detail in the subsequent sections and in the SPI Register Description chapter.

Table 6-2. Interrupt Control Summary

Field	R/W	Description	Default
TRCV_MSK	rw	1: Mask all Interrupts from asserting IRQ_B 0: Enable any Interrupt to assert IRQ_B	0
T1CMP[23:0]	rw	TMR1 compare value. If TMR1CMP_EN=1 and the Event Timer matches this value, TMR1IRQ is set.	0xFFF
T2CMP[23:0]	rw	TMR2 compare value. If TMR2CMP_EN=1 and the Event Timer matches this value, TMR2IRQ is set.	0xFFF

Table continues on the next page...

Table 6-2. Interrupt Control Summary (continued)

Field	R/W	Description	Default
T2PRIMECMP[15:0]	rw	TMR2-prime compare value. If TMR2CMP_EN=1 and TC2PRIME_EN=1 and the lower 16 bits of Event Timer matches this value, TMR2IRQ is set.	0xFF
T3CMP[23:0]	rw	TMR3 compare value. If TMR3CMP_EN=1 and the Event Timer matches this value, TMR3IRQ is set.	0xFFF
T4CMP[23:0]	rw	TMR4 compare value. If TMR4CMP_EN=1 and the Event Timer matches this value, TMR4IRQ is set.	0xFFF
TMR1CMP_EN	rw	1: Allow an Event Timer Match to T1CMP to set TMR1IRQ 0: Don't allow an Event Timer Match to T1CMP to set TMR1IRQ	0
TMR2CMP_EN	rw	1: Allow an Event Timer Match to T2CMP or T2PRIMECMP to set TMR2IRQ 0: Don't allow an Event Timer Match to T2CMP or T2PRIMECMP to set TMR2IRQ	0
TMR3CMP_EN	rw	1: Allow an Event Timer Match to T3CMP to set TMR3IRQ 0: Don't allow an Event Timer Match to T3CMP to set TMR3IRQ	0
TMR4CMP_EN	rw	1: Allow an Event Timer Match to T4CMP to set TMR4IRQ 0: Don't allow an Event Timer Match to T4CMP to set TMR4IRQ	0
TC2PRIME_EN	rw	1: Allow a match of the lower 16 bits of Event Timer to T2PRIMECMP to set TMR2IRQ 0: Don't allow a match of the lower 16 bits of Event Timer to T2PRIMECMP to set TMR2IRQ	0
RX_WTR_MARK[7:0]	rw	Receive byte count (octets) needed to trigger a RXWTRMRKIRQ interrupt. A setting of 0 generates an interrupt at end of the Frame Length field (first byte after SFD). A setting of 1 generates an interrupt after the first byte of Frame Control Field, etc.	0xFF

6.4 Functional Description

6.4.1 Interrupt Status Bit Structure

The modem has 13 interrupt sources, each represented in the register map by an interrupt status bit. All interrupt status bits share a common, generic structure. If a particular interrupt source is enabled, a "TRIGGERING EVENT" for that interrupt source will always set the status bit. A write-1-to-clear input from the SPI module will clear the status bit, but only if there is not a simultaneous triggering event. The triggering event will take priority (occur) over a software clear attempt if both events coincide.

The CLK and CLK_EN are provided by the SPI. This allows interrupt status bits to be read and cleared in the HIBERNATE state, when the modem's crystal oscillator is disabled and the SPI SCLK is the only "clock" available in the IC. The clock gate

guarantees that the status bit sees a clock only when either a triggering event occurs or a write-1-to-clear pulse arrives from the SPI. This reduces interrupt status bit power consumption to an absolute minimum. The common interrupt status bit structure is shown in this figure.

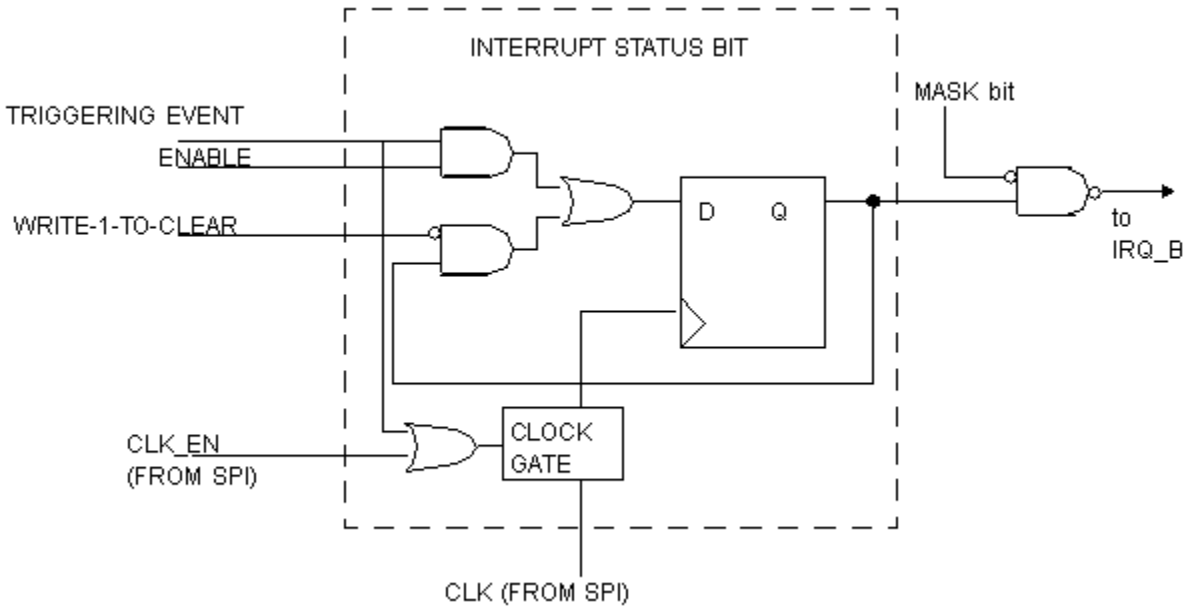


Figure 6-1. Common Interrupt Bit Structure

NOTE

For any modem interrupt source, if the triggering event occurs, the Interrupt Status Bit will be set regardless of the state of the corresponding MASK bit. If any of the 13 interrupts is to be ignored, software should set the corresponding MASK bit and apply the appropriate bit mask when reading the IRQSTS1, IRQSTS2, or IRQSTS3 register to mask out the unwanted status bit.

6.4.2 Clearing Interrupts

All interrupt status bits use a write-1-to-clear protocol. Writing a '1' to the interrupt status bit in any one of the IRQSTS1, IRQSTS2, or IRQSTS3 registers clears the offending interrupt. Writing a '0' to an interrupt status bit has no effect on the bit. Interrupt status bits are not affected by reads.

6.4.3 Timer Interrupts

The modem features a 24-bit Event Timer which runs at the 802.15.4 bit rate of 250 kHz. The modem has four timer interrupts (TMR1IRQ, TMR2IRQ, TMR3IRQ, and TMR4IRQ), each with its own 24-bit compare register (T1CMP, T2CMP, T3CMP, and T4CMP) and each with its own compare enable (TMR1CMP_EN, TMR2CMP_EN, TMR3CMP_EN, and TMR4CMP_EN).

For each timer compare enable:

- If the compare enable bit is set - A match on the respective 24-bit compare value to the Event Timer will cause the corresponding interrupt status bit to become set.
- If the compare enable is low - Event Timer matches will not cause the corresponding interrupt status bit to become set.

In addition, a 16-bit T2PRIMECMP compare value is provided along with a compare-enable bit TC2PRIME_EN.

- When TC2PRIME_EN is set and TMR2CMP_EN is also set - A match on T2PRIMECMP with the *lower 16 bits* of Event Timer will cause TMR2IRQ to become set rather than a full 24-bit compare.

6.4.4 PLL Unlock Interrupt

The PLL_UNLOCK_IRQ status bit indicates the PLL has come out of lock during a TX, RX or CCA transceiver operation. The modems Sequence Manager will begin monitoring for PLL unlocks only after a complete transceiver warm-up has occurred; unlocks that occur during warm-up will not cause a PLL unlock. A PLL unlock that occurs after the warm-up period will cause a sequence abort.

6.4.5 Filterfail Interrupt

The modems Packet Processor performs filtering on all received packets to determine whether the packet is intended for the device based on rules which if violated, will cause a Filterfail interrupt. A Filterfail interrupt will occur during packet reception when a packet fails filtering rules. On a Filterfail interrupt, the FILTERFAIL_CODE register can be read to ascertain more information about the nature of the filter failure.

NOTE

A Filterfail interrupt is not expected to be widely used in mission modes; it has been provided primarily for debug purposes.

NOTE

If a packet is received and First-Stage filtering fails (i.e., one or more of bits filterfail_code[0-3] is set, the Second-Stage filterfail_code bits, 3-9, should be ignored since the modem's packet processor will not know where to search for PAN ID and address fields in the incoming byte stream for illegal addressing modes.

This table indicates each Filterfail code bit and it respective reason the packet was rejected.

Table 6-3. Filterfail Codes

FILTERFAIL CODE BIT	REASON FOR FILTERFAIL
filterfail_code[0]	Fails Stage 1 Frame Length Checking (FL < 5 or FL > MAXFRAMELENGTH)
filterfail_code[1]	Fails Stage 1 Section 7.2.1.1.6 or Section 7.2.1.1.8 Checking (DST_ADDR_MODE or SRC_ADDR_MODE = 1)
filterfail_code[2]	Fails Stage 1 Section 7.2.1.1.5 Checking (Illegal PAN_ID_COMPRESSION Usage)
filterfail_code[3]	Fails Stage 1 Frame Version Checking
filterfail_code[4]	Fails Stage 2 Auto-RxAck Checking (Illegal Ack Frame Format in Sequence TR)
filterfail_code[5]	Fails Stage 2 Frame Type Checking (Incorrect Frame Filter Bit setting)
filterfail_code[6]	Fails Stage 2 Frame Length Checking (Illegal Beacon, Data, or Cmd FL)
filterfail_code[7]	Fails Stage 2 Addressing Mode Checking (Illegal Addressing Mode for Beacon, Data, or Cmd)
filterfail_code[8]	Fails Stage 2 Sequence Number Matching (Sequence TR Only)
filterfail_code[9]	Fails Stage 2 PAN ID or Address Checking (Beacon, Data, or Cmd)

6.4.6 RX Watermark Interrupt

During packet reception, a RX Watermark interrupt will occur when the number of received bytes matches the contents of the RX_WTR_MARK register minus 1.

NOTE

For the purpose of defining RX_WTR_MARK, the first byte received is the Frame Length field (PHR). The second byte received is the least-significant byte of Frame Control Field, etc.

For example, to cause an RX Watermark Interrupt to occur after the 10th received byte, set RX_WTR_MARK=11.

6.4.7 CCA Interrupt

The modems Clear Channel Assessment (CCA) has been completed; CCA interrupts occur at the end of both CCA and Energy Detect (ED) measurement intervals. For TX and Slotted (beacon-enabled networks) TX modes, the CCAIRQ is asserted indicating the CCA determined the channel is busy. The result of the CCA measurement is reported back to software in the CCA bit of the IRQSTS2 Register. The CCA bit indicates either a busy channel (1) or an idle channel (0).

6.4.8 RX Interrupt

RX interrupts occur at the end of the transceivers RX operation. An RXIRQ in combination with a SEQIRQ is considered a "Data Indication". RX interrupts are not generated on packets which fail FCS (CRC check) or fail packet filtering.

To receive a Data Indication (including RXIRQ) on packets which fail CRC, clear the CRC_MSK bit of the PHY_CTR2 register to 0. To inhibit packet filtering and enable Data Indication on packets which would otherwise fail packet filtering rules, set the PROMISCUOUS bit of the PHY_CTRL4 register to 1.

6.4.9 TX Interrupt

TX interrupts (TX_IRQ) occur at the end of the transceivers TX operation.

6.4.10 Sequencer Interrupt

The Sequencer Interrupt (SEQIRQ), indicates that an autosequence has completed and the Sequence Manager has returned to its idle state. A SEQIRQ will always occur at the end of an autosequence even if the autosequence terminated abnormally, such as a Software Abort, a TC3 Timeout or a PLL Unlock Abort.

NOTE

The ABORT_STS register can be read to determine which of these abnormal terminations occurred, if any (see the registers section for more details).

The SEQIRQ always occurs whenever the Sequence Manager transitions from non-idle to idle state. When SEQIRQ occurs, software can be sure that the Sequence Manager is in its idle state and a new sequence can be programmed immediately.

6.4.11 Interrupts from Exiting Low Power Modes

The modem has two low power modes which generate an 'Wake' interrupt. A Wake interrupt (WAKE_IRQ) occurs from either of these two conditions:

1. Exit from RESET
2. Exit from HIBERNATE

A transition out of RESET or HIBERNATE will always cause a WAKE_IRQ when unmasked. WAKE_IRQ is the only modem interrupt whose default state is "unmasked" so the MCU can expect this interrupt without any prior programming of the device. The MCU can interpret the WAKE_IRQ interrupt as a "Modem Ready" indication.

6.4.11.1 Exiting from RESET

The modem is put into reset and will stay in reset through the assertion of RST_B (reset bar or $\overline{\text{RESET}}$). In the interest of lowest power, there is no external pull-up resistor on input RST_B. As part of the MCU initialization, an internal GPIO is programmed as an output and then driven low to reset the modem. The RST_B input is asynchronous and needs to be held low for only a short period. In the reset condition, the modem is totally powered down and no clocks are available. Coming out of reset, the WAKE_IRQ interrupt status bit is set and causes an assertion on IRQ_B because WAKE_MSK is 0 by default. After the interrupt request is seen by the MCU, the modem is alive and ready for SPI transactions.

6.4.11.2 Exiting from HIBERNATE

The hibernate or low power mode is a specific mode where the XTAL is disabled. The SPI is capable of operation in modem low power modes, except Reset. Going out of hibernate mode through SPI means that the XTAL needs to be start up again, the digital regulator forced to high power mode and a new wake up interrupt (WAKE_IRQ) is generated as a result.

Operation in Hibernate mode allows most modem registers and the complete Packet Buffer to be accessed in radio's lowest-power operational state thereby, enabling minimal power consumption especially during the register-initialization phase of the IC's software.

In Hibernate, the SPI operates in "asynchronous mode". To write a register, the SPI module generates both clock and clock enable to the Register block since the crystal oscillator is disabled; for reads, no latching of read-back data takes place since the oscillator is disabled therefore, read-back data cannot change.

6.4.12 Packet Buffer Error Interrupt

A Packet Buffer Error Interrupt, PB_ERR_IRQ, indicates that a Packet Buffer Underrun Error has occurred. This condition can occur if a SPI burst read access to the Packet Buffer begins before the incoming packet has been completely received and the SPI Packet Buffer read address overtakes the RX demodulator write address.

Packet Buffer data obtained from a SPI transfer which results in a PB_ERR_IRQ has been corrupted and should be discarded. However, the packet contents have not been lost; the PB_ERR_IRQ should be cleared and the SPI transfer should then be restarted.

Chapter 7
Modem: Timer Information

7.1 Event Timer Block

The modem contains an internal Event Timer block that manages system timing.

The Event Timer consists of a prescaler and a 24-bit counter which increment whenever the crystal clock is operating (i.e. the modem is not in Hibernate mode). Interrupts to the MCU may be generated when the "current time" of the counter matches several pre-determined values set in registers via SPI write operations. The current time is accessible at any time via a SPI read operation as well as programmable via a SPI write operation.

The Event Timer provides the following functions:

- Timer to generate current system time
- Interrupt generation at pre-determined system times
- Abort an RX sequence at pre-determined system time
- Latches "timestamp" value during packet reception
- Initiates timer-triggered sequences

7.2 Event Timer Time Base

The Event Timer's base clock (tmr_clk) is derived from a programmable prescaler that is clocked by the 32 MHz crystal source. The prescaler provides counter input frequencies from 500 kHz down to 15.625 kHz, which sets the granularity and resolution of the current time. The prescaler, and thus the Event Timer only increment when the crystal oscillator is active. The field TMR_PRESCALE[2:0] at Indirect Access Register 0x28 establishes the tmr_clk frequency as shown in the following table.

Table 7-1. Event Timer Prescaler Settings

Indirect Access Register 0x28 TMR_PRESCALE[2:0]	Event Timer Time Base	Maximum Event Timer Duration
000	Reserved	Reserved
001	Reserved	Reserved
010	500 kHz	33.554 seconds
011 (default)	250 kHz	67.109 seconds
100	125 kHz	134.218 seconds
101	62.5 kHz	268.436 seconds
110	31.25 kHz	536.871 seconds
111	15.625 kHz	1073.742 seconds

The 24-bit counter automatically rolls over upon reaching its maximum value at the corresponding maximum Event Timer durations also provided in the table.

7.3 Setting Current Time

"Current Time" is defined as the value of the Event Timer internal counter. The current time is programmable, but does not have to be programmed. In the reset condition, the modem current time is set to zero. Current time advances from zero at the tmr_clk clock rate and rolls over to zero after reaching its maximum value.

Programming "current time" is accomplished by using four SPI registers:

1. T1CMP_LSB, Direct Register 0x17, T1CMP[7:0]
2. T1CMP_MSB, Direct Register 0x18, T1CMP[15:8]
3. T1CMP_USB, Direct Register 0x19, T1CMP[23:16]
4. PHY_CTRL4, Direct Register 0x07, Bit 2, TMRLOAD

When field TMRLOAD is programmed to high, the value of “current time” is set to the value in T1CMP[23:0]. Thus, T1CMP[23:0] is first programmed to the desired current time value, then TMRLOAD is programmed to 1, which initiates the timer load. The change to the “current time” value occurs within 3 crystal clock cycles, after which normal incrementing resumes on the next rising tmr_clk edge. TMRLOAD is not required to be programmed to zero for the Event Timer to resume normal operation. TMRLOAD is a write-only, self-clearing bit. Writing a 0 to TMRLOAD has no effect. The readback value of TMRLOAD is always 0.

7.4 Reading Current Time

The current value of the Event Timer can be read via the SPI by reading EVENT_TMR_LSB (Direct Register 0x0C) for EVENT_TMR[7:0], EVENT_TMR_MSB (Direct Register 0x0D) for EVENT_TMR[15:8], and EVENT_TMR_USB (Direct Register 0x0E) for EVENT_TMR[23:16].

The “current time” may be obtained using three single-byte SPI reads or a 3-byte SPI burst read (or as part of a longer burst read operation). It is important to realize that the Event Timer may increment during these SPI read operations or between successive SPI reads if single-byte SPI reads are used. To counter this during such an access, the modem latches the “current time” to protect the MCU from obtaining an incorrect value. The “current time” most significant 16 bits (MSB, USB) are latched when the least significant 8 bits (LSB) SPI location is read. This guarantees a stable value until the MCU completes a read of all 3 bytes constituting the “current time” before it is allowed to update. If such latching is undesired, it can be turned off by setting the EVENT_TMR_DO_NOT_LATCH bit to 1. The EVENT_TMR_DO_NOT_LATCH bit resides in the SEQ_MGR_CTRL register (Indirect Access Register 0x3A, bit 3).

NOTE

The preferred procedure to obtain the "current time" value from the modem is to perform a 3-byte burst read of the "current time" starting at the LSB address.

7.5 Latching the Timestamp

The modem has the ability generate a Timestamp, or to latch a copy of the “current time” while continuing to increment its internal counter. This timestamp value latched within the Event Timer corresponds to the beginning of a receive packet where the actual packet data begins after the SFD (Start-of-Frame-Delimiter) has been received. The timestamp[23:0] can be accessed via SPI by reading TIMESTAMP_LSB (Direct Register 0x0F) for timestamp[7:0], TIMESTAMP_MSB (Direct Register 0x10) for timestamp[15:8], and TIMESTAMP_USB (Direct Register 0x11) for timestamp[23:16]. The timestamp remains latched until another packet is received at which point the timestamp[23:0] value is updated and re-latched.

7.6 Event Timer Comparators

The modem incorporates four full 24-bit programmable fields that compare to the Event Timer's "current time". The intent of these compares are to enable the host to schedule events relative to the "current time". When a match between the "current time" and any one of the four timer compare values occurs, a corresponding flag is sent to internal interrupt logic. This causes the appropriate bit in the IRQSTS3 register (Direct register 0x2) to be set and depending on the interrupt mask control bit, generate an interrupt event on the IRQ_B pin.

7.6.1 Timer Compare Fields

There are four 24-bit timer compare fields and one 16-bit timer compare field:

1. T1CMP[23:0], (T1CMP_LSB, T1CMP_MSB, T1CMP_USB), Direct Address 0x17, 0x18, 0x19 respectively.
2. T2CMP[23:0], (T2CMP_LSB, T2CMP_MSB, T2CMP_USB), Direct Address 0x1A, 0x1B, 0x1C respectively.
3. T3CMP[23:0], (T3CMP_LSB, T3CMP_MSB, T3CMP_USB), Direct Address 0x12, 0x13, 0x14 respectively.
4. T4CMP[23:0], (T4CMP_LSB, T4CMP_MSB, T4CMP_USB), Direct Address 0x1D, 0x1E, 0x1F respectively.

Additionally, there is a special 16-bit timer compare field for situations where the upper byte of EVENT_TMR is a 'don’t care' case.

1. T2PRIMECMP[15:0], (T2PRIMECMP_LSB, T2PRIMECMP_MSB), Direct Address 0x15 and 0x16 respectively.

The TMR2IRQ status bit can be set by a match to T2CMP or T2PRIMECMP:

- If register bit TC2PRIME_EN=1 (PHY_CTRL4 register, Direct Address 0x7, bit 0), then TMR2IRQ becomes set by a match to T2PRIMECMP (16-bit compare).
- If TC2PRIME_EN=0, then TMR2IRQ becomes set by a match to T2CMP (24-bit compare).

7.6.2 Timer Compare-Enable Bits

Each timer comparator has a enable bit that enables or disables the compare function. To enable, write a “1” to the corresponding comparator. The default condition is the timer disabled (reset to “0”):

1. TMR1CMP_EN, PHY_CTRL3 Register, Direct Address 0x05, bit 4
2. TMP2CMP_EN, PHY_CTRL3 Register, Direct Address 0x05, bit 5
3. TMR3CMP_EN, PHY_CTRL3 Register, Direct Address 0x05, bit 6
4. TMR4CMP_EN, PHY_CTRL3 Register, Direct Address 0x05, bit 7

If a timer comparator is enabled using its associated bit, the corresponding interrupt status bit (TMRxIRQ) will be set by a timer compare match to its respective TxCMP register. If a timer comparator is disabled using its associated bit, the corresponding interrupt status bit (TMRxIRQ) will not be set by a timer compare match to its respective TxCMP register.

7.6.3 Timer Interrupt Status Bits

When enabled, all four fields can be continuously compared to the current value of the Event Timer counter. When a match occurs, the following corresponding internal status flags assert:

1. TMR1IRQ, IRQSTS3 Register, Direct Address 0x02, bit 0
2. TMR2IRQ, IRQSTS3 Register, Direct Address 0x02, bit 1
3. TMR3IRQ, IRQSTS3 Register, Direct Address 0x02, bit 2
4. TMR4IRQ, IRQSTS3 Register, Direct Address 0x02, bit 3

The four interrupt status bits are write-1-to-clear. The status bit remains set until a 1 is written to the bit location in the IRQSTS3 register. Writing 0 to the bit or reading the bit will not change its state.

7.6.4 Timer Interrupt Masks

When a comparator match occurs and the internal status flag asserts, the following interrupt masks can enable an interrupt on the IRQ pin:

1. TMR1MSK, IRQSTS3 Register, Direct Address 0x02, bit 4
2. TMR2MSK, IRQSTS3 Register, Direct Address 0x02, bit 5
3. TMR3MSK, IRQSTS3 Register, Direct Address 0x02, bit 6
4. TMR4MSK, IRQSTS3 Register, Direct Address 0x02, bit 7

If the interrupt mask is set to “1” (masked), the respective interrupt status bit, TMRxIRQ, will not cause an interrupt on the IRQ_B pin. If the interrupt mask is set to “0” (enabled), the respective interrupt status bit, TMRxIRQ, will cause an interrupt on the IRQ_B pin. Timer mask bits, TMRxMSK, have no effect on the state of the TMRxIRQ interrupt status bits; they only allow (or prevent) the respective TMRxIRQ bit from generating an interrupt on IRQ_B.

7.6.5 Setting Compare Values

Because the primary timer compare fields are 24-bit values, they are each shared across 3 sequential SPI register addresses. The timer compare value can be changed using 3 single-byte SPI writes, one 3-byte SPI burst write or as part of a longer multi-byte write operation.

Note

Not all bits of the timer compare value are updated simultaneously by the SPI. To prevent the Event Timer from generating a false match to a partially updated timer compare value, software should clear the respective TMRxCMP_EN bit prior to changing the timer compare register TxCMP. Once TxCMP has been updated, software can re-enable the respective TMRxCMP_EN bit.

7.7 Intended Event Timer Usage

It is intended that the system use the "current time" value and the timer compare functions of the Event Timer to schedule system events, including:

- Generating time-based interrupts
- Abort an RX operation
- Triggering transceiver operations

7.7.1 Generating Time-Based Interrupts

Generating time-based interrupts is accomplished by setting timer compare values relative to the "current time" allowing the Event Timer counter to increment until a timer compare match is generated, and using this match to generate an interrupt to the host. The general procedure is as follows:

1. Disable the timer compare.
2. Enable the timer compare interrupt mask.
3. Read the "current time" value from event_tmr[23:0].
4. Add an offset to the "current time" value to equal the desired "future time".
5. Program the appropriate timer_compare value (TxCMP) to "future time".
6. Program the appropriate TMRxCMP_EN bit to enable the compare.
7. Allow a timer compare match to set the status register bit and generate an interrupt (see the Timer Interrupt Status Bits section). The appropriate internal status register bit is always set upon a TxCMP match. An external interrupt is generated when the corresponding SPI interrupt mask bit, TMRxMSK, is clear.
8. Program the appropriate TMRxCMP_EN bit to disable the compare function. If this is not done, the compare function will continue to run and generate another interrupt every time the counter rolls over and again matches the comparator.

7.7.2 Using T3CMP to Abort an RX operation

The Event Timer provides a timer-based mechanism to automatically abort an RX operation. The modem is put into an RX operation when XCVSEQ[2:0] (PHY_CTRL1 register, Direct Address 0x03) is set to 0x01. An RX autosequence will commence and will begin a search for preamble. Software can program, in advance, a hardware timeout to occur such that the RX autosequence can be automatically aborted by hardware after a certain amount of time elapses without a packet being detected.

The general procedure is as follows:

1. Read the “current time” value from EVENT_TMR[23:0].
2. Add an offset to the “current time” value to equal desired “future time” to abort the RX operation.

3. Program register T3CMP[23:0] to value “future time”.
4. Program TC3TMOUT, (PHY_CTRL4 Register, Direct Address 0x07, bit 6) to 1, to enable the hardware aborting of the RX operation.
5. Program XCVSEQ=1, to start the RX autosequence.
6. When “current time” equals T3CMP[23:0], the modem aborts the RX autosequence and the Sequence Manager returns to SEQ_IDLE state. The SEQIRQ interrupt status bit (IRQSTS1 Register, Direct Address 0, bit 0) will be set, and the TMR3IRQ interrupt status bit will also be set. The RXIRQ interrupt status bit (IRQSTS1 Direct Address 0, bit 2) will not be set since no packet was received before the timeout.

7.7.3 Using T2CMP or T2PRIMECMP to Trigger Transceiver Operations

The Event Timer provides a timer-based mechanism to automatically launch a transceiver autosequence. Using this method, a sequence can be “scheduled” for time in the future, and the MCU can be put into a reduced power state, without the need to wake up at the appropriate time to start the autosequence.

The general procedure is as follows:

1. Read the “current time” value from EVENT_TMR[23:0].
2. Add an offset to the “current time” value to equal desired “future time” to launch the autosequence.
3. For either a 24-bit or a 16-bit compare:
   - For a 24-bit compare: program register T2CMP[23:0] to value “future time”, and set TC2PRIME_EN=0.
   - For a 16-bit compare: program register T2PRIMECMP[15:0] to value “future time”, and set TC2PRIME_EN=1.
4. Set TMRTRIGEN=1 and program the desired autosequence into XCVSEQ[2:0]. Both TMRTRIGEN and XCVSEQ fields reside in the PHY_CTRL1 register (Direct Address 0x03).
5. When “current time” equals T2CMP[23:0] (for a 24-bit compare), or T2PRIMECMP[15:0] (for a 16-bit compare), the modem will launch the scheduled autosequence in accordance with the XCVSEQ[2:0] field. The TMR2IRQ interrupt status bit will become set.

6. The MCU can wake up at this point (if TMR2MSK=0), or can elect to continue in low-power mode until the autosequence completes, at which time SEQIRQ will become set. If SEQMSK=0, then SEQIRQ will interrupt the MCU at the completion of the transceiver operation.

Chapter 8
Modem SPI Interface

8.1 Modem SPI Overview

The modem's SPI interface allows an MCU to communicate with the modem's register set and Packet Buffer. The modem's SPI is a slave-only interface; the MCU must drive R_SSEL_B, R_SCLK and R_MOSI. Write and read access to both Direct and Indirect registers is supported, and transfer length can be single-byte, or bursts of unlimited length. Write and read access to the Packet buffer can also be single-byte, or a burst mode of unlimited length. The modem's SPI interface is asynchronous to the rest of the IC. No relationship between R_SCLK and the modem's internal oscillator is assumed. And no relationship between R_SCLK and the CLK_OUT pin is assumed. All synchronization of the SPI interface to the modem IC takes place inside the SPI module, alleviating the burden on other block designers to provide this synchronization internal to their modules. SPI synchronization takes place in both directions: SPI-to-IC (register writes), and IC-to-SPI (register reads). The SPI is capable of operation in all modem Power Modes, except Reset. Operation in Hibernate mode allows most modem registers, and the complete Packet Buffer, to be accessed in the modem's lowest-power operational state, enabling minimal power consumption, especially during the register-initialization phase of the IC. The SPI design features a compact, single-byte control word, reducing SPI access latency to a minimum. Most SPI access types require only a single-byte control word, with the address embedded in the control word. During control word transfer (the first byte of any SPI access), the contents of the IRQSTS1 register (the modem's highest-priority status register) are are always shifted out, so that the MCU gets access to IRQSTS1, with the minimum possible latency, on every SPI access.

8.2 Modem SPI Basic Operation

The modem operates as a SPI slave only. The microcontroller supplies the interface clock and acts as SPI master.

8.2.1 SPI Pin Definition

8.2.1.1 Radio Select SEL (R_SSEL_B)

The MCU Master SPI selects and activates the modem's Slave SPI by asserting R_SSEL_B (low).

8.2.1.2 Radio SPI Clock (R_SCLK)

The MCU Master SPI provides a serial bit clock to the modem's Slave SPI on R_SCLK.

8.2.1.3 Radio MOSI - Master Data Out Slave Data In (R_MOSI)

The MCU Master SPI transmits Control, Address and Write data to the modem's SPI Slave on R_MOSI.

8.2.1.4 Radio MISO - Master Data In, Slave Data Out (R_MISO)

The modem's Slave SPI transmits readback data to the MCU Master SPI on R_MISO.

8.2.1.4.1 Setting R_MISO Off Impedance

MISO_HIZ_EN - Determines the output state of the modem's R_MISO output when R_SSEL_B is deasserted (high).

1: R_MISO output is tristated (default)

0: R_MISO output is driven low by the SPI slave

8.2.1.4.2 R_MISO Pull-up Requirement

Because the modem tri states its R_MISO output when a SPI transfer is not in progress (R_SSEL_B deasserted), a pull-up is required to ensure that the SPI Master does not see a floating MISO input.

In a configuration consisting of only a single Master (MCU) and single Slave (the modem), this pull-up can be an internal pull-up on the MCU's MISO input. Moreover, after initialization, the modem's MISO_HIZ_EN bit can be cleared (enabling the modem to drive R_MISO at all times), and the MCU pull-up can be disabled.

In a configuration consisting of more than just 1 SPI master and 1 SPI slave, the pull-up on R_MISO should be external, and the MISO_HIZ_EN bit should be left in its default state.

8.2.2 SPI Timing

8.2.2.1 SPI Transfer Protocol

The modem's SPI is a slave-only interface, and follows a CPHA=0 and CPOL=0 protocol. Taken together, CPHA and CPOL determine the clock polarity and clock edge used to transfer data between the SPI master and slave, in both communication directions. CPHA=0 (clock phase) indicates that data is captured on the leading edge of SCK and changed on the following edge. CPOL=0 (clock polarity) indicates the inactive state value of R_SCLK is low. All data is transferred MSB-first. The following diagram depicts the transfer protocol of the modem's SPI.

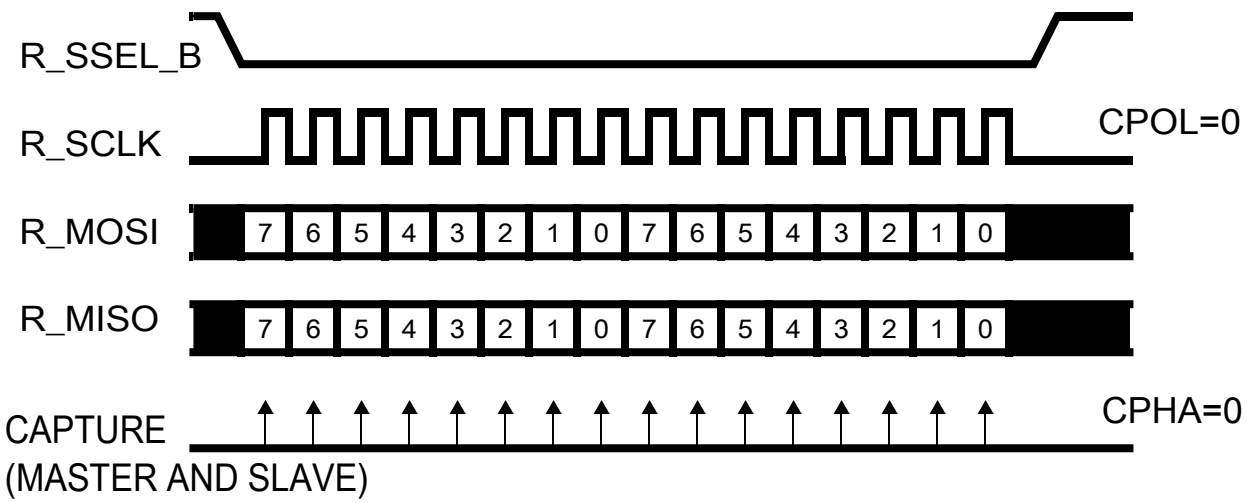


Figure 8-1. Transfer Protocol of the modem's SPI

8.2.2.2 SPI Timing: R_SSEL_B to R_SCLK

The following diagram describes timing constraints that must be guaranteed by the system designer.

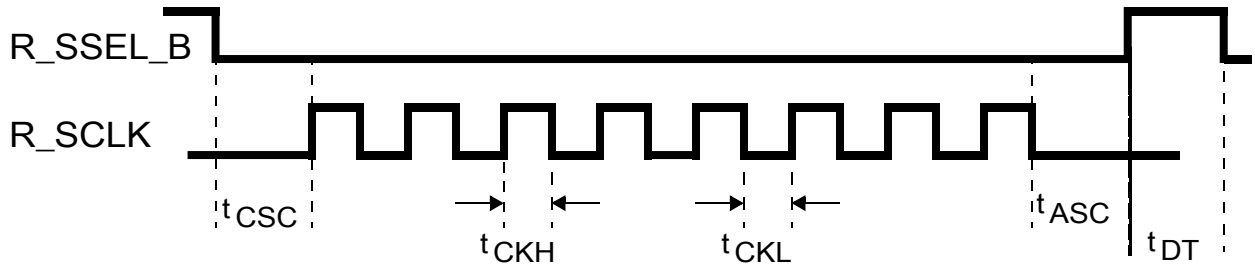


Figure 8-2. SPI Timing of R_SSEL to R_SCLK

tCSC (CS-to-SCK delay): 31.25ns

tASC (After SCK delay): 31.25ns

tDT (Minimum CS idle time): 62.5ns

tCKH (Minimum SCLK high time): 31.25ns (for SPI writes); 55ns (for SPI reads)

tCKL (Minimum SCKH high time): 31.25ns (for SPI writes); 55ns (for SPI reads)

NOTE

The SPI Master device shall only deassert R_SSEL_B on byte boundaries, and only after guaranteeing the tASC constraint shown above.

8.3 SPI Transactions

8.3.1 SPI Control Word

Every SPI transaction begins with one single-byte control word. The control word consists of the address, direction (write or read), transaction target (register or Packet Buffer), and for the Packet Buffer, the access mode (burst or byte). For most transactions, data transfer follows immediately after the control word. However, for Indirect Access Register transactions, and for Packet Buffer byte-mode transactions, an additional address byte follows the control word, before data transfer begins. Bit 7 of the control word selects the transfer direction (1=READ, 0=WRITE). Bit 6 selects the transfer target (1=PACKET BUFFER, 0=REGISTER). For Register accesses, the remaining bits select the register address. For Packet Buffer Access, bit 5 selects the access mode (1=BYTE, 0=BURST). For Packet Buffer Access, the remaining bits in the control word are reserved and ignored by the modem. Details and examples of control word usage appear in the following sections. The following table depicts an overview of the control word.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Access Mode	Access Type
1	0	Register address [5:0]						Register Access	Read
0	0	Register address [5:0]							Write
1	1	0	Reserved					Packet Buffer Burst access	Read
0	1	0	Reserved						Write
1	1	1	Reserved					Packet Buffer Byte access	Read
0	1	1	Reserved						Write

Figure 8-3. Control Word Overview

8.3.2 Direct Register Write Access (single byte)

The following diagram depicts a single-byte write access to a Direct Register in the modem.

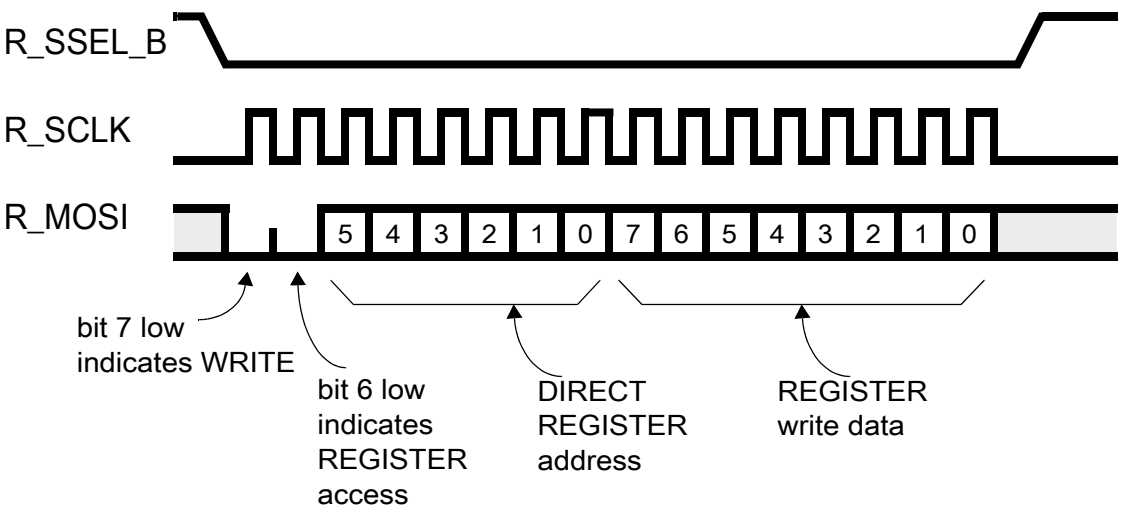


Figure 8-4. Single-Byte Write Access to a Direct Register

8.3.3 Direct Register Read Access (single byte)

The following diagram depicts a single-byte read access to a Direct Register in the modem.

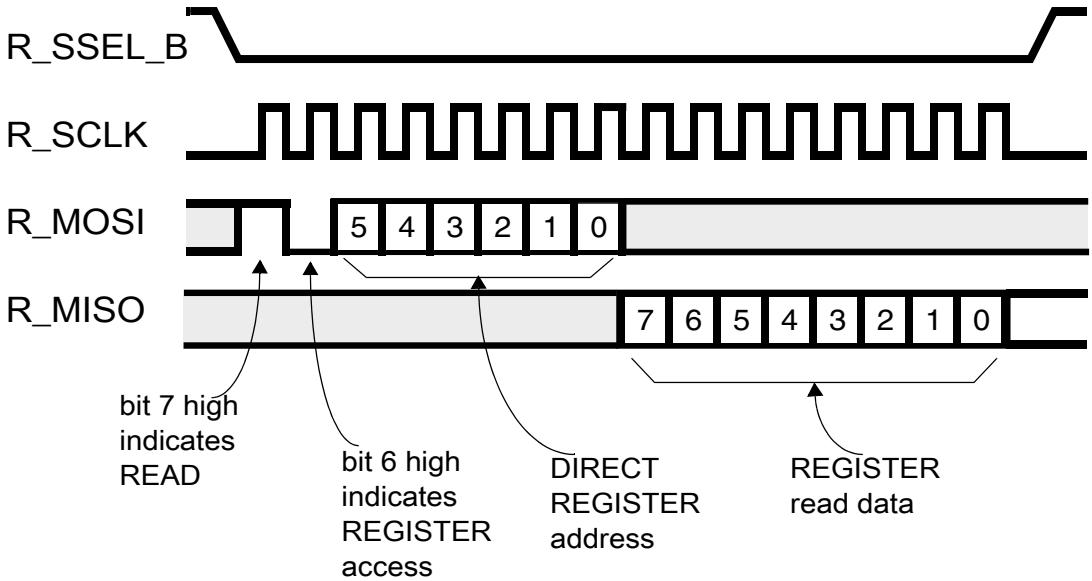


Figure 8-5. Single-Byte read Access to a Direct Register

8.3.4 Direct Register Write Access (multi byte)

The following diagram depicts a multi-byte write access to Direct Register space in the modem.

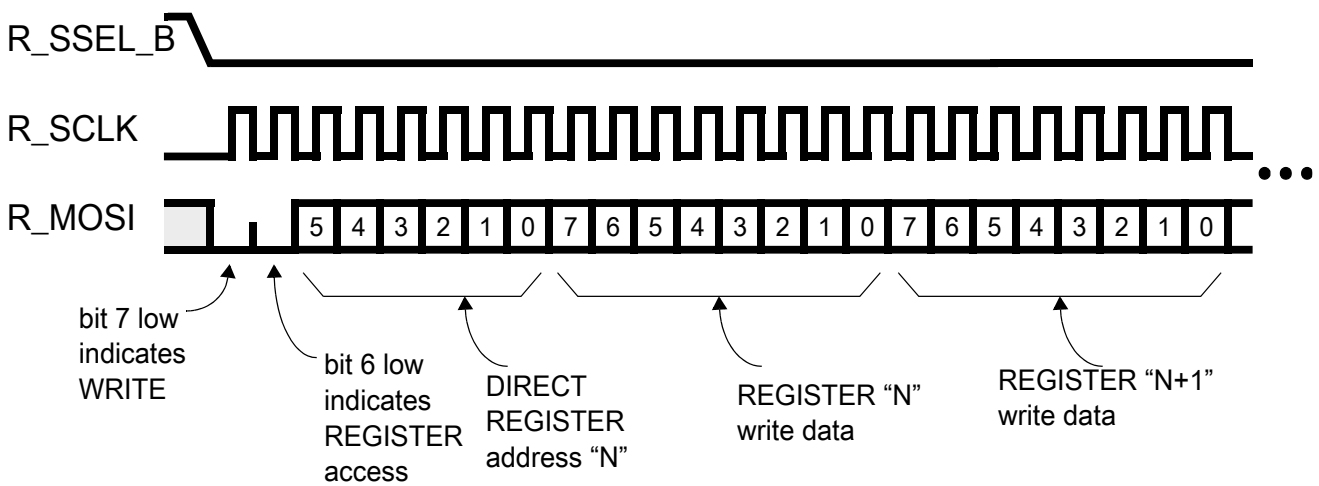


Figure 8-6. Multi-Byte Write Access to a Direct Register Space

8.3.5 Direct Register Read Access (multi byte)

The following diagram depicts a multi-byte read access to Direct Register space in the modem.

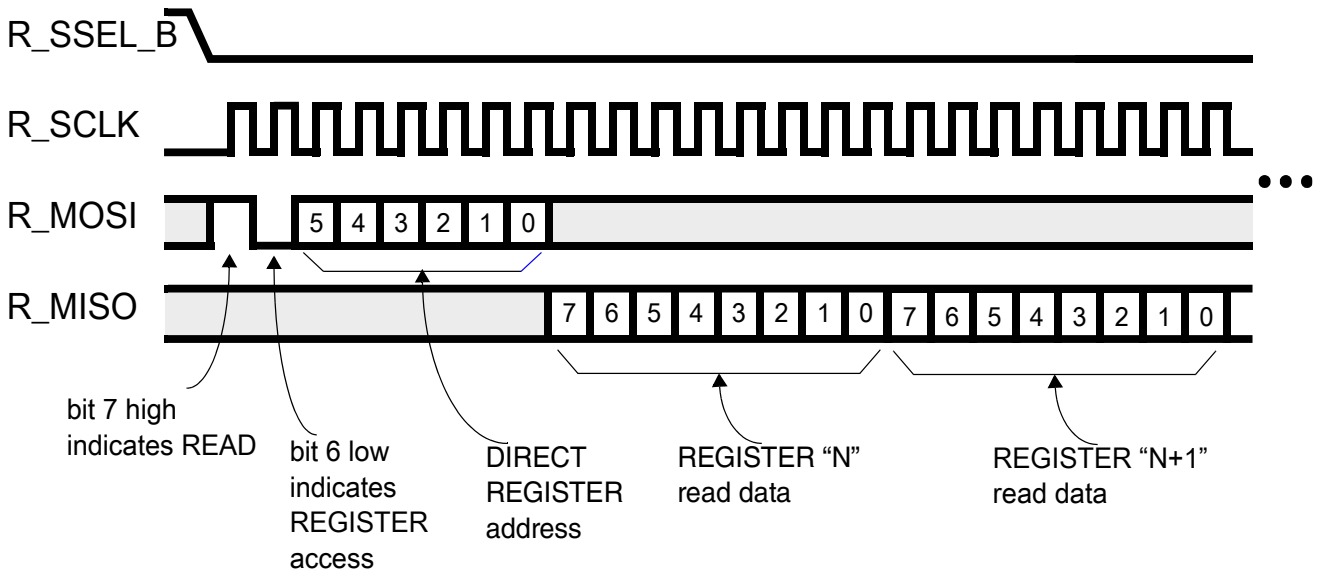


Figure 8-7. Multi-Byte Read Access to a Direct Register Space

8.3.6 Indirect Register Write Access (multi byte)

The following diagram depicts a multi-byte write access to Indirect Register space in the modem.

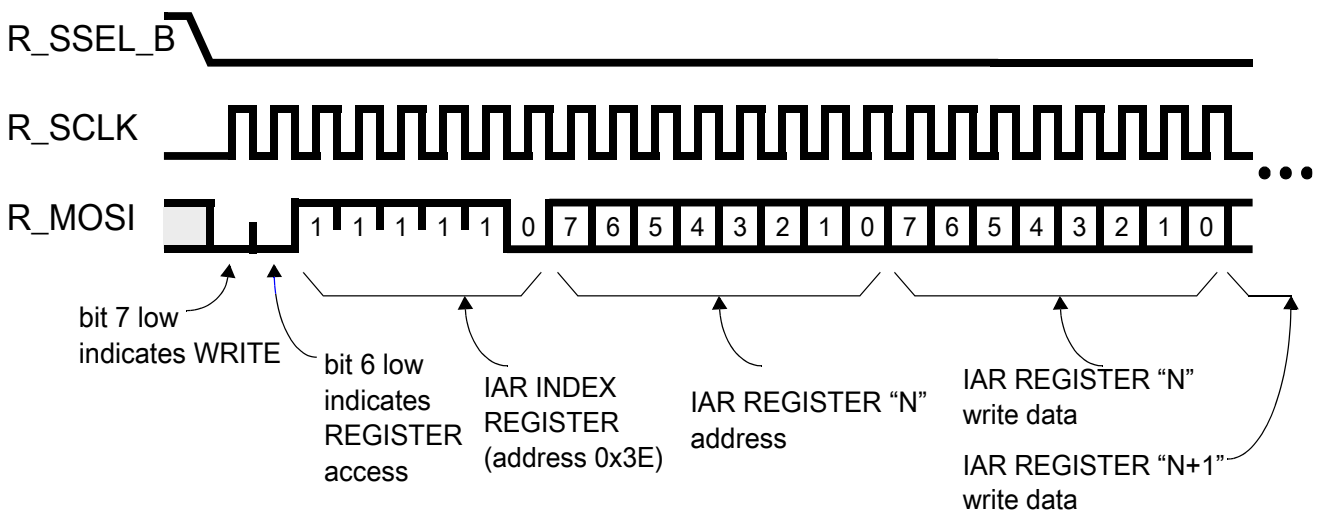


Figure 8-8. Multi-Byte Write Access to an Indirect Register Space

NOTE

Multi-byte SPI transactions to an Indirect Address space are not permitted in Hibernate state. Use single-byte transactions only.

8.3.7 Indirect Register Read Access (multi byte)

The following diagram depicts a multi-byte read access to Indirect Register space in the modem.

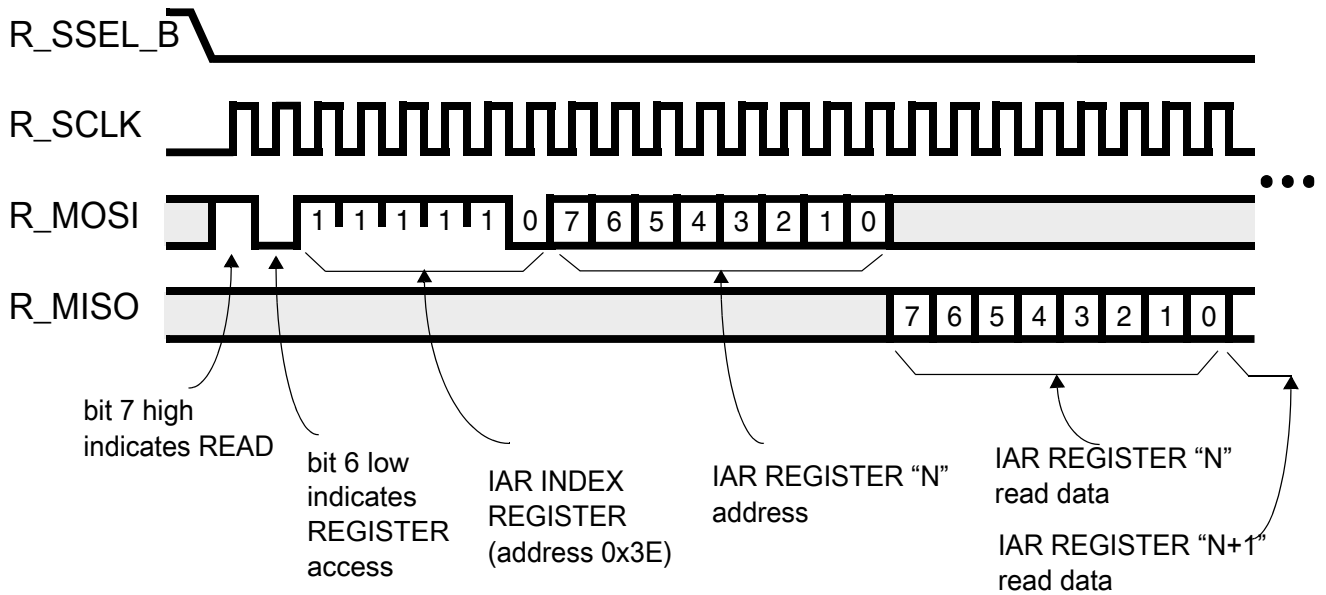


Figure 8-9. Multi-Byte Read Access to an Indirect Register Space

NOTE

Multi-byte SPI transactions to Indirect Address space are not permitted in Hibernate state. Use single-byte transactions only.

8.3.8 Synchronous and Asynchronous Operating Modes

The SPI module is designed to be fully operational in all modem power states, except reset. Operation in the Hibernate state means the modem's crystal oscillator is disabled. Most modem registers, direct and indirect, can be accessed, both read and write, during HibernateIn all non-Hibernate states, the SPI operates in "synchronous mode"; to write a register, the SPI module generates a clock enable to the Register block, where the enable is used to gate the 32MHz oscillator to each individual register; for reads, the readback data is registered using the 32MHz clock at the end of the SPI address phase, so that readback data cannot change during shift-out on R_MISO. In Hibernate, the SPI operates in "asynchronous mode"; to write a register, the SPI module generates both clock and clock enable to the Register block (because the crystal oscillator is disabled); for reads, no latching of readback data takes place, since the oscillator is disabled, readback data cannot change.

The SPI module automatically determines its operating state, based on the xtal_ready signal from the crystal oscillator. This signal is captured by the SPI module during the SPI address phase, during the first R_SCLK period. If it is high, the SPI operates in synchronous mode (as described above) for the duration of the SPI transaction. If xtal_ready is captured low, the SPI operates in asynchronous mode (as described above) for the duration of the SPI transaction. The transaction ends when R_SSEL_B is deasserted.

Transitions on xtal_ready are triggered by a SPI write access to the PWR_MODES register. A SPI write to clear the XTALEN bit of the PWR_MODES register will cause an immediate high-to-low transition of xtal_ready; this transition occurs at the end of the SPI transfer, after the last data bit is shifted in on R_MOSI. No further registers can be written during this transfer, even if the transaction is multi-byte, because the PWR_MODES registers is at the end of Direct Address space. A SPI write to set the XTALEN bit of PWR_MODES register will cause a delayed low-to-high transition on xtal_ready, because there is a finite crystal warm-up time. The SPI module will continue to operate in asynchronous mode, until a new SPI transaction starts with xtal_ready asserted.

8.3.9 Shifting Out IRQSTS1 During Control Word

During each SPI transaction, while the control word is being shifted in on R_MOSI (SPI address phase), the SPI module always shifts out the contents of Direct Register 0 "IRQSTS1" on R_MISO. The IRQSTS1 register is the highest-priority register in the modem; it contains the transceiver interrupt status bits, and must be accessed with low latency on any 802.15.4 interrupt. Because multiple registers must be accessed on transceiver interrupts, Direct Register space has been designed to put the highest-priority registers at the beginning of address space. A single, multi-byte transfer can be performed to access all of these registers. For such interrupt-driven SPI transfers, software can take advantage of the automatic shifting of IRQSTS1 during the SPI address phase, by programming a multi-byte SPI transfer to begin at Address 1 (not Address 0), because Address 0 data is provided "for free" on R_MISO during the control word phase. Thus, a 7-byte SPI transfer (to read Direct Registers 0 through 6) would require only 7 x 8 = 56 SPI clock periods, instead of 64 clock periods, which would be the case if Register 0 had to be addressed explicitly during such a transaction. The following diagram depicts the latency advantage that can be gained using this approach.

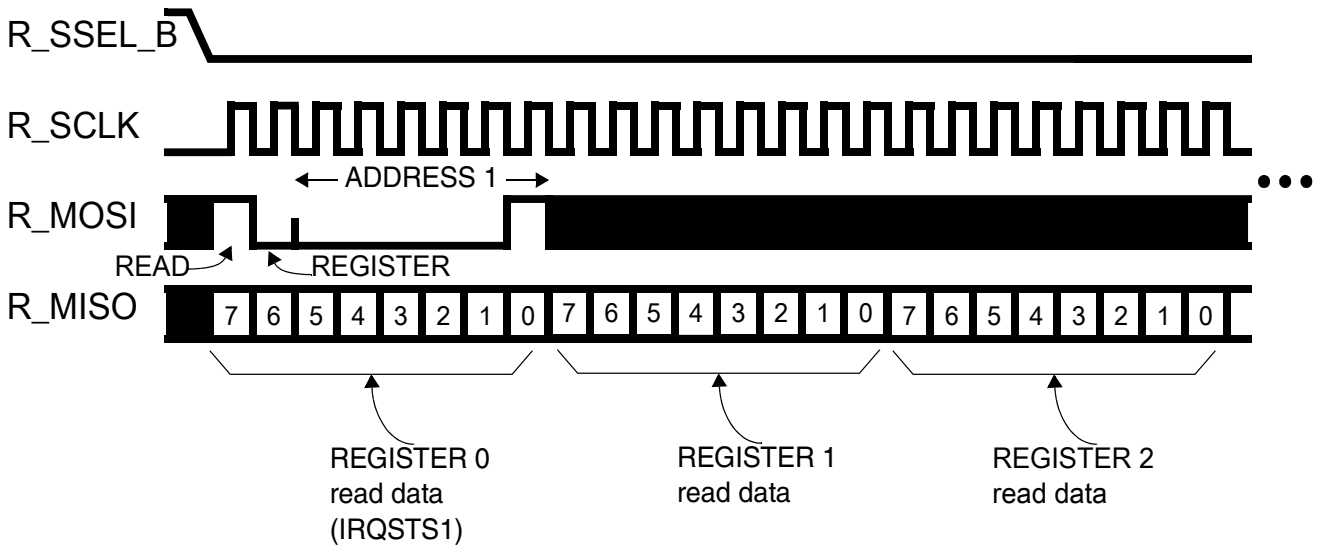


Figure 8-10. IRQSTS1 on R_MISO During Control Word

8.3.10 Packet Buffer

The Packet Buffer is a 128-byte Random Access Memory (RAM) dedicated to the storage of 802.15.4 packet contents for both TX and RX sequences. For TX sequences, software stores the contents of the packet buffer starting with the Frame Length byte at Packet Buffer Address 0, followed by the packet contents at the subsequent Packet Buffer Addresses. For RX sequences the incoming packet's Frame Length is stored in a register, external to the Packet Buffer.Software will read this register to determine the number of bytes of Packet Buffer to read. This facilitates DMA transfer through the SPI. For receive packets, an LQI byte is stored at the byte immediately following the last byte of the packet (Frame Length +1).

The Packet Buffer is not a FIFO. It is a byte-addressable memory capable of burst-mode transfers. Reading from the Packet Buffer does not destroy its contents. As long as no subsequent received packets are received, the Packet Buffer may be downloaded as many times as desired and the contents will not change.

8.3.10.1 Packet Buffer Architecture

The 802.15.4 standard specifies a maximum packet size of 127 bytes. The Packet Buffer is sized to accommodate a maximum-length packet, plus one additional byte. For TX, this additional byte is the “Frame Length” byte, or PHR, as it is known in 802.15.4

nomenclature. Software loads the Frame Length byte into address 0 of the Packet Buffer (the first data phase of a multi-byte SPI burst). The packet contents (PSDU) follow. The following diagram depicts the 802.15.4 packet structure, detailing the PHR and PSDU.

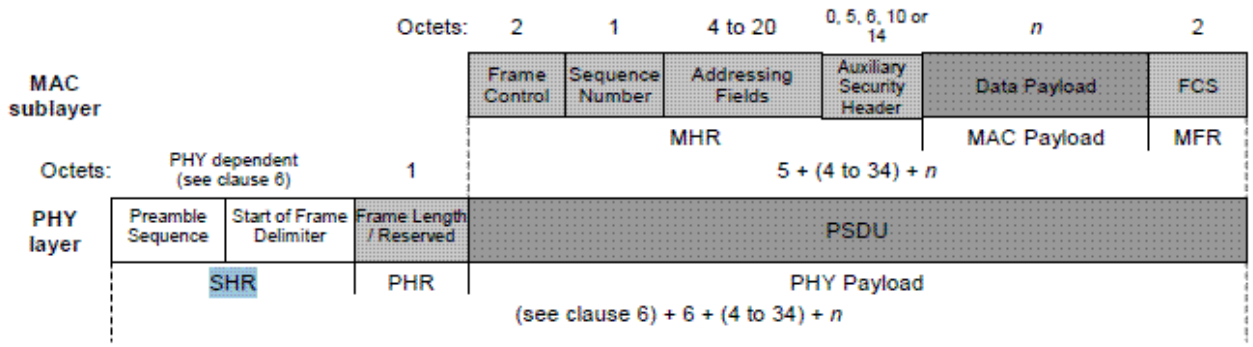


Figure 8-11. 802.15.4 standard packet architecture

The last 2 bytes of PSDU are CRC bytes for the 802.15.4 FCS (Frame Check Sequence). The radio hardware computes and transmits these 2 bytes automatically, so software need not load them into the Packet Buffer. Thus, for TX sequences, the number of bytes to load into the buffer, after Frame Length, is “Frame Length - 2”. For RX, the received Frame Length is routed to a SPI-readable register, RX_FRAME_LENGTH, not the Packet Buffer. The remainder of the packet’s PSDU, is stored in the Packet Buffer, starting at address 0. At Data Indication, software should read the RX_FRAME_LENGTH register to determine the number of bytes of Packet Buffer it needs to read, in order to download the entire packet via SPI transfer. A DMA transfer can be set up to perform the transfer, if the MCU is so equipped. The PSDU includes the 2 received FCS bytes. For RX packets, after the last packet FCS byte, an LQI byte is stored. Therefore, to download the entire packet plus LQI from the packet buffer, the number of bytes (data phases) in the SPI burst will be RX_FRAME_LENGTH+1, where RX_FRAME_LENGTH is the contents of the register of the same name. The following diagram depicts the contents of a TX buffer containing a maximum-length packet (just prior to transmission), and an RX buffer containing a maximum length packet (just after a data indication).

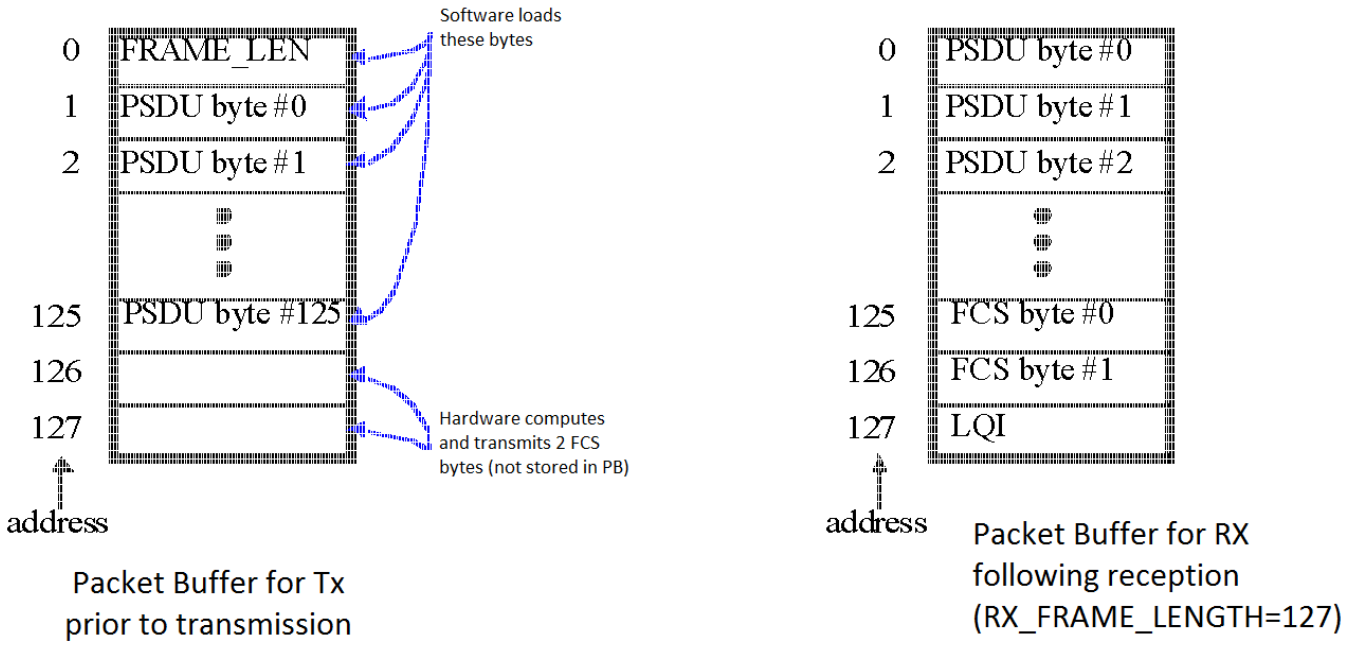


Figure 8-12. Packet buffer architecture

8.3.10.2 Packet Buffer SPI Transactions

8.3.10.2.1 Packet Buffer Access Modes: Burst Mode vs Byte Mode

Burst mode is recommended for Packet Buffer uploading (TX) and downloading (RX). For Packet Buffer Burst mode, no Packet Buffer address need be provided by the SPI. SPI hardware will always start the burst at Packet Buffer address 0. A single SPI transfer of up to 128 bytes can be used for either uploading or downloading in burst mode. This, in combination with DMA, reduces SPI latency to an absolute minimum.

In case the user desires to access an individual byte within the Packet Buffer, and not the entire buffer, a "Byte" access mode has also been provided. In this mode, the SPI transfer's second byte indicates the starting Packet Buffer address for the transfer. The subsequent bytes write (or read) Packet Buffer contents, starting at the indicated address. Byte mode is less efficient than Burst mode, because an address byte must follow the SPI control word.

In both Byte and Burst modes, there is no hardware limit to the number of bytes that can be uploaded or downloaded through the SPI. If a SPI transfer of greater than 128 bytes is attempted, the Packet Buffer address will wrap around to 0 after Packet Buffer address 127 has been accessed.

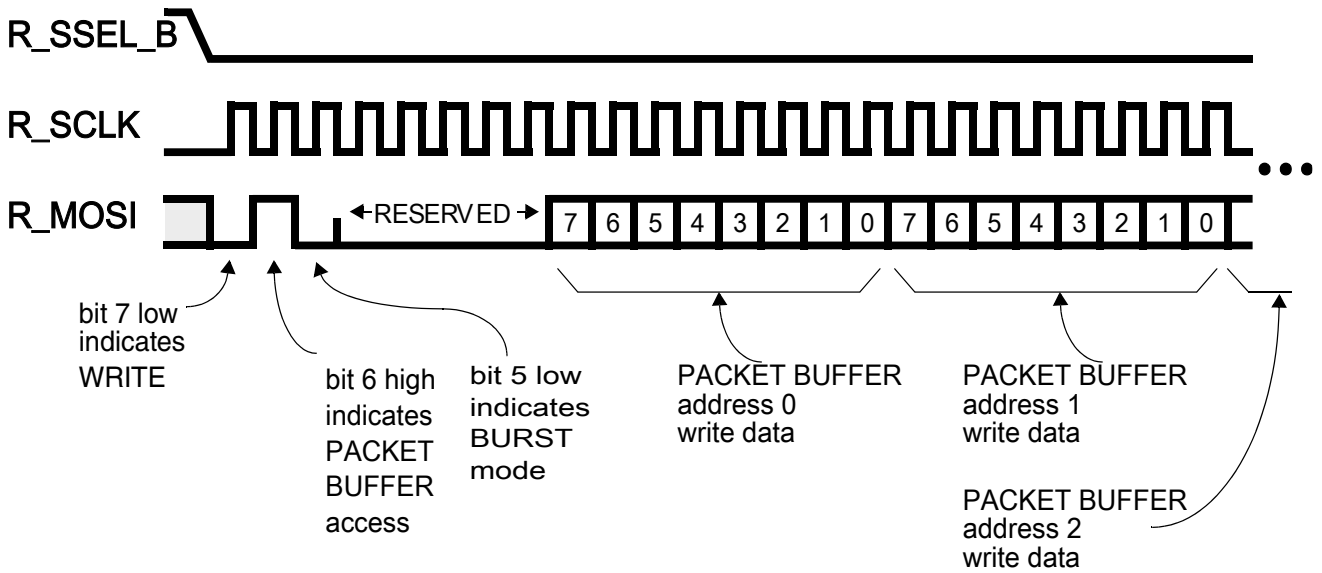


Figure 8-13. Packet Buffer Write (Burst Mode)

8.3.10.2.2 Packet Buffer Write Access (Burst Mode)

The following diagram depicts a SPI burst-mode write access to the Packet Buffer.

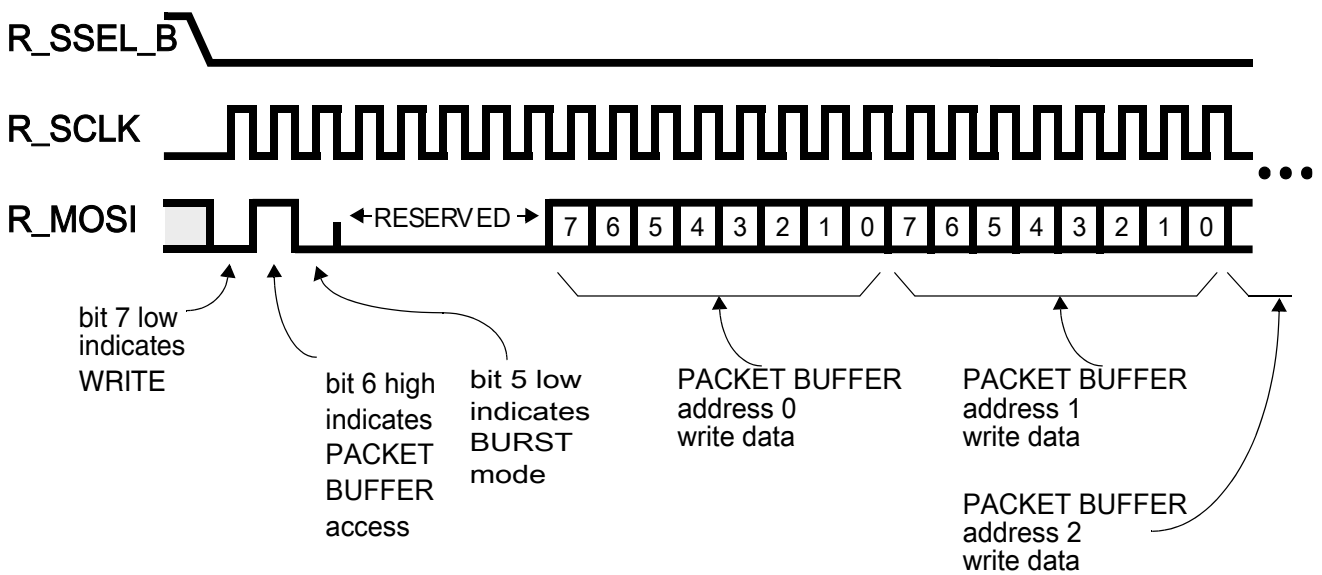


Figure 8-14. Packet Buffer Write (Burst Mode)

8.3.10.2.3 Packet Buffer Read Access (Burst Mode)

The following diagram depicts a SPI burst-mode read access to the Packet Buffer.

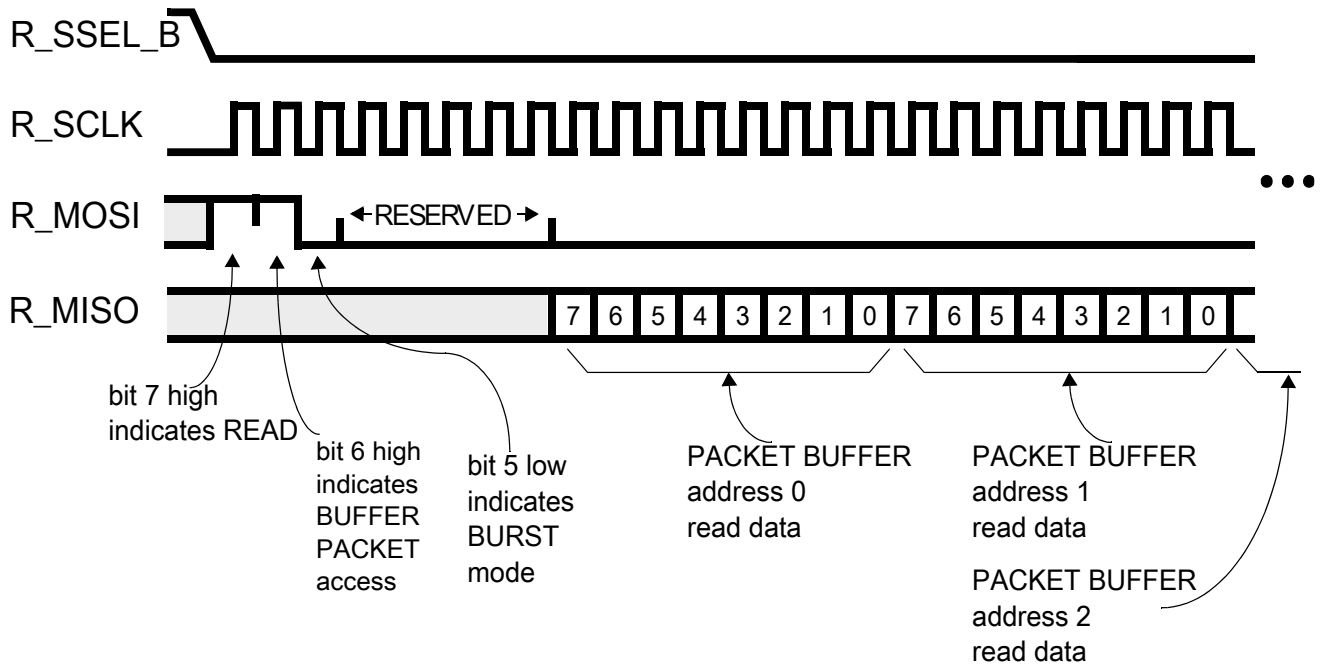


Figure 8-15. Packet Buffer Read (Burst Mode)

8.3.10.2.4 Packet Buffer Write Access (Byte Mode)

The following diagram depicts a SPI byte-mode write access to the Packet Buffer.

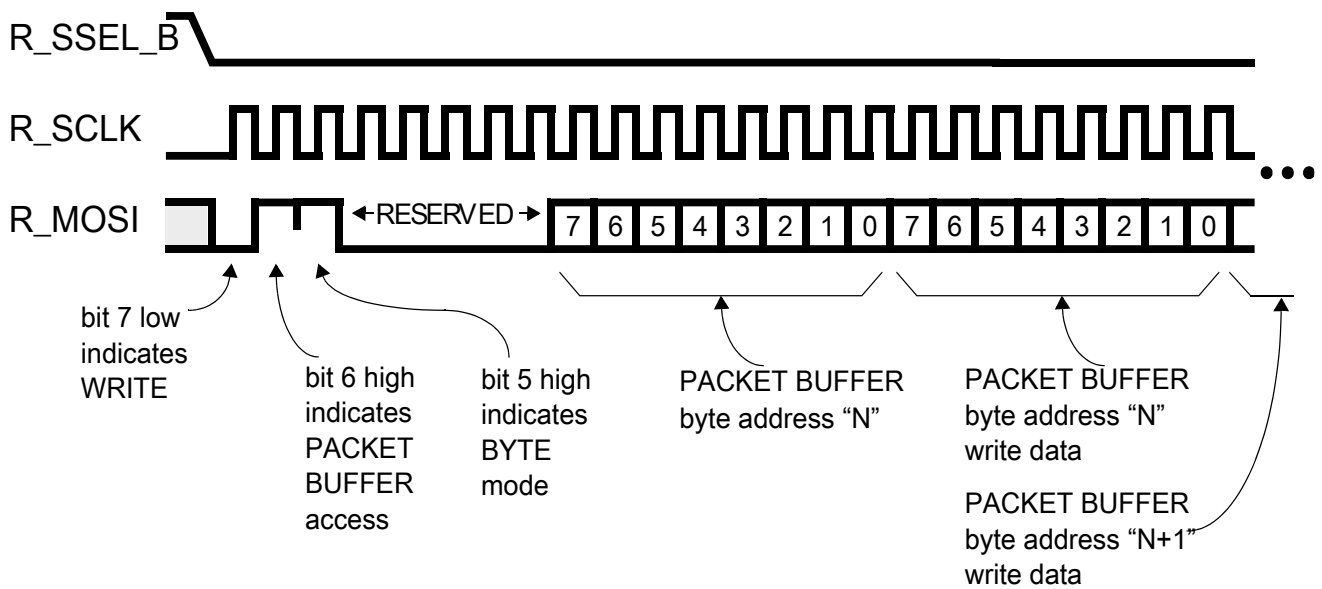


Figure 8-16. Packet Buffer Write (Byte Mode)

8.3.10.2.5 Packet Buffer Read Access (Byte Mode)

The following diagram depicts a SPI byte-mode read access to the Packet Buffer.

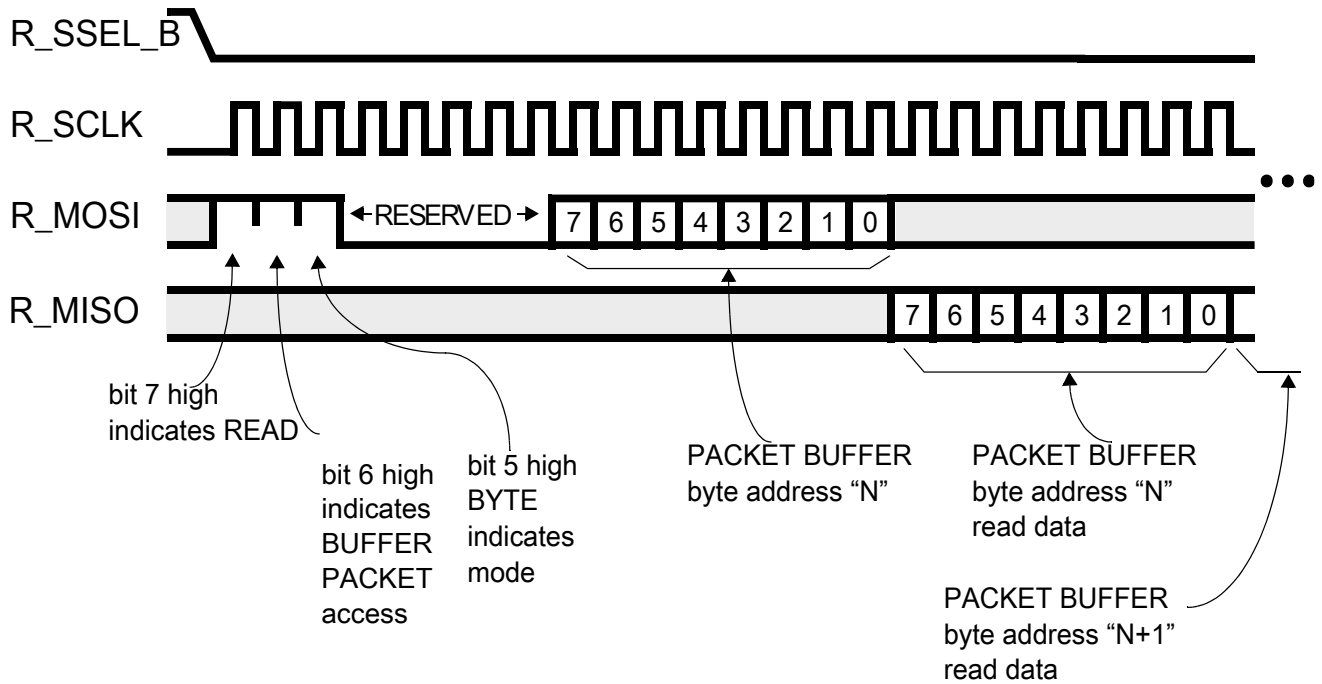


Figure 8-17. Packet Buffer Read (Byte Mode)

8.3.10.3 Handling of Auto-RXACK Packets

According to 802.15.4 protocol, when a packet is transmitted by a device, an acknowledge packet is often expected from the intended recipient. The radio features a hardware autosequence to transmit a packet, and wait for an acknowledge packet. This autosequence is initiated by programming a “Sequence TR” with RXACKRQD=1. During this autosequence, the received acknowledge packet is not stored in the Packet Buffer. The Packet Processor (hardware) merely checks the incoming packet for correct Sequence Number and FCS, and notifies software when the correct Acknowledge packet is received.

However, in Active Promiscuous mode, all received packets must be made available to software. Therefore in Active Promiscuous mode, auto-RXACK packets will be stored in the Packet Buffer.

8.3.10.4 Downloading Packet Buffer Contents During Reception

For packet reception, it is recommended that software wait for Data Indication prior to downloading Packet Buffer contents. This approach takes full advantage of the radio's Packet Processor hardware to check if the incoming packet was intended for the device, and for the FCS-check logic to verify CRC. However, hardware does not prohibit reading of Packet Buffer content while a packet is being received. Early downloading can mean earlier software access to packet contents for software processing, which can be advantageous during periods of high software workload or short turnaround requirements. To download the Packet Buffer during reception, the following steps should be followed:

1. Set the RX_WTR_MARK register to trigger a RXWTRMRKIRQ interrupt after the Nth byte of the packet has been received. The RX_WTR_MARK threshold corresponds directly to Packet Buffer address, plus 1. So, if an RXWTRMRKIRQ is desired after Packet Buffer address 7 has been loaded with receive data, set RX_WTR_MARK=8.
2. Unmask the RXWTRMRKIRQ by programming RX_WMRK_MSK=0 in the PHY_CTRL2 register.
3. Unmask the PB_ERR_IRQ by programming PB_ERR_MSK=0 in the PHY_CTRL3 register.
4. At the RXWTRMRKIRQ interrupt, read the RX_FRAME_LENGTH register to determine the ultimate length of the packet. At this point, the entire Packet Buffer has not been loaded with receive data, but the first "N" bytes (at least) are available, where N = RX_WTR_MARK-1.
5. Begin the SPI burst transfer of length "RX_FRAME_LENGTH", (or "RX_FRAME_LENGTH+1" to also get LQI) to download the Packet Buffer contents.

Caution must be used when using early downloading:

1. At Data Indication (which will occur some time after RXWTRMRKIRQ), software must check the validity of the received packet. At SEQIRQ, if RXIRQ is also asserted (IRQSTS1 register), then the packet passed filtering and FCS. Otherwise, it failed either filtering, CRC, or both.
2. It is responsibility of the user to select a combination of SPI Transfer Rate, and RX_WTR_MARK threshold, to ensure that the SPI does not attempt to read from Packet Buffer addresses that have not yet been loaded with receive data (i.e., an underrun condition). The SPI transfer rate is determined by the SCLK frequency of the SPI Master. Receive data is loaded into the Packet Buffer at a constant rate of 1 byte every 32 µs, so in general the SPI transfer rate will be considerably faster than the buffer loading rate. Software must calculate the number of bytes to be

downloaded (based on RX_FRAME_LENGTH), and then set the RX_WTR_MARK threshold based on the ratio of SPI transfer rate to buffer loading rate.

If the steps above have not been followed correctly, and a Packet Buffer underrun occurs, this means the SPI read address of the Packet Buffer has overtaken the RX demodulator write address. A PB_ERR_IRQ interrupt will be raised to indicate this condition. A PB_ERR_IRQ is not catastrophic. The following steps should be taken to recover from a PB_ERR_IRQ:

1. Abort the SPI transfer in progress, and discard the data received so far, because the SPI transfer has been corrupted.
2. Clear the PB_ERR_IRQ by writing a '1' to the PB_ERR_IRQ bit in the IRQSTS2 register.
3. Restart the SPI transfer to download the Packet Buffer again. The design of the Packet Buffer ensures that read data has not been lost.

8.3.10.5 Registers Related to Packet Buffer Operation

PB_ERR_IRQ - Indicates that a Packet Buffer Underrun Error has occurred. This condition can occur if a SPI burst read access to the Packet Buffer begins before the incoming packet is completely received, and the SPI PB read address overtakes the RX demodulator write address. (See Downloading Packet Buffer Contents During Reception for information on how to avoid this condition, and what to do if it does occur). Software must write a '1' to this bit to clear the interrupt. The PB_ERR_IRQ interrupt can occur only during SPI BURST accesses to Packet Buffer; SPI BYTE mode accesses to Packet Buffer will not trigger a PB_ERR_IRQ even if the SPI address is greater than the RX Demodulator address.

1: Packet Buffer Underrun Error has occurred since the last time this bit was cleared by software

0: Packet Buffer Underrun Error has not occurred since the last time this bit was cleared by software

ACTIVE_PROMISCUOUS - Normally, received packets that are of Frame Type "Acknowledge", received in response to a packet transmitted by the radio, are not stored in the Packet Buffer. Such packets are expected during a Sequence TR with RXACKRQD=1. In Active Promiscuous mode, software must have access to all receive packet data, even "auto-RXACK" packets. This bit allows such packets to be stored in the Packet Buffer.

1: Allow "auto-RXACK" packets to be stored in the Packet Buffer

0: Don't allow "auto-RXACK" packets to be stored in the Packet Buffer (default)

8.4 Configuring MCU for Proper SPI Operation

The MCU port control must be configured as follows to operate with the modem SPI:

- PTB10 should be configured as SPI1_PCS0
- PTB11 should be configured as SPI1_SCK
- PTB16 should be configured as SPI1_SOUT
- PTB17 should be configured as SPI1_SIN. Also, the default configuration of the modem R_MISO is that the modem tri-states this output when the R_SSEL_B is not asserted. The PTB17 pad should be configured with a pull-up or pull-down enabled.

The MCU DSPI module must also be configured for proper operation to meet the modem SPI transaction format. The following conditions must be met:

- MCU is master
- Maximum baud rate is 9 MHz for reads, and 16 MHz for writes
- Proper clock format must be selected, i.e., CPHA=0 and CPOL=0
- SPI data must be transferred MSB first
- The following modem SPI timings parameters must not be violated:
  - t_{CSC} (R_SSEL_B assertion to SCK delay) must be no less than 31.25ns
  - t_{ASC} (SCK to R_SSEL_B deassertion delay) must be no less than 31.25ns
  - t_{DT} (R_SSEL_B deasserted idle time) must be no less than 62.5ns

The following sections describe the DSPI configuration in more detail. Refer to the MCU: Serial Peripheral Interface (SPI) chapter.

8.4.1 DSPI Mode Configuration

The following DSPI1 register settings are required to communicate correctly with the modem SPI:

- MSTR (SPI1_MCR, bit 31). MSTR =1 to configure the DSPI for master mode
- PCSIS[0] (SPI1_MCR, bit 16). PCSIS[0]=1 must be set to indicate that the inactive state of PCS0 is high
- CPOL (SPI1_CTARn, bit 26) - CPOL=0 to select the correct clock polarity (active high / inactive low)
- CPHA (SPI1_CTARn, bit 25) - CPHA=0 to select the correct clock phase (data captured on rising edge, changed on falling edge)
- LSBFE (SPI1_CTARn, bit 24) - LSBFE=0 to transfer data MSB first

8.4.2 DSPI Baud Rate

The DSPI baud rate needs to be configured so that it is no more than 9MHz for modem SPI reads and 16MHz for modem SPI writes. The DSPI baud rate is determined by the MCU bus clock frequency, as well as by several programmable register bits in the DSPI.

- PBR (SPI1_CTARn, bits 17:16) - selects the baud rate prescaler value
- BR (SPI1_CTARn, bits 3:0) - select the baud rate scaler
- DBR (SPI1_CTARn, bit 31) - selects a double baud rate mode

As an example, assuming a 50MHz bus clock, PBR could be programmed for a divide-by-3 value, BR could be programmed for a divide-by-2 value, and DBR could be programmed to 0 to achieve an 8.3MHz SPI clock rate. For SPI writes, the PBR could be changed to a divide-by-2 value to realize a 12.5MHz SPI clock rate.

If a 32MHz bus clock is used, PBR could be programmed for a divide-by-2 value, BR could be programmed for a divide-by-2 value, and DBR could be programmed to 0 to achieve an 8MHz SPI clock rate. For SPI writes, the DBR could be changed to 1 to realize a 16MHz SPI clock rate.

Refer to the MCU: Serial Peripheral Interface (SPI) chapter for more information on baud rate.

8.4.3 DSPI Timing Control

The DSPI provides the following programmable control relating to the SPI timing parameters t_{CSC}, t_{ASC}, and t_{DT}:

- PCSSCK (SPI1_CTARn, bits 23:22) and CSSCK (SPI1_CTARn, bits 15:12) - these control the delay from PCS assertion to the first SCK edge, that is, t_{CSC}
- PASC (SPI1_CTARn, bits 21:20) and ASC (SPI1_CTARn, bits 11:8) - these control the delay from the last SCK edge to the negation of PCS, that is, t_{ASC}
- PDT (SPI1_CTARn, bits 19:18) and DT (SPI1_CTARn, bits 7:4) - these control the delay from the negation of PCS to the assertion of PCS for the next SPI frame, that is, t_{DT}

Since the maximum bus clock in the MCU is 50MHz, t_{CSC} can be no more than 40ns regardless of programming. So PCSSCK and CSSCK can be programmed to their default values (device-by-1 and divide-by-2 respectively) to minimize this delay and still meet the requirement that t_{CSC} be no less than 32.5ns.

Likewise, t_{ASC} can be no more than 40ns regardless of programming since the MCU bus clock 50MHz limit. So PASC and ASC can also be programmed to their default values (device-by-1 and divide-by-2 respectively) to minimize this delay and still meet the requirement that t_{ASC} be no less than 32.5ns.

However, depending on the bus frequency, the PDT and DT may need to be programmed to ensure that t_{DT} is no less than 62.5ns. If the bus clock is 32MHz or less, the default programming for PDT and DT can be used, since then t_{DT}will be no more than 62.5ns. However, with a 50MHz bus clock, the default programming for PDT and DT would cause t_{DT} to be 40ns, which does not meet the modem requirement. Programming DT to a divide-by-4 value (instead of default divide-by-2) when the bus clock is 50MHz would increase t_{DT} to 80ns to meet the modem timing requirement.

Refer to the MCU: Serial Peripheral Interface (SPI) chapter for more information on baud rate and clock delay generation.

Chapter 9
Modem: SPI Register Descriptions

9.1 Introduction

This section provides a complete listing of all the modem registers, including register address, access modes, descriptions, and references to other documents for more complete information.

The transceiver registers allow the MCU to communicate and control the radio as well as receive feedback and status information. All radio registers are accessible via a SPI interface only. No other means of accessing the transceiver registers is possible. The registers are divided into two categories; direct and indirect. The first 64 registers are “Direct Access”, meaning only a single SPI control word is required to identify the register to be accessed. The remaining registers are “Indirect Access” and use a pointer register followed by a data register as discussed following.

Indirect access registers use a pointer register and a window register for access in the indirect space. The pointer register (IAR_INDEX) and the data register (IAR_DATA) occupy the last two registers slots in Direct Space; addresses 0x3E and 0x3F respectively. Both direct- and indirect-access registers can be accessed in SPI burst mode, in which the SPI control word addresses the first register in the burst. After the first register in the burst is accessed, the hardware will auto-increment the register address so that subsequent sequential registers can be accessed during a single multi-byte SPI transaction. The length of the burst is not limited by hardware.

Direct access burst mode is available in all power states, including HIBERNATE. Indirect access burst mode is restricted to states during which the crystal oscillator is running (e.g., DOZE, IDLE, RUN); this excludes HIBERNATE state. Indirect register access in HIBERNATE state is restricted to single-byte.

A few registers, which control special functions within the IC, should not be accessed in HIBERNATE state; these are identified in the “Register Description” tables later in this document. However, most registers are 100% accessible in all power states, including HIBERNATE.

In the following Register Summary table, registers are listed in ascending address order, starting with direct address space (direct addresses 0x00 - 0x3F), followed by indirect address space (addresses 0x00 - 0xFF). For each register, address (direct or indirect) appears in the left-hand column. The register bits are then listed horizontally from left to right, bit 7 ... bit 0. The right-hand column contains the register mnemonic. For each bit, an access mode is shown (r = read-only; rw = read/write; w1tc=write-1-to-clear; sc=self-clearing; etc.). And for all read/write registers, a default value is provided. Unpopulated registers, or unpopulated register bits within a partially-populated register, are indicated by empty cells in the table, and should be considered reserved; these bit positions read back 0, and only 0 should be written to them.

9.2 Modem Memory map and register definition

NOTE

The Direct and Indirect registers can be accessed only through the SPI interface. Direct access by a single SPI control word and Indirect using the pointer register (IAR_INDEX) and the data register (IAR_DATA), which occupy the last two registers in direct access; addresses 0x3E and 0x3F respectively.

Modem memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	Interrupt Request Status 1 (Modem_IRQSTS1)	8	w1c	00h	9.2.1/134
1	Interrupt Request Status 2 (Modem_IRQSTS2)	8	R	01h	9.2.2/136
2	Interrupt Request Status 3 (Modem_IRQSTS3)	8	R/W	F0h	9.2.3/137
3	PHY Control 1 (Modem_PHY_CTRL1)	8	R/W	00h	9.2.4/139
4	PHY Control 2 (Modem_PHY_CTRL2)	8	R/W	FFh	9.2.5/140
5	PHY Control 3 (Modem_PHY_CTRL3)	8	R/W	06h	9.2.6/141
6	Receive Frame Length (Modem_RX_FRM_LEN)	8	R	00h	9.2.7/142
7	PHY Control 4 (Modem_PHY_CTRL4)	8	R/W	08h	9.2.8/143
8	SRC Control (Modem_SRC_CTRL)	8	R/W	0Ch	9.2.9/144
9	SRC Address SUM LSB (Modem_SRC_ADDRS_SUM_LSB)	8	R/W	00h	9.2.10/145
A	SRC Address SUM MSB (Modem_SRC_ADDRS_SUM_MSB)	8	R/W	00h	9.2.11/145
B	CCA1 ED FNL (Modem_CCA1_ED_FNL)	8	R	00h	9.2.12/146
C	Event Timer LSB (Modem_EVENT_TIMER_LSB)	8	R	00h	9.2.13/146
D	Event Timer MSB (Modem_EVENT_TIMER_MSB)	8	R	00h	9.2.14/147
E	Event Timer USB (Modem_EVENT_TIMER_USB)	8	R	00h	9.2.15/147

Table continues on the next page...

Modem memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
F	Timestamp LSB (Modem_TIMESTAMP_LSB)	8	R	00h	9.2.16/147
10	Timestamp MSB (Modem_TIMESTAMP_MSB)	8	R	00h	9.2.17/148
11	Timestamp USB (Modem_TIMESTAMP_USB)	8	R	00h	9.2.18/148
12	Timer 3 Compare Value LSB (Modem_T3CMP_LSB)	8	R/W	FFh	9.2.19/149
13	Timer 3 Compare Value MSB (Modem_T3CMP_MSB)	8	R/W	FFh	9.2.20/149
14	Timer 3 Compare Value USB (Modem_T3CMP_USB)	8	R/W	FFh	9.2.21/149
15	Timer 2-Prime Compare Value LSB (Modem_T2PRIMECMP_LSB)	8	R/W	FFh	9.2.22/150
16	Timer 2-Prime Compare Value MSB (Modem_T2PRIMECMP_MSB)	8	R/W	FFh	9.2.23/150
17	Timer 1 Compare Value LSB (Modem_T1CMP_LSB)	8	R/W	FFh	9.2.24/150
18	Timer 1 Compare Value MSB (Modem_T1CMP_MSB)	8	R/W	FFh	9.2.25/151
19	Timer 1 Compare Value USB (Modem_T1CMP_USB)	8	R/W	FFh	9.2.26/151
1A	Timer 2 Compare Value LSB (Modem_T2CMP_LSB)	8	R/W	FFh	9.2.27/151
1B	Timer 2 Compare Value MSB (Modem_T2CMP_MSB)	8	R/W	FFh	9.2.28/152
1C	Timer 2 Compare Value USB (Modem_T2CMP_USB)	8	R/W	FFh	9.2.29/152
1D	Timer 4 Compare Value LSB (Modem_T4CMP_LSB)	8	R/W	FFh	9.2.30/152
1E	Timer 4 Compare Value MSB (Modem_T4CMP_MSB)	8	R/W	FFh	9.2.31/153
1F	Timer 4 Compare Value USB (Modem_T4CMP_USB)	8	R/W	FFh	9.2.32/153
20	PLL Integer Value for PAN0 (Modem_PLL_INT0)	8	R/W	0Ch	9.2.33/153
21	PLL Frequency Fractional Value for PAN0 (Modem_PLL_FRAC0_LSB)	8	R/W	00h	9.2.34/154
22	PLL Frequency Fractional Value for PAN0 (Modem_PLL_FRAC0_MSB)	8	R/W	90h	9.2.35/154
23	PA Power Control (Modem_PA_PWR) (Modem_PA_PWR)	8	R/W	18h	9.2.36/155
24	Sequence Manager State (Modem_SEQ_STATE)	8	R	00h	9.2.37/155
25	Link Quality Indicator (Modem_LQI_VALUE)	8	R	00h	9.2.38/155
26	RSSI CCA CNT (Modem_RSSI_CCA_CNT)	8	R	00h	9.2.39/156
28	ASM Control 1 (Modem_ASM_CTRL1)	8	R/W	00h	9.2.40/156
29	ASM Control 2 (Modem_ASM_CTRL2)	8	R/W	00h	9.2.41/157
2A	ASM Data (Modem_ASM_DATA_0)	8	R/W	00h	9.2.42/158
2B	ASM Data (Modem_ASM_DATA_1)	8	R/W	00h	9.2.42/158
2C	ASM Data (Modem_ASM_DATA_2)	8	R/W	00h	9.2.42/158
2D	ASM Data (Modem_ASM_DATA_3)	8	R/W	00h	9.2.42/158
2E	ASM Data (Modem_ASM_DATA_4)	8	R/W	00h	9.2.42/158
2F	ASM Data (Modem_ASM_DATA_5)	8	R/W	00h	9.2.42/158
30	ASM Data (Modem_ASM_DATA_6)	8	R/W	00h	9.2.42/158
31	ASM Data (Modem_ASM_DATA_7)	8	R/W	00h	9.2.42/158
32	ASM Data (Modem_ASM_DATA_8)	8	R/W	00h	9.2.42/158

Table continues on the next page...

Modem memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
33	ASM Data (Modem_ASM_DATA_9)	8	R/W	00h	9.2.42/158
34	ASM Data (Modem_ASM_DATA_A)	8	R/W	00h	9.2.42/158
35	ASM Data (Modem_ASM_DATA_B)	8	R/W	00h	9.2.42/158
36	ASM Data (Modem_ASM_DATA_C)	8	R/W	00h	9.2.42/158
37	ASM Data (Modem_ASM_DATA_D)	8	R/W	00h	9.2.42/158
38	ASM Data (Modem_ASM_DATA_E)	8	R/W	00h	9.2.42/158
39	ASM Data (Modem_ASM_DATA_F)	8	R/W	00h	9.2.42/158
3B	Overwrite Version Number (Modem_OVERWRITE_VER)	8	R/W	00h	9.2.43/158
3C	CLK_OUT Control (Modem_CLK_OUT_CTRL)	8	R/W	88h	9.2.44/159
3D	Power Modes (Modem_PWR_MODES)	8	R/W	11h	9.2.45/160
3E	IAR Index (Modem_IAR_INDEX)	8	W	00h	9.2.46/161
3F	IAR Data (Modem_IAR_DATA)	8	R/W	00h	9.2.47/162

9.2.1 Interrupt Request Status 1 (Modem_IRQSTS1)

This register provides IRQ status information.

Address: 0h base + 0h offset = 0h

Bit	7	6	5	4
Read	RX_FRM_PEND	PLL_UNLOCK_IRQ	FILTERFAIL_IRQ	RXWTRMRKIRQ
Write		w1c	w1c	w1c
Reset	0	0	0	0

Bit	3	2	1	0
Read	CCAIRQ	RXIRQ	TXIRQ	SEQIRQ
Write	w1c	w1c	w1c	w1c
Reset	0	0	0	0

Modem_IRQSTS1 field descriptions

Field	Description
7 RX_FRM_PEND	Frame Pending Interrupt Status Shows the status of the frame pending bit of the frame control field for the most-recently received packet. This is a read-only bit. 0 Indicates that an unlock event has not occurred in the PLL. 1 Indicates that an unlock event has not occurred in the PLL.
6 PLL_UNLOCK_ IRQ	PLL Unlock Interrupt Status Shows whether an unlock event has occurred in the PLL. This is write a 1 to clear bit.

Table continues on the next page...

Modem_IRQSTS1 field descriptions (continued)

Field	Description
	0 Indicates that an unlock event has not occurred in the PLL. 1 Indicates that an unlock event has not occurred in the PLL.
5 FILTERFAIL_IRQ	Receiver Packet Filter Fail Interrupt Status Shows whether the most-recently received packet has been rejected due to elements within the packet. This is write a 1 to clear bit. • If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=0, FILTERFAIL_IRQ applies to PAN0. • If PAN0 and PAN1 occupy different channels and CURRENT_NETWORK=1, FILTERFAIL_IRQ applies to PAN1. • If PAN0 and PAN1 occupy the same channel, FILTERFAIL_IRQ is the logical 'AND' of the individual PANs' Filter Fail status.
4 RXWTRMRKIRQ	Receiver Byte Count Water Mark Interrupt Status Shows whether the number of bytes specified in the RX_WTR_MARK register has been reached. This is write a 1 to clear bit. 0 Indicates that the number of bytes specified in the RX_WTR_MARK register has not been reached 1 Indicates that the number of bytes specified in the RX_WTR_MARK register has been reached
3 CCAIRQ	Clear Channel Assessment Interrupt Status Shows the status of a CCA operation. This is write a 1 to clear bit. See the sequence manager section for a description of the timing of this interrupt. 0 Indicates that the completion of a CCA operation has not occurred. 1 Indicates that the completion of a CCA operation has occurred.
2 RXIRQ	Receiver Interrupt Status Shows the status of a receive operation. This is write a 1 to clear bit. 0 Indicates that the completion of a receive operation has not occurred. 1 Indicates that the completion of a receive operation has occurred.
1 TXIRQ	Transmitter Interrupt Status Shows the status of a transmit operation. This is write a 1 to clear bit. 0 Indicates that the completion of a transmit operation has not occurred. 1 Indicates that the completion of a transmit operation has occurred.
0 SEQIRQ	Sequence-end Interrupt Request Status This interrupt will assert whenever the Sequence Manager transitions from non-idle to idle state, for any reason. This is write a 1 to clear bit. 0 Indicates that the completion of an autosequence has not occurred. 1 Indicates that the completion of an autosequence has occurred.

9.2.2 Interrupt Request Status 2 (Modem_IRQSTS2)

This register provides IRQ status information.

Address: 0h base + 1h offset = 1h

Bit	7	6	5	4
Read	CRCVALID	CCA	SRCADDR	PI
Write				
Reset	0	0	0	0

Bit	3	2	1	0
Read	TMRSTATUS/WAKE_FROM_HIB	ASM_IRQ	PB_ERR_IRQ	WAKE_IRQ
Write		w1c	w1c	w1c
Reset	0	0	0	1

Modem_IRQSTS2 field descriptions

Field	Description
7 CRCVALID	Code Redundancy Check Valid Indicates the compare result between the FCS field, in the most-recently received frame, and the internally calculated CRC value. This flag is cleared at the next receiver warmup. 0 Rx FCS != calculated CRC (incorrect) 1 Rx FCS = calculated CRC (correct)
6 CCA	Channel Idle/Busy This indicator is valid at CCAIRQ and also at SEQIRQ. This flag is cleared at next receiver warmup. 0 Channel is idle. 1 Channel is busy.
5 SRCADDR	Source Address Match Shows whether all three of these conditions are true: The incoming packet was a data request. Source address matching is enabled, SCRADDR_EN=1. Source address checksum computed by packet processor matched at least one entry in the source address table. 0 At least one of these conditions is not true. 1 All three of these conditions are true.
4 PI	Poll Indication Indicates whether the received packet was a data request.

Table continues on the next page...

Modem_IRQSTS2 field descriptions (continued)

Field	Description
	0 Indicates that the received packet was not a data request. 1 Indicates that the received packet was a data request, regardless of whether a source address table match occurred, or whether source address matching is enabled. (See SRCADDR_EN in direct register 0x08, MODEM_SRC_CTRL.)
3 TMRSTATUS/WAKE_FROM_HIB	Timer Status/Wake from Hibernate This is a dual-purpose, read-only status bit. Its value is dependent on the WAKE_IRQ bit. Used to indicate either wakeup status or timer status. 0 If WAKE_IRQ=1, a 0 indicates that a wakeup from POR has occurred. If WAKE_IRQ=0, a 0 indicates that no TMRxIRQ is asserted. See the IRQST3 register for the individual timer interrupt status bits. 1 If WAKE_IRQ=1, a 1 indicates that a wakeup from Hibernate has occurred. If WAKE_IRQ=0, a 1 indicates that at least one of the TMRxIRQ is asserted. See the IRQST3 register for the individual timer interrupt status bits.
2 ASM_IRQ	AES Interrupt Flag A 1 indicates the completion of a AES operation completed. This is write a 1 to clear bit.
1 PB_ERR_IRQ	Packet Buffer Underrun Error Interrupt Status Shows whether a packet buffer underrun error has occurred since the last time this bit was cleared by software. This is a write 1 to clear bit. PB_ERR_IRQ will occur only when the SPI packet buffer access is Burst mode (not Byte mode). 0 A packet buffer underrun error has not occurred since the last time this bit was cleared by software. 1 A packet buffer underrun error has occurred since the last time this bit was cleared by software.
0 WAKE_IRQ	Wake Interrupt Status Indicates that either a wake-from-POR or a wake-from-Hibernate event has occured. This is write a 1 to clear bit. 0 Indicates that neither wake event has occurred. 1 Indicates that one of these wake events has occurred. See the WAKE_FROM_HIB/TMRSTATUS bit to see which type of wake has occurred.

9.2.3 Interrupt Request Status 3 (Modem_IRQSTS3)

This register provides IRQ status information.

Address: 0h base + 2h offset = 2h

Bit	7	6	5	4	3	2	1	0
Read	TMR4MSK	TMR3MSK	TMR2MSK	TMR1MSK	TMR4IRQ	TMR3IRQ	TMR2IRQ	TMR1IRQ
Write					w1c	w1c	w1c	w1c
Reset	1	1	1	1	0	0	0	0

Modem_IRQSTS3 field descriptions

Field	Description
7 TMR4MSK	Timer Comparator 4 Interrupt Mask 0 Allows interrupt when comparator matches event timer count. 1 Disables interrupt generation, but allows a TMR4IRQ flag to be set.
6 TMR3MSK	Timer Comparator 3 Interrupt Mask 0 Allows interrupt when comparator matches event timer count. 1 Disables interrupt generation, but allows a TMR3IRQ flag to be set.
5 TMR2MSK	Timer Comparator 2 Interrupt Mask 0 Allows interrupt when comparator matches event timer count. 1 Disables interrupt generation, but allows a TMR2IRQ flag to be set.
4 TMR1MSK	Timer Comparator 1 Interrupt Mask 0 Allows interrupt when comparator matches event timer count. 1 Disables interrupt generation, but allows a TMR1IRQ flag to be set.
3 TMR4IRQ	Timer Comparator 4 Interrupt Status Indicates whether the T4CMP comparator value matched the event timer counter. This is a write 1 to clear bit. 0 Indicates that the T4CMP comparator value does not match the event timer counter. 1 Indicates that the T4CMP comparator value does match the event timer counter.
2 TMR3IRQ	Timer Comparator 3 Interrupt Status Indicates whether the T1CMP comparator value matched the event timer counter. This is a write 1 to clear bit. 0 Indicates that the T3CMP comparator value does not match the event timer counter. 1 Indicates that the T3CMP comparator value does match the event timer counter.
1 TMR2IRQ	Timer Comparator 2 Interrupt Status Indicates whether the T2CMP comparator value matched the event timer counter. This is a write 1 to clear bit. 0 Indicates that the T2CMP comparator value does not match the event timer counter. 1 Indicates that the T2CMP comparator value does match the event timer counter.
0 TMR1IRQ	Timer Comparator 1 Interrupt Status Indicates whether the T1CMP comparator value matched the event timer counter. This is a write 1 to clear bit. 0 Indicates that the T1CMP comparator value does not match the event timer counter. 1 Indicates that the T1CMP comparator value does match the event timer counter.

9.2.4 PHY Control 1 (Modem_PHY_CTRL1)

This register is used to control the PHY.

Address: 0h base + 3h offset = 3h

Bit	7	6	5	4	3	2	1	0
Read Write	TMRTRIGEN	SLOTTED	CCABFRTX	RXACKRQD	AUTOACK	XCVSEQ		
Reset	0	0	0	0	0	0	0	0

Modem_PHY_CTRL1 field descriptions

Field	Description
7 TMRTRIGEN	Timer 2 Trigger Enable This control bit allows the transceivers programmed sequence to be immediate or delayed by T2CMP or T2PRIMECMP count. 0 Programmed sequence initiates immediately upon write to XCVSEQ. 1 Allow timer TC2 (or TC2') to initiate a preprogrammed sequence (see XCVSEQ register).
6 SLOTTED	Slotted Mode For beacon-enabled networks. Applies only to Sequences T, TR, and R, ignored during all other sequences. Used, in concert with CCABFRTX, to determine how many CCA measurements are required prior to a transmit operation. Also used during R sequence to determine whether the ensuing transmit acknowledge frame (if any) needs to be synchronized to a backoff slot boundary. 0 Non-slotted 1 Slotted
5 CCABFRTX	CCA Before TX Applies only to sequences T and TR. Ignored during all other sequences. 0 No CCA required. Transmit operation begins immediately. 1 At least one CCA measurement is required prior to the transmit operation (see also SLOTTED).
4 RXACKRQD	Receive Acknowledge Frame Required Applies only to Sequence TR, ignored during all other sequences. 0 An ordinary receive frame (any type of frame) follows the transmit frame. 1 A receive acknowledge frame is expected to follow the transmit frame (non-acknowledge frames are rejected).
3 AUTOACK	Auto Acklowledge Enable Applies only to sequence R and sequence TR. Ignored during other sequences. 0 The sequence manager will not follow a receive frame with a Tx Ack frame, under any condition. The autosequence will terminate after the receive frame. 1 The sequence manager will follow a receive frame with an automatic hardware-generated Tx Ack frame, assuming other necessary conditions are met.
XCVSEQ	Transceiver Sequence Selector

Table continues on the next page...

Modem_PHY_CTRL1 field descriptions (continued)

Field	Description
	Selects an autosequence for the sequence manager to execute. Sequence initiation can be immediate, or scheduled (see TMRTRIGEN). A write of XCVSEQ=IDLE will abort any ongoing sequence. A write of XCVSEQ=IDLE must always be performed after a sequence is complete, and before a new sequence is programmed. Any write to XCVSEQ other than XCVSEQ=IDLE during an ongoing sequence, shall be ignored. The mapping of XCVSEQ to sequence types is as follows: 000 Idle (sequence I) 001 Receive (sequence R) 010 Transmit (sequence T) 011 CCA (sequence C) 100 Transmit/Receive (sequence TR) 101 Continuous CCA (sequence CCCA) 110 Reserved 111 Reserved

9.2.5 PHY Control 2 (Modem_PHY_CTRL2)

This register is used to control the PHY.

Address: 0h base + 4h offset = 4h

Bit	7	6	5	4	3	2	1	0
Read Write	CRC_MSK	PLL_ UNLOCK_ MSK	FILTERFAIL _MSK	RX_WMRK_ MSK	CCAMSK	RXMSK	TXMSK	SEQMSK
Reset	1	1	1	1	1	1	1	1

Modem_PHY_CTRL2 field descriptions

Field	Description
7 CRC_MSK	CRC Mask 0 The sequence manager ignores CRCVALID and considers the receive operation complete after the last octet of the frame has been received. 1 The sequence manager requires CRCVALID=1 at the end of the received frame in order for the receive operation to complete successfully; if CRCVALID=0, sequence manager will return to preamble-detect mode after the last octet of the frame has been received.
6 PLL_UNLOCK_ MSK	PLL Unlock Mask 0 Allows PLL unlock event to generate an interrupt on IRQ_B. 1 A PLL unlock event will set the PLL_UNLOCK_IRQ status bit, but an interrupt is not generated on IRQ_B.
5 FILTERFAIL_ MSK	Filter Fail Mask 0 Allows packet processor filtering Failure to generate an interrupt on IRQ_B. 1 A packet processor filtering failure will set the FILTERFAIL_IRQ status bit, but an interrupt is not generated on IRQ_B.

Table continues on the next page...

Modem_PHY_CTRL2 field descriptions (continued)

Field	Description
4 RX_WMRK_MSK	Recieve Watermark Mask 0 Allows a Received Byte Count match to the RX_WTR_MARK threshold register to generate an interrupt on IRQ_B. 1 A Received Byte Count match to the RX_WTR_MARK threshold register will set the RXWTRMRKIRQ status bit, but an interrupt is not generated on IRQ_B.
3 CCAMSK	Clear Channel Assessment Mask 0 Allows completion of a CCA operation to generate an interrupt on IRQ_B. 1 Completion of a CCA operation will set the CCAIRQ status bit, but an interrupt is not generated on IRQ_B.
2 RXMSK	Receive Mask 0 Allows completion of a RX operation to generate an interrupt on IRQ_B. 1 Completion of a RX operation will set the RXIRQ status bit, but an interrupt is not generated on IRQ_B.
1 TXMSK	Transmit Mask 0 Allows completion of a TX operation to generate an interrupt on IRQ_B. 1 Completion of a TX operation will set the TXIRQ status bit, but an interrupt is not generated on IRQ_B.
0 SEQMSK	Sequence Mask 0 Allows completion of an autosequence to generate an interrupt on IRQ_B. 1 Completion of an autosequence will set the SEQIRQ status bit, but an interrupt is not generated on IRQ_B.

9.2.6 PHY Control 3 (Modem_PHY_CTRL3)

This register is used to control the PHY.

Address: 0h base + 5h offset = 5h

Bit	7	6	5	4
Read Write	TMR4CMP_EN	TMR3CMP_EN	TMR2CMP_EN	TMR1CMP_EN
Reset	0	0	0	0

Bit	3	2	1	0
Read Write	Reserved	ASM_MSK	PB_ERR_MSK	WAKE_MSK
Reset	0	1	1	0

Modem_PHY_CTRL3 field descriptions

Field	Description
7 TMR4CMP_EN	Timer 4 Comparator Enable 0 Don’t allow an event timer match to T4CMP to set TMR4IRQ. 1 Allow an event timer match to T4CMP to set TMR4IRQ.

Table continues on the next page...

Modem_PHY_CTRL3 field descriptions (continued)

Field	Description
6 TMR3CMP_EN	Timer 3 Comparator Enable 0 Don't allow an event timer match to T3CMP to set TMR3IRQ. 1 Allow an event timer match to T3CMP to set TMR3IRQ.
5 TMR2CMP_EN	Timer 2 Comparator Enable 0 Don't allow an event timer match to T2CMP to set TMR2IRQ. 1 Allow an event timer match to T2CMP to set TMR2IRQ.
4 TMR1CMP_EN	Timer 1 Comparator Enable 0 Don't allow an event timer match to T1CMP to set TMR1IRQ. 1 Allow an event timer match to T1CMP to set TMR1IRQ.
3 Reserved	Reserved. This field is reserved.
2 ASM_MSK	ASM Mask 0 Enable AES Interrupt to assert IRQ_B. 1 Mask AES Interrupt from asserting IRQ_B.
1 PB_ERR_MSK	Packet Buffer Error Mask 0 Enable packet buffer error interrupt to assert IRQ_B. 1 Mask Packet Buffer Error Interrupt from asserting IRQ_B.
0 WAKE_MSK	Wake Mask 0 Enable Wake Interrupt to assert IRQ_B 1 Mask Wake Interrupt from asserting IRQ_B.

9.2.7 Receive Frame Length (Modem_RX_FRM_LEN)

Contents of the PHR (PHY header), or FrameLength field, of the most recently received packet.

Address: 0h base + 6h offset = 6h

Bit	7	6	5	4	3	2	1	0
Read	Reserved	RX_FRAME_LENGTH						
Write								
Reset	0	0	0	0	0	0	0	0

Modem_RX_FRM_LEN field descriptions

Field	Description
7 Reserved	Reserved. This field is reserved.

Table continues on the next page...

Modem_RX_FRM_LEN field descriptions (continued)

Field	Description
RX_FRAME_ LENGTH	Receive Frame Length This read-only field contains the contents of the PHR (PHY header), or FrameLength field, of the most recently received packet.

9.2.8 PHY Control 4 (Modem_PHY_CTRL4)

This register is used to control the PHY.

Address: 0h base + 7h offset = 7h

Bit	7	6	5	4
Read Write	TRCV_MSK	TC3TMOUT	PANCORDNTR0	CCATYPE
Reset	0	0	0	0

Bit	3	2	1	0
Read	CCATYPE		PROMISCUOUS	TC2PRIME_EN
Write		TMRLOAD		
Reset	1	0	0	0

Modem_PHY_CTRL4 field descriptions

Field	Description
7 TRCV_MSK	Transceiver IRQ_B All interrupt Mask 0 Enable any interrupt to assert IRQ_B. 1 Mask all interrupts from asserting IRQ_B.
6 TC3TMOUT	Timer 3 Function Control 0 TMR3 is a software timer only. 1 Enable TMR3 to abort Rx or CCCA operations.
5 PANCORDNTR0	PAN Coordinator on PAN0 Device Enable Device is a PAN Coordinator on PAN0. Allows device to receive packets with no destination address, if source PAN ID matches.
4–3 CCATYPE	Clear Channel Assessment Type Selects one of four possible functions for CCA or ED, listed below. 00 Energy detect 01 CCA mode 1 10 CCA mode 2 11 CCA mode 3
2 TMRLOAD	Timer Load

Table continues on the next page...

Modem_PHY_CTRL4 field descriptions (continued)

Field	Description
	A low to high transition of this bit causes the contents of register 'T1CMP[23:0]' to be loaded into the Event Timer.This is a self clearing bit and always reads zero.
1 PROMISCUOUS	Promiscuous Bypasses most packet filtering. 0 Normal mode. 1 All packet filtering except frame length checking (FrameLength =5 or greater and FrameLength =127 or less) is bypassed.
0 TC2PRIME_EN	TMR2-prime Compare Match Enable 0 Don't allow a match of the lower 16 bits of the event timer to T2PRIMECMP to set TMR2IRQ. 1 Allow a match of the lower 16 bits of Event Timer to T2PRIMECMP to set TMR2IRQ.

9.2.9 SRC Control (Modem_SRC_CTRL)

This register is used to control the SRC.

Address: 0h base + 8h offset = 8h

Bit	7	6	5	4	3	2	1	0
Read	INDEX				ACK_FRM_PND	SRCADDR_EN		
Write							INDEX_EN	INDEX_DISABLE
Reset	0	0	0	0	1	1	0	0

Modem_SRC_CTRL field descriptions

Field	Description
7–4 INDEX	Index If SRCADDR_EN=0, the value of ACK_FRM_PND is copied into the FramePending subfield of the Frame Control field of an auto-TxAck frame, if conditions are met for such a frame to be sent following a receive frame (see sequence manager section, sequence R and sequence TR). If SRCADDR_EN=1, ACK_FRM_PND is ignored, and the result of the Source Address Matching table-lookup determines the setting of the FramePending subfield of the Frame Control Field.
3 ACK_FRM_PND	Acknowledge Frame Pending If SRCADDR_EN=0, the value of ACK_FRM_PND is copied into the FramePending subfield of the Frame Control field of an auto-TxAck frame, if conditions are met for such a frame to be sent following a receive frame (see sequence manager section, sequence R and sequence TR). If SRCADDR_EN=1, ACK_FRM_PND is ignored, and the result of the Source Address Matching table-lookup determines the setting of the FramePending subfield of the Frame Control Field.
2 SRCADDR_EN	Source Address Matching Enable Part of the Source Address Matching feature. Enables the source address matching feature.
1 INDEX_EN	Source Address Matching Index Enable

Table continues on the next page...

Modem_SRC_CTRL field descriptions (continued)

Field	Description
	Part of the Source Address Matching feature. Setting INDEX_EN activates the selected INDEX in the Source Address Table. Source Address Matching logic will be allowed to compare against the stored checksum associated with this index when an incoming MAC Command data request packet is received. The INDEX_EN bit is write-only, and self-clearing. Writing a 1 to INDEX_EN will enable the selected INDEX. Writing a 0 to this bit has no effect. Reads of INDEX_EN always return zero.
0 INDEX_DISABLE	Source Address Matching Index Disable Part of the source address matching feature. Setting INDEX_DISABLE deactivates (invalidates) the selected INDEX in the Source Address Table. Source Address Matching logic will no longer compare against the stored checksum associated with this index when an incoming MAC Command data request packet is received. The INDEX_DISABLE bit is write-only, and self-clearing. Writing a 1 to INDEX_DISABLE will disable the selected INDEX. Writing a 0 to this bit has no effect. Reads of INDEX_DISABLE always return zero.

9.2.10 SRC Address SUM LSB (Modem_SRC_ADDRS_SUM_LSB)

Address: 0h base + 9h offset = 9h

Bit	7	6	5	4	3	2	1	0
Read Write	SRC_ADDRS_SUM							
Reset	0	0	0	0	0	0	0	0

Modem_SRC_ADDRS_SUM_LSB field descriptions

Field	Description
SRC_ADDRS_ SUM	Source Address Checksum LSB Part of the Source Address Matching feature. Software loads a specific index of the Source Address Table with its software-computed checksum by writing to this register, having previously selected the desired index by writing to the INDEX field of the SRC_CTRL Register. This is a 16 bit register. The 2 bytes can be written in any order. Software then activates the selected table index by asserting the INDEX_EN bit of the SRC_CTRL Register. Reads from this register return the 16-bit value from the source address table selected by the INDEX field of the SRC_CTRL register.

9.2.11 SRC Address SUM MSB (Modem_SRC_ADDRS_SUM_MSB)

Address: 0h base + Ah offset = Ah

Bit	7	6	5	4	3	2	1	0
Read Write	SRC_ADDRS_SUM							
Reset	0	0	0	0	0	0	0	0

Modem_SRC_ADDRS_SUM_MSB field descriptions

Field	Description
SRC_ADDRS_SUM	Source Address Checksum MSB Part of the Source Address Matching feature. Software loads a specific index of the Source Address Table with its software-computed checksum by writing to this register, having previously selected the desired index by writing to the INDEX field of the SRC_CTRL Register. This is a 16 bit register. The 2 bytes can be written in any order. Software then activates the selected table index by asserting the INDEX_EN bit of the SRC_CTRL Register. Reads from this register return the 16-bit value from the source address table selected by the INDEX field of the SRC_CTRL register.

9.2.12 CCA1 ED FNL (Modem_CCA1_ED_FNL)

Address: 0h base + Bh offset = Bh

Bit	7	6	5	4	3	2	1	0
Read	CCA1_ED_FNL							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_CCA1_ED_FNL field descriptions

Field	Description
CCA1_ED_FNL	CCA1/ED Final Average Value Output register to show final averaged RSSI value or compensated value of the same at the end CCA mode1/ED computations.

9.2.13 Event Timer LSB (Modem_EVENT_TIMER_LSB)

Address: 0h base + Ch offset = Ch

Bit	7	6	5	4	3	2	1	0
Read	EVENT_TIMER							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_EVENT_TIMER_LSB field descriptions

Field	Description
EVENT_TIMER	Event Timer LSB Holds the current value of the 24-bit event timer. The hardware latches the upper 2 bytes of EVENT_TMR on each read of least signficant byte.

9.2.14 Event Timer MSB (Modem_EVENT_TIMER_MSB)

Address: 0h base + Dh offset = Dh

Bit	7	6	5	4	3	2	1	0
Read	EVENT_TIMER							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_EVENT_TIMER_MSB field descriptions

Field	Description
EVENT_TIMER	Event Timer MSB Holds the current value of the 24-bit event timer. The hardware latches the upper 2 bytes of EVENT_TMR on each read of least signficant byte.

9.2.15 Event Timer USB (Modem_EVENT_TIMER_USB)

Address: 0h base + Eh offset = Eh

Bit	7	6	5	4	3	2	1	0
Read	EVENT_TIMER							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_EVENT_TIMER_USB field descriptions

Field	Description
EVENT_TIMER	Event Timer USB Holds the current value of the 24-bit event timer. The hardware latches the upper 2 bytes of EVENT_TMR on each read of least signficant byte.

9.2.16 Timestamp LSB (Modem_TIMESTAMP_LSB)

Address: 0h base + Fh offset = Fh

Bit	7	6	5	4	3	2	1	0
Read	TIMESTAMP							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_TIMESTAMP_LSB field descriptions

Field	Description
TIMESTAMP	Timestamp LSB Holds the latched value of the event timer current time corresponding to the beginning of the just received Rx packet, at SFD detect.

9.2.17 Timestamp MSB (Modem_TIMESTAMP_MSB)

Address: 0h base + 10h offset = 10h

Bit	7	6	5	4	3	2	1	0
Read	TIMESTAMP							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_TIMESTAMP_MSB field descriptions

Field	Description
TIMESTAMP	Timestamp MSB Holds the latched value of the event timer current time corresponding to the beginning of the just received Rx packet, at SFD detect.

9.2.18 Timestamp USB (Modem_TIMESTAMP_USB)

Address: 0h base + 11h offset = 11h

Bit	7	6	5	4	3	2	1	0
Read	TIMESTAMP							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_TIMESTAMP_USB field descriptions

Field	Description
TIMESTAMP	Timestamp USB Holds the latched value of the event timer current time corresponding to the beginning of the just received Rx packet, at SFD detect.

9.2.19 Timer 3 Compare Value LSB (Modem_T3CMP_LSB)

Address: 0h base + 12h offset = 12h

Bit	7	6	5	4	3	2	1	0
Read Write	T3CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T3CMP_LSB field descriptions

Field	Description
T3CMP	TMR3 Compare Value If TMR3CMP_EN=1 and the event timer matches this value, TMR3IRQ is set.

9.2.20 Timer 3 Compare Value MSB (Modem_T3CMP_MSB)

Address: 0h base + 13h offset = 13h

Bit	7	6	5	4	3	2	1	0
Read Write	T3CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T3CMP_MSB field descriptions

Field	Description
T3CMP	TMR3 Compare Value If TMR3CMP_EN=1 and the event timer matches this value, TMR3IRQ is set.

9.2.21 Timer 3 Compare Value USB (Modem_T3CMP_USB)

Address: 0h base + 14h offset = 14h

Bit	7	6	5	4	3	2	1	0
Read Write	T3CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T3CMP_USB field descriptions

Field	Description
T3CMP	TMR3 Compare Value If TMR3CMP_EN=1 and the event timer matches this value, TMR3IRQ is set.

9.2.22 Timer 2-Prime Compare Value LSB (Modem_T2PRIMECMP_LSB)

Address: 0h base + 15h offset = 15h

Bit	7	6	5	4	3	2	1	0
Read Write	T2PRIMECMP							
Reset	1	1	1	1	1	1	1	1

Modem_T2PRIMECMP_LSB field descriptions

Field	Description
T2PRIMECMP	TMR2-Prime Compare Value If TMR2CMP_EN=1 and TC2PRIME_EN=1 and the lower 16 bits of the event timer matches this value, TMR2IRQ is set.

9.2.23 Timer 2-Prime Compare Value MSB (Modem_T2PRIMECMP_MSB)

Address: 0h base + 16h offset = 16h

Bit	7	6	5	4	3	2	1	0
Read Write	T2PRIMECMP							
Reset	1	1	1	1	1	1	1	1

Modem_T2PRIMECMP_MSB field descriptions

Field	Description
T2PRIMECMP	TMR2-Prime Compare Value If TMR2CMP_EN=1 and TC2PRIME_EN=1 and the lower 16 bits of the event timer matches this value, TMR2IRQ is set.

9.2.24 Timer 1 Compare Value LSB (Modem_T1CMP_LSB)

Address: 0h base + 17h offset = 17h

Bit	7	6	5	4	3	2	1	0
Read Write	T1CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T1CMP_LSB field descriptions

Field	Description
T1CMP	TMR1 Compare Value If TMR1CMP_EN=1 and the event timer matches this value, TMR1IRQ is set.

9.2.25 Timer 1 Compare Value MSB (Modem_T1CMP_MSB)

Address: 0h base + 18h offset = 18h

Bit	7	6	5	4	3	2	1	0
Read Write	T1CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T1CMP_MSB field descriptions

Field	Description
T1CMP	TMR1 Compare Value If TMR1CMP_EN=1 and the event timer matches this value, TMR1IRQ is set.

9.2.26 Timer 1 Compare Value USB (Modem_T1CMP_USB)

Address: 0h base + 19h offset = 19h

Bit	7	6	5	4	3	2	1	0
Read Write	T1CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T1CMP_USB field descriptions

Field	Description
T1CMP	TMR1 Compare Value If TMR1CMP_EN=1 and the event timer matches this value, TMR1IRQ is set.

9.2.27 Timer 2 Compare Value LSB (Modem_T2CMP_LSB)

Address: 0h base + 1Ah offset = 1Ah

Bit	7	6	5	4	3	2	1	0
Read Write	T2CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T2CMP_LSB field descriptions

Field	Description
T2CMP	TMR2 Compare Value If TMR2CMP_EN=1 and the event timer matches this value, TMR2IRQ is set.

9.2.28 Timer 2 Compare Value MSB (Modem_T2CMP_MSB)

Address: 0h base + 1Bh offset = 1Bh

Bit	7	6	5	4	3	2	1	0
Read Write	T2CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T2CMP_MSB field descriptions

Field	Description
T2CMP	TMR2 Compare Value If TMR2CMP_EN=1 and the event timer matches this value, TMR2IRQ is set.

9.2.29 Timer 2 Compare Value USB (Modem_T2CMP_USB)

Address: 0h base + 1Ch offset = 1Ch

Bit	7	6	5	4	3	2	1	0
Read Write	T2CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T2CMP_USB field descriptions

Field	Description
T2CMP	TMR2 Compare Value If TMR2CMP_EN=1 and the event timer matches this value, TMR2IRQ is set.

9.2.30 Timer 4 Compare Value LSB (Modem_T4CMP_LSB)

Address: 0h base + 1Dh offset = 1Dh

Bit	7	6	5	4	3	2	1	0
Read Write	T4CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T4CMP_LSB field descriptions

Field	Description
T4CMP	TMR4 Compare Value If TMR4CMP_EN=1 and the event timer matches this value, TMR4IRQ is set.

9.2.31 Timer 4 Compare Value MSB (Modem_T4CMP_MSB)

Address: 0h base + 1Eh offset = 1Eh

Bit	7	6	5	4	3	2	1	0
Read Write	T4CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T4CMP_MSB field descriptions

Field	Description
T4CMP	TMR4 Compare Value If TMR4CMP_EN=1 and the event timer matches this value, TMR4IRQ is set.

9.2.32 Timer 4 Compare Value USB (Modem_T4CMP_USB)

Address: 0h base + 1Fh offset = 1Fh

Bit	7	6	5	4	3	2	1	0
Read Write	T4CMP							
Reset	1	1	1	1	1	1	1	1

Modem_T4CMP_USB field descriptions

Field	Description
T4CMP	TMR4 Compare Value If TMR4CMP_EN=1 and the event timer matches this value, TMR4IRQ is set.

9.2.33 PLL Integer Value for PAN0 (Modem_PLL_INT0)

Address: 0h base + 20h offset = 20h

Bit	7	6	5	4	3	2	1	0
Read Write	Reserved				PLL_INT0			
Reset	0	0	0	0	1	1	0	0

Modem_PLL_INT0 field descriptions

Field	Description
7–5 Reserved	This field is reserved.
PLL_INT0	PLL Frequency Integer Value for PAN0 The equation for PLL freqency is: F = ((PLL_INT0+64) + (PLL_FRAC0/65536)) * 32 MHz.

9.2.34 PLL Frequency Fractional Value for PAN0 (Modem_PLL_FRAC0_LSB)

Address: 0h base + 21h offset = 21h

Bit	7	6	5	4	3	2	1	0
Read Write	PLL_FRAC0_LSB							
Reset	0	0	0	0	0	0	0	0

Modem_PLL_FRAC0_LSB field descriptions

Field	Description
PLL_FRAC0_ LSB	PLL Frequency Fractional Value for PAN0 The equation for PLL frequency is: F = ((PLL_INT0+64) + (PLL_FRAC0/65536)) * 3 2MHz.

9.2.35 PLL Frequency Fractional Value for PAN0 (Modem_PLL_FRAC0_MSB)

Address: 0h base + 22h offset = 22h

Bit	7	6	5	4	3	2	1	0
Read Write	PLL_FRAC0_MSB							
Reset	1	0	0	1	0	0	0	0

Modem_PLL_FRAC0_MSB field descriptions

Field	Description
PLL_FRAC0_ MSB	PLL Frequency Fractional Value for PAN0 The equation for PLL frequency is: F = ((PLL_INT0+64) + (PLL_FRAC0/65536)) * 32 MHz.

9.2.36 PA Power Control (Modem_PA_PWR) (Modem_PA_PWR)

Address: 0h base + 23h offset = 23h

Bit	7	6	5	4	3	2	1	0
Read Write	Reserved			PA_PWR				
Reset	0	0	0	1	1	0	0	0

Modem_PA_PWR field descriptions

Field	Description
7–5 Reserved	This field is reserved.
PA_PWR	PA Power Control Adjusts the PA's output power. Linear range is 3 to 31 decimal.

9.2.37 Sequence Manager State (Modem_SEQ_STATE)

Address: 0h base + 24h offset = 24h

Bit	7	6	5	4	3	2	1	0
Read	SEQ_STATE							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_SEQ_STATE field descriptions

Field	Description
SEQ_STATE	Sequence Manager State This read-only register reflects the status of the sequence manager, RX manager, TX manager, and PLL manager state machines. The SEQ_STATE_CTRL[1:0] field of the SEQ_MGR_CTRL register, determines which sequence manager state is monitored by this register. See the sequence manager section for the mapping.

9.2.38 Link Quality Indicator (Modem_LQI_VALUE)

Address: 0h base + 25h offset = 25h

Bit	7	6	5	4	3	2	1	0
Read	LQI_VALUE							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_LQI_VALUE field descriptions

Field	Description
LQI_VALUE	Link Quality Indicator This is the link quality indicator for the most recently received packet. (LQI is also available in the packet buffer, at the end of the received packet data; see packet buffer section for details). See the CCA section for details on the LQI algorithm.

9.2.39 RSSI CCA CNT (Modem_RSSI_CCA_CNT)

Address: 0h base + 26h offset = 26h

Bit	7	6	5	4	3	2	1	0
Read	RSSI_CCA_CNT							
Write								
Reset	0	0	0	0	0	0	0	0

Modem_RSSI_CCA_CNT field descriptions

Field	Description
RSSI_CCA_CNT	RSSI/CCA Continuous Value Continuous update of average RSSI value, available throughout packet reception.

9.2.40 ASM Control 1 (Modem_ASM_CTRL1)

Address: 0h base + 28h offset = 28h

Bit	7	6	5	4	3	2	1	0
Read	0	0	SELFTST	CTR	CBC	AES	0	Reserved
Write	CLEAR	START					LOAD_MAC	
Reset	0	0	0	0	0	0	0	0

Modem_ASM_CTRL1 field descriptions

Field	Description
7 CLEAR	Clear Write 1 to clear all memory in ASM. Writing a 0 does nothing. See the ASM section for programming details.
6 START	Start Write 1 to start the CBC or CTR or CCM or AES mode encryption. If SELFTST is set to 1, then start will start selftest mode.
5 SELFTST	Self Test

Table continues on the next page...

Modem_ASM_CTRL1 field descriptions (continued)

Field	Description
	Write 1 to start set mode to self test (not self clearing). 0 Module not in selftest mode. 1 Module in selftest mode.
4 CTR	Need Name Write 1 to put ASM into counter encryption mode. 0 Counter mode encryption will not be performed. 1 Counter mode encryption will be performed.
3 CBC	Need Name Write 1 to put ASM into CBC-MAC mode. 0 CBC-MAC generation will not be performed. 1 CBC-MAC generation will be performed.
2 AES	Need Name Write 1 to put ASM into AES mode. 0 AES mode encrption will not be performed. 1 AES mode encrption will be performed.
1 LOAD_MAC	Need Name Write 1 to pre-load MAC with a value. Writing a 0 does nothing.
0 Reserved	Reserved This field is reserved.

9.2.41 ASM Control 2 (Modem_ASM_CTRL2)

Address: 0h base + 29h offset = 29h

Bit	7	6	5	4	3	2	1	0
Read	DATA_REG_TYPE_SELECT			Reserved			LOAD_MAC	Reserved
Write								
Reset	0	0	0	0	0	0	0	0

Modem_ASM_CTRL2 field descriptions

Field	Description
7–5 DATA_REG_ TYPE_SELECT	Data Register Type Select This register selects the type of data to be written to or read from the 16x8 Data register field. 000 Key: The 16 AES Data register holds the key used for encryption. 001 Data: The 16 AES Data register hold the data to be encrypted or decrypted 010 CTR: The 16 AES Data register hold the counter value used to encrypt the data in counter mode.

Table continues on the next page...

Modem_ASM_CTRL2 field descriptions (continued)

Field	Description
	011 CTR Result: The 16 AES Data register hold the counter mode encyption result 100 CBC Result: The 16 AES Data register hold CBC-MAC generated results. 101 MAC: The 16 AES Data register hold the starting value for CBC-MAC generation. 110 AES Result: The 16 AES Data register hold the plain AES mode encryption result. 111 Reserved.
4–2 Reserved	Reserved This field is reserved.
1 LOAD_MAC	Test Pass If this bit is 1 then the hardware passed the self test. If self test fails then the module is not usable and the outputs will be set to all 0. The default state for this bit is 0. The software must initiate a self test before the module can be used.
0 Reserved	Reserved This field is reserved.

9.2.42 ASM Data (Modem_ASM_DATA*n*)

Address: 0h base + 2Ah offset + (1d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read Write	ASM_DATA							
Reset	0	0	0	0	0	0	0	0

Modem_ASM_DATA*n* field descriptions

Field	Description
ASM_DATA	ASM Data The 16x8 AES Data registers (128 bits) block serve as the interface between the ASM's internal memory segments and the external MCU. The destination or source of the data in these registers is controlled through the DATA_REG_TYPE_SELECT field in ASM_CTRL2. See the ASM section for programming detail. Register ASM_DATA_0 contains the fields ASM_DATA[7] through ASM_DATA_[0], ASM_DATA_1 contains the fields ASM_DATA[15] through ASM_DATA[8].

9.2.43 Overwrite Version Number (Modem_OVERWRITE_VER)

Address: 0h base + 3Bh offset = 3Bh

Bit	7	6	5	4	3	2	1	0
Read Write	OVERWRITE_VER							
Reset	0	0	0	0	0	0	0	0

Modem_OVERWRITE_VER field descriptions

Field	Description
OVERWRITE_VER	Overwrite Version Number Contains the version number of the overwrites.h file, used by stack software to write non-default values into the radio's register map.

9.2.44 CLK_OUT Control (Modem_CLK_OUT_CTRL)

This register provides control for CLK_OUT.

Address: 0h base + 3Ch offset = 3Ch

Bit	7	6	5	4
Read Write	CLK_OUT_EXTEND	CLK_OUT_HIZ	CLK_OUT_SR	CLK_OUT_DS
Reset	1	0	0	0

Bit	3	2	1	0
Read Write	CLK_OUT_EN	CLK_OUT_DIV		
Reset	1	0	0	0

Modem_CLK_OUT_CTRL field descriptions

Field	Description
7 CLK_OUT_EXTEND	CLK_OUT Extend Briefly extend CLK_OUT after deassertion of CLK_OUT_EN. 0 Extends CLK_OUT for 128 clock cycles after the deassertion of CLK_OUT_EN. 1 Deassertion of CLK_OUT_EN immediately stops CLK_OUT.
6 CLK_OUT_HIZ	CLK_OUT High Impedence Select Selects whether CLK_OUT is placed into high-impedence mode or output mode. 0 CLK_OUT pad is in output mode (Output Buffer Enable=1, and Input Buffer Enable=0). 1 Place the CLK_OUT pad in high-impedence mode (Output Buffer Enable=0, and Input Buffer Enable=0).
5 CLK_OUT_SR	CLK_OUT Slew Rate Enables or disables slew rate limiting for CLK_OUT pad. 0 Disable slew rate limiting for CLK_OUT pad. 1 Enable slew rate limiting for CLK_OUT pad.
4 CLK_OUT_DS	CLK_OUT Drive Strength Selects the drive strength for CLK_OUT. 0 Normal drive strength for CLK_OUT pad. 1 High drive strength for CLK_OUT pad.

Table continues on the next page...

Modem_CLK_OUT_CTRL field descriptions (continued)

Field	Description
3 CLK_OUT_EN	Transmitter Interrupt Status Shows the status of a transmit operation. This is write a 1 to clear bit. 0 CLK_OUT pad is held at 0 (assuming CLK_OUT_HIZ=0). 1 CLK_OUT pad is driven with a clock whose frequency is determined by the CLK_OUT_DIV setting.
CLK_OUT_DIV	CLK_OUT Divide If programming CLK_OUT to >8 MHz, it is advisable to set CLK_OUT_DS=1. (High drive strength CLK_OUT pad.) The CLK_OUT frequency (if CLK_OUT_EN=1), is as follows: 000 32 MHz 001 16 MHz 010 8 MHz 011 4 MHz 100 1 MHz 101 250 kHz 110 62.5 kHz 111 32.787 kHz (32M Hz / 976)

9.2.45 Power Modes (Modem_PWR_MODES)

This register determines the power modes.

Address: 0h base + 3Dh offset = 3Dh

Bit	7	6	5	4
Read	Reserved		XTAL_READY	XTALEN
Write				
Reset	0	0	0	1

Bit	3	2	1	0
Read	ASM_CLK_EN	Reserved	AUTODOZE	PMC_MODE
Write				
Reset	0	0	0	1

Modem_PWR_MODES field descriptions

Field	Description
7–6 Reserved	Reserved This field is reserved.
5 XTAL_READY	XTAL Ready Indicates whether the 32 MHz crystal oscillator has completed its warmup and is supplying clocks to the digital core. Read-only.

Table continues on the next page...

Modem_PWR_MODES field descriptions (continued)

Field	Description
	0 The 32 MHz crystal oscillator has not completed its warmup and is supplying clocks to the digital core.. 1 The 32 MHz crystal oscillator has completed its warmup and is supplying clocks to the digital core.
4 XTALEN	CLK_OUT Slew Rate HIBERNATE mode can be entered by setting XTALEN=0 and PMC_MODE=0. DOZE mode can be entered by setting XTALEN=1 and PMC_MODE=0. IDLE mode can be entered by setting XTALEN=1 and PMC_MODE=1. A programmed low-to-high transition on XTALEN will cause the WAKE_IRQ and WAKE_FROM_HIB status bits to become set, after the oscillator warmup has completed (see IRQSTS2 register). XTALEN=1 is mandatory for all transceiver operations, including transceiver timers. 0 Disable the 32 MHz crystal oscillator. 1 Enable the 32 MHz crystal oscillator.
3 ASM_CLK_EN	ASM Clock Enable.
2 Reserved	Reserved This field is reserved.
1 AUTODOZE	Automatic Doze Mode Automatic Doze mode allows the PMC operating mode to be controlled automatically by the sequence manager. In this mode, the PMC will be in high-power mode during all transceiver sequences (i.e., Run state), and in low power mode at all other times (i.e., Doze state). If AUTODOZE=0, then PMC operating mode is under direct software control (PMC_MODE bit) 0 Disable Automatic Doze mode. 1 Enable Automatic Doze mode.
0 PMC_MODE	PMC Mode Selects whether the PMC is in high-power mode or low-power mode. 0 PMC is in low-power mode. 1 PMC is in high-power mode.

9.2.46 IAR Index (Modem_IAR_INDEX)

Address: 0h base + 3Eh offset = 3Eh

Bit	7	6	5	4	3	2	1	0
Read								
Write	IAR_INDEX							
Reset	0	0	0	0	0	0	0	0

Modem_IAR_INDEX field descriptions

Field	Description
IAR_INDEX	IAR Index

Modem_IAR_INDEX field descriptions (continued)

Field	Description
	A pointer into Indirect Access address space. During an SPI register access, when IAR_INDEX appears as the address in the control word (first SPI byte), the following byte will be loaded into IAR_INDEX, and the byte after that will write (or read) a byte from the indirect access register pointed to by IAR_INDEX. Hardware will automatically increment the pointer for subsequent byte accesses within the SPI burst transfer. See the SPI chapter for more information. Indirect Access auto-incrementing is not available in Hibernate state and should not be attempted; only single-access indirect register transfers are available in Hibernate. Direct address auto-incrementing is available in all power states. IAR_INDEX is not affected by any soft reset bit (see SOFT_RESET register).

9.2.47 IAR Data (Modem_IAR_DATA)

Address: 0h base + 3Fh offset = 3Fh

Bit	7	6	5	4	3	2	1	0
Read Write	IAR_DATA							
Reset	0	0	0	0	0	0	0	0

Modem_IAR_DATA field descriptions

Field	Description
IAR_DATA	IAR DATA Window into Indirect Access Space. Data byte to be written (or read) to (from) this address during an Indirect Register Access will be written to (or read from) the Indirect Register pointed to by IAR_INDEX for the first data phase of the transfer, and the Indirect Address pointer will be automatically incremented for subsequent data phases during the SPI burst transfer. Indirect Access auto-incrementing is not available in Hibernate state and should not be attempted. Only single-access indirect register transfers are available in Hibernate. Direct address auto-incrementing is available in all power states. IAR_INDEX is not affected by any soft reset bit (see SOFT_RESET register). The read-write access of this register is determined by the read-write access of the register in Indirect Access space to which the IAR_INDEX is pointing.

9.3 Indirect registers memory map and register definition

The Indirect radio registers contains many bitfields for communicating with the radio.

NOTE

The Direct and Indirect registers can be accessed only through the SPI interface. Direct access by a single SPI control word and Indirect using the pointer register (IAR_INDEX) and the data register (IAR_DATA), which occupy the last two registers in direct access; addresses 0x3E and 0x3F, respectively.

Indirect_Modem memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	Part Identification (Indirect_Modem_PART_ID)	8	R		9.3.1/165
1	XTAL 32 MHz Trim (Indirect_Modem_XTAL_TRIM)	8	R/W	77h	9.3.2/165
3	MAC PAN ID for PAN0 (Indirect_Modem_MACPANID0_LSB)	8	R/W	FFh	9.3.3/166
4	MAC PAN ID for PAN0 (Indirect_Modem_MACPANID0_MSB)	8	R/W	FFh	9.3.3/166
5	MAC Short Address for PAN0 (Indirect_Modem_MACSHORTADDRS0_LSB)	8	R/W	FFh	9.3.4/166
6	MAC Short Address for PAN0 (Indirect_Modem_MACSHORTADDRS0_MSB)	8	R/W	FFh	9.3.4/166
7	MAC Long Address for PAN0 (Indirect_Modem_MACLONGADDRS0_0)	8	R/W	FFh	9.3.5/167
8	MAC Long Address for PAN0 (Indirect_Modem_MACLONGADDRS0_1)	8	R/W	FFh	9.3.5/167
9	MAC Long Address for PAN0 (Indirect_Modem_MACLONGADDRS0_2)	8	R/W	FFh	9.3.5/167
A	MAC Long Address for PAN0 (Indirect_Modem_MACLONGADDRS0_3)	8	R/W	FFh	9.3.5/167
B	MAC Long Address for PAN0 (Indirect_Modem_MACLONGADDRS0_4)	8	R/W	FFh	9.3.5/167
C	MAC Long Address for PAN0 (Indirect_Modem_MACLONGADDRS0_5)	8	R/W	FFh	9.3.5/167
D	MAC Long Address for PAN0 (Indirect_Modem_MACLONGADDRS0_6)	8	R/W	FFh	9.3.5/167
E	MAC Long Address for PAN0 (Indirect_Modem_MACLONGADDRS0_7)	8	R/W	FFh	9.3.5/167
F	Receive Frame Filter (Indirect_Modem_RX_FRAME_FILTER)	8	R/W	0Fh	9.3.6/167
10	Frequency Integer for PAN1 (Indirect_Modem_PLL_INT1)	8	R/W	1Fh	9.3.7/168
11	Frequency Fractional Value for PAN1 (Indirect_Modem_PLL_FRAC1_LSB)	8	R/W	FFh	9.3.8/169
12	Frequency Fractional Value for PAN1 (Indirect_Modem_PLL_FRAC1_MSB)	8	R/W	FFh	9.3.8/169
13	Frequency Fractional Value for PAN1 (Indirect_Modem_MACPANID1_LSB)	8	R/W	FFh	9.3.9/169
14	Frequency Fractional Value for PAN1 (Indirect_Modem_MACPANID1_MSB)	8	R/W	FFh	9.3.9/169
15	MAC Short Address for PAN1 (Indirect_Modem_MACSHORTADDRS1_LSB)	8	R/W	FFh	9.3.10/170
16	MAC Short Address for PAN1 (Indirect_Modem_MACSHORTADDRS1_MSB)	8	R/W	FFh	9.3.10/170
17	MAC Long Address for PAN1 (Indirect_Modem_MACLONGADDRS1_0)	8	R/W	FFh	9.3.11/170

Table continues on the next page...

Indirect_Modem memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
18	MAC Long Address for PAN1 (Indirect_Modem_MACLONGADDRS1_1)	8	R/W	FFh	9.3.11/170
19	MAC Long Address for PAN1 (Indirect_Modem_MACLONGADDRS1_2)	8	R/W	FFh	9.3.11/170
1A	MAC Long Address for PAN1 (Indirect_Modem_MACLONGADDRS1_3)	8	R/W	FFh	9.3.11/170
1B	MAC Long Address for PAN1 (Indirect_Modem_MACLONGADDRS1_4)	8	R/W	FFh	9.3.11/170
1C	MAC Long Address for PAN1 (Indirect_Modem_MACLONGADDRS1_5)	8	R/W	FFh	9.3.11/170
1D	MAC Long Address for PAN1 (Indirect_Modem_MACLONGADDRS1_6)	8	R/W	FFh	9.3.11/170
1E	MAC Long Address for PAN1 (Indirect_Modem_MACLONGADDRS1_7)	8	R/W	FFh	9.3.11/170
1F	Dual PAN Control (Indirect_Modem_DUAL_PAN_CTRL)	8	R/W	FFh	9.3.12/171
20	Channel Frequency Dwell Time (Indirect_Modem_DUAL_PAN_DWELL)	8	R/W	00h	9.3.13/172
21	Dual PAN Status (Indirect_Modem_DUAL_PAN_STS)	8	R	00h	9.3.14/172
22	Clear Channel Assessment 1 Threshold (Indirect_Modem_CCA1_THRESH)	8	R/W	4Bh	9.3.15/173
23	Clear Channel Assessment / ED Offset Computation (Indirect_Modem_CCA1_ED_OFFSET_COMP)	8	R/W	6Dh	9.3.16/174
24	LQI Offset Computation (Indirect_Modem_LQI_OFFSET_COMP)	8	R/W	24h	9.3.17/174
25	CCA Control (Indirect_Modem_CCA_CTRL)	8	R/W	5Fh	9.3.18/174
26	Clear Channel Assessment 2 Threshold Peak Compare (Indirect_Modem_CCA2_CORR_PEAKS)	8	R/W	60h	9.3.19/175
27	Clear Channel Assessment 2 Threshold (Indirect_Modem_CCA2_THRESH)	8	R/W	82h	9.3.20/176
28	TMR PRESCALE (Indirect_Modem_TMR_PRESCALE)	8	R/W	03h	9.3.21/176
2A	GPIO Data (Indirect_Modem_GPIO_DATA)	8	R/W	00h	9.3.22/177
2B	GPIO Direction Control (Indirect_Modem_GPIO_DIR)	8	R/W	00h	9.3.23/178
2C	GPIO Pullup Enable (Indirect_Modem_GPIO_PUL_EN)	8	R/W	FFh	9.3.24/179
2D	GPIO Pullup Select (Indirect_Modem_GPIO_SEL)	8	R/W	FFh	9.3.25/181
2E	GPIO Drive Strength (Indirect_Modem_GPIO_DS)	8	R/W	FFh	9.3.26/182
30	Antenna Control (Indirect_Modem_ANT_PAD_CTRL)	8	R/W	08h	9.3.27/183
31	Miscellaneous Pad Control (Indirect_Modem_MISC_PAD_CTRL)	8	R/W	1Ah	9.3.28/185
35	RX_BYTE_COUNT (Indirect_Modem_RX_BYTE_COUNT)	8	R	00h	9.3.29/185
36	RX_WTR_MARK (Indirect_Modem_RX_WTR_MARK)	8	R/W	FFh	9.3.30/186
38	TXDELAY (Indirect_Modem_TXDELAY)	8	R/W	00h	9.3.31/186
39	ACKDELAY (Indirect_Modem_ACKDELAY)	8	R/W	3Dh	9.3.32/187

Table continues on the next page...

Indirect_Modem memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
51	Antenna AGC and FAD Control (Indirect_Modem_ANT_AGC_CTRL)	8	R/W	40h	9.3.33/187
56	LPPS_CTRL (Indirect_Modem_LPPS_CTRL)	8	R/W	1Eh	9.3.34/188
5B	RSSI (Indirect_Modem_RSSI)	8	R	00h	9.3.35/189
6E	XTAL Control (Indirect_Modem_XTAL_CTRL)	8	R/W	03h	9.3.36/190

9.3.1 Part Identification (Indirect_Modem_PART_ID)

The Part Identification register contains information on the mask set, the version, and the manufacturer of the transceiver.

Address: 0h base + 0h offset = 0h

Bit	7	6	5	4	3	2	1	0
Read	MANUF_ID		VERSION			MASK_SET		
Write								
Reset								

Indirect_Modem_PART_ID field descriptions

Field	Description
7–6 MANUF_ID	Manufacturer ID
5–3 VERSION	Version This field contains information about the quadrature version of the LNA mixer and VCO. 001 Single quadrature version of the LNA mixer and VCO. 010 Double quadrature version of the LNA mixer and VCO.
MASK_SET	Mask Set This field contains information about the mask set of the transceiver.

9.3.2 XTAL 32 MHz Trim (Indirect_Modem_XTAL_TRIM)

Address: 0h base + 1h offset = 1h

Bit	7	6	5	4	3	2	1	0
Read Write	XTAL_TRIM							
Reset	0	1	1	1	0	1	1	1

Indirect_Modem_XTAL_TRIM field descriptions

Field	Description
XTAL_TRIM	XTAL 32 MHz Trim XTAL 32 MHz trimming. See the XTAL32M section for more information.

9.3.3 MAC PAN ID for PAN0 (Indirect_Modem_MACPANID0*n*)

Address: 0h base + 3h offset + (1d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read Write	MACPANID0							
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_MACPANID0*n* field descriptions

Field	Description
MACPANID0	MAC PAN ID for PAN0 The packet processor compares the incoming packet's destination PAN ID against the contents of this register to determine whether the packet is addressed to this device. Or, if the incoming packet is a Beacon frame, the packet processor compares the incoming packet source PAN ID against this register. Also, if PANCORDNTR0=1, and the incoming packet has no Destination Address field, and if the incoming packet is a Data or MAC Command frame, the packet processor compares the incoming packet source PAN ID against this register.

9.3.4 MAC Short Address for PAN0 (Indirect_Modem_MACSHORTADDRS0*n*)

Address: 0h base + 5h offset + (1d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read Write	MACSHORT_ADDRS0							
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_MACSHORTADDRS0*n* field descriptions

Field	Description
MACSHORT_ ADDRS0	MAC Short Address for PAN0 MAC Short Address for PAN0, for 16-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine whether the packet is addressed to this device.

9.3.5 MAC Long Address for PAN0 (Indirect_Modem_MACLONGADDRS0*n*)

Address: 0h base + 7h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read Write	MACLONGADDRS0							
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_MACLONGADDRS0*n* field descriptions

Field	Description
MACLONGADDRS0	MAC Long Address for PAN0 MAC Long Address for PAN0, for 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine whether the packet is addressed to this device.

9.3.6 Receive Frame Filter (Indirect_Modem_RX_FRAME_FILTER)

Address: 0h base + Fh offset = Fh

Bit	7	6	5	4
Read Write	FRM_VER		ACTIVE_ PROMISCUOUS	NS_FT
Reset	0	0	0	0

Bit	3	2	1	0
Read Write	CMD_FT	ACK_FT	DATA_FT	BEACON_FT
Reset	1	1	1	1

Indirect_Modem_RX_FRAME_FILTER field descriptions

Field	Description
7–6 FRM_VER	Frame Version Selector The incoming packet's Frame Control Field is parsed to obtain the FrameVersion subfield, and that value is compared against this register, in accordance with the following. Frames received with FrameVersion 2 or 3 will be treated identically to FrameVersion 1, with respect to parsing of the Auxiliary Security Header. Other than this Header, all 4 frame versions will be treated identically 00 Any FrameVersion accepted (0, 1, 2 and 3). 01 Accept only FrameVersion 0 packets (2003 compliant). 10 Accept only FrameVersion 1 packets (2006 compliant). 11 Accept FrameVersion 0 and 1 packets; reject all others.
5 ACTIVE_ PROMISCUOUS	Active Promiscuous

Table continues on the next page...

Indirect_Modem_RX_FRAME_FILTER field descriptions (continued)

Field	Description
	0 Normal operation (default). 1 Provide data indication on all received packets under the same rules that apply in Promiscuous mode, however acknowledge those packets under rules that apply in non-Promiscuous mode
4 NS_FT	Not-Specified (Reserved) Frame Type Field determines whether Not-Specified (Reserved) frame types are enabled. 0 Reject all reserved frames. 1 Not-Specified (reserved) frame type enabled. No packet filtering is performed, except for frame length checking (FrameLength is greater than or equal to 5 and FrameLength is less than or equal to 127).
3 CMD_FT	MAC Command Frame Type Field determines whether MAC Command frame types are enabled. 0 Reject all MAC Command frames. 1 MAC Command frame type enabled.
2 ACK_FT	Acknowledge Frame Type Field determines whether Acknowledge frame types are enabled. 0 Reject all Acknowledge frames. 1 Acknowledge frame type enabled.
1 DATA_FT	Data Frame Type Field determines whether Data frame types are enabled. 0 Reject all Data frames. 1 Data frame type enabled.
0 BEACON_FT	Beacon Frame Type Field determines whether Beacon frame types are enabled. 0 Reject all Beacon frames. 1 Beacon frame type enabled.

9.3.7 Frequency Integer for PAN1 (Indirect_Modem_PLL_INT1)

Address: 0h base + 10h offset = 10h

Bit	7	6	5	4	3	2	1	0
Read Write	Reserved				PLL_INT1			
Reset	0	0	0	1	1	1	1	1

Indirect_Modem_PLL_INT1 field descriptions

Field	Description
7–5 RESERVED	This field is reserved.

Table continues on the next page...

Indirect_Modem_PLL_INT1 field descriptions (continued)

Field	Description
PLL_INT1	Frequency Integer for PAN1 F = ((PLL_INT1+64) + (PLL_FRAC1/65536)) * 32 MHz. See PLL section for more details. This register should not be programmed (should be left in its default state) if Dual PAN mode is not in use.

9.3.8 Frequency Fractional Value for PAN1 (Indirect_Modem_PLL_FRAC1*n*)

Address: 0h base + 11h offset + (1d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read Write	PLL_FRAC1							
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_PLL_FRAC1*n* field descriptions

Field	Description
PLL_FRAC1	Frequency Fractional Value for PAN1 F = ((PLL_INT1+64) + (PLL_FRAC1/65536)) * 32 MHz. This register should not be programmed (should be left in its default state) if Dual PAN mode is not in use.

9.3.9 Frequency Fractional Value for PAN1 (Indirect_Modem_MACPANID1*n*)

Address: 0h base + 13h offset + (1d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read Write	MACPANID1							
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_MACPANID1*n* field descriptions

Field	Description
MACPANID1	MAC PAN ID for PAN1 The packet processor compares the incoming packet's Destination PAN ID against the contents of this register to determine if the packet is addressed to this device; or if the incoming packet is a Beacon frame, the packet processor compares the incoming packet source PAN ID against this register. Also, if PANCORDNTR0=1, and the incoming packet has no Destination Address field, and if the incoming packet is a Data or MAC Command frame, the packet processor compares the incoming packet source PAN ID against this register.

9.3.10 MAC Short Address for PAN1 (Indirect_Modem_MACSHORTADDRS1*n*)

Address: 0h base + 15h offset + (1d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read Write	MACSHORTADDRS1							
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_MACSHORTADDRS1*n* field descriptions

Field	Description
MACSHORTADDRS1	MAC Short Address for PAN1 For 16-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine if the packet is addressed to this device.

9.3.11 MAC Long Address for PAN1 (Indirect_Modem_MACLONGADDRS1*n*)

Address: 0h base + 17h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read Write	MACLONGADDRS1							
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_MACLONGADDRS1*n* field descriptions

Field	Description
MACLONGADDRS1	MAC Long Address for PAN1 For 64-bit destination addressing mode. The packet processor compares the incoming packet's Destination Address against the contents of this register to determine whether the packet is addressed to this device.

9.3.12 Dual PAN Control (Indirect_Modem_DUAL_PAN_CTRL)

Address: 0h base + 1Fh offset = 1Fh

Bit	7	6	5	4
Read Write	DUAL_PAN_SAM_LVL			
Reset	1	1	1	1

Bit	3	2	1	0
Read	CURRENT_NETWORK	PANCORDNTR1	DUAL_PAN_AUTO	ACTIVE_NETWORK
Write				
Reset	1	1	1	1

Indirect_Modem_DUAL_PAN_CTRL field descriptions

Field	Description
7–4 DUAL_PAN_ SAM_LVL	Source Address Matching Level For the Source Address Matching Table, sets the dividing line between the PAN0 and PAN1 sections of the table. Table indeces at or above this level belong to PAN1. Table entries below this level belong to PAN0. Default = 12 (entire table is PAN0). In Dual PAN mode, if both PANs occupy the same channel, hardware detects this and makes the entire Source Address Matching Table available to both PANs simultaneously; in this case, DUAL_PAN_SAM_LVL has no effect.
3 CURRENT_ NETWORK	Current Network This read-only bit indicates which PAN is currently selected by hardware in automatic Dual PAN mode. 0 PAN0 is selected. 1 PAN1 is selected.
2 PANCORDNTR1	PAN Coordinator on PAN1 Device is a PAN Coordinator on PAN1. Allows device to receive packets with no destination address, if Source PAN ID matches.
1 DUAL_PAN_ AUTO	Dual Pan Automatic Operating Mode Activates automatic Dual PAN operating mode. In this mode, PAN-switching is controlled by hardware at a pre-programmed rate, determined by DUAL_PAN_DWELL. Whenever DUAL_PAN_AUTO=0, CURRENT_NETWORK=ACTIVE_NETWORK at all times. In other words, software directly controls which PAN is selected. Whenever DUAL_PAN_AUTO=1, CURRENT_NETWORK is controlled by hardware. See the packet processor section for more detail. 0 Manual Dual PAN mode (or Single PAN mode). 1 Auto Dual PAN Mode.
0 ACTIVE_ NETWORK	Active Network Selects the PAN on which to transceive, by activating a PAN parameter set (PAN0 or PAN1). In Manual Dual PAN mode (or Single PAN mode), this bit selects the active PAN parameter set (channel and addressing parameters) which governs all autosequences. In Auto Dual PAN mode, this bit selects the PAN on which to begin transceiving, latched at the point at which DUAL_PAN_DWELL register is written. See packet processor section for more detail.

Table continues on the next page...

Indirect_Modem_DUAL_PAN_CTRL field descriptions (continued)

Field	Description
	0 Select PAN0. 1 Select PAN1.

9.3.13 Channel Frequency Dwell Time (Indirect_Modem_DUAL_PAN_DWELL)

Address: 0h base + 20h offset = 20h

Bit	7	6	5	4	3	2	1	0
Read Write	DUAL_PAN_DWELL							
Reset	0	0	0	0	0	0	0	0

Indirect_Modem_DUAL_PAN_DWELL field descriptions

Field	Description
DUAL_PAN_ DWELL	Channel Frequency Dwell Time In Auto Dual PAN mode, hardware will toggle the PAN, after dwelling on the current PAN for the interval described below (assuming Preamble/SFD not detected). A write to DUAL_PAN_DWELL, always re-initilizes the DWELL TIMER to the programmed value. If a write to DUAL_PAN_DWELL occurs during an autosequence, the DWELL TIMER will begin counting down immediately. If a write to DUAL_PAN_DWELL occurs when there is no autosequence underway, the DWELL TIMER will not begin counting until the next autosequence begins; it will begin counting at the start of the sequence warmup. In Auto Dual PAN mode, hardware will toggle the PAN, after dwelling on the current PAN for the interval described below (assuming Preamble/SFD not detected). A write to DUAL_PAN_DWELL, always re-initilizes the DWELL TIMER to the programmed value. If a write to DUAL_PAN_DWELL occurs during an autosequence, the DWELL TIMER will begin counting down immediately. If a write to DUAL_PAN_DWELL occurs when there is no autosequence underway, the DWELL TIMER will not begin counting until the next autosequence begins; it will begin counting at the start of the sequence warmup. A write to DUAL_PAN_DWELL also causes the value of ACTIVE_NETWORK to get latched into the hardware. This latched value will be the starting point for the automatic dual-pan mode (i.e., start on PAN0 or on PAN1). The starting value takes effect immediately (if sequence is underway and DUAL_PAN_AUTO=1), or is otherwise delayed until sequence starts and DUAL_PAN_AUTO=1. DUAL_PAN_DWELL should not be accessed in the Hibernate state.

9.3.14 Dual PAN Status (Indirect_Modem_DUAL_PAN_STS)

Address: 0h base + 21h offset = 21h

Bit	7	6	5	4	3	2	1	0
Read	RECD_ON_ PAN1	RECD_ON_ PAN0	DUAL_PAN_REMAIN					
Write								
Reset	0	0	0	0	0	0	0	0

Indirect_Modem_DUAL_PAN_STS field descriptions

Field	Description
7 RECD_ON_ PAN1	Received on PAN1 Indicates the packet that was just received, was received on PAN1. In Dual PAN mode operating on 2 different channels, RECD_ON_PAN0 will be set if CURRENT_NETWORK=0 when the packet was received, regardless of FILTERFAIL status. In Dual PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN0 will be set only if a valid packet was received on PAN0 (PAN0's FILTERFAIL_FLAG is deasserted). RECD_ON_PAN0 remains valid until the start of the next autoseqeuence. See the packet processor section for more detail.
6 RECD_ON_ PAN0	Received on PAN0 Indicates the packet that was just received, was received on PAN0. In Dual PAN mode operating on 2 different channels, RECD_ON_PAN0 will be set if CURRENT_NETWORK=0 when the packet was received, regardless of FILTERFAIL status. In Dual PAN mode operating with same channel on both networks, CURRENT_NETWORK will be ignored and RECD_ON_PAN0 will be set only if a valid packet was received on PAN0 (PAN0's FILTERFAIL_FLAG is deasserted). RECD_ON_PAN0 remains valid until the start of the next autoseqeuence. See the packet processor section for more detail.
DUAL_PAN_ REMAIN	Dual PAN Remaining Time This read-only register indicates time remaining before next PAN switch in auto Dual PAN mode. The units for this register, depend on the PRESCALER setting (bits [1:0]) in the DUAL_PAN_DWELL register. The readback value indicates that between N-1 and N timebase units remain until the next PAN switch. For example, a DUAL_PAN_REMAIN readback value of 3, with a DUAL_PAN_DWELL PRESCALER setting of 2 (10 ms), indicates that between 20 ms (2 * 10 ms) and 30 ms (3 * 10 ms), remain until the next automatic PAN switch. 00 0.5 ms 01 2.5 ms 10 10 ms 11 50 ms

9.3.15 Clear Channel Assessment 1 Threshold (Indirect_Modem_CCA1_THRESH)

Address: 0h base + 22h offset = 22h

Bit	7	6	5	4	3	2	1	0
Read Write	CCA1_THRESH							
Reset	0	1	0	0	1	0	1	1

Indirect_Modem_CCA1_THRESH field descriptions

Field	Description
CCA1_THRESH	Clear Channel Assessment 1 Threshold Programmable energy threshold register to detect CCA mode 1.

9.3.16 Clear Channel Assessment / ED Offset Computation (Indirect_Modem_CCA1_ED_OFFSET_COMP)

Address: 0h base + 23h offset = 23h

Bit	7	6	5	4	3	2	1	0
Read Write	CCA1_ED_OFFSET_COMP							
Reset	0	1	1	0	1	1	0	1

Indirect_Modem_CCA1_ED_OFFSET_COMP field descriptions

Field	Description
CCA1_ED_ OFFSET_COMP	Clear Channel Assessment / ED Offset Computation Programmable amount to offset CCA/ED computations.

9.3.17 LQI Offset Computation (Indirect_Modem_LQI_OFFSET_COMP)

Address: 0h base + 24h offset = 24h

Bit	7	6	5	4	3	2	1	0
Read Write	LQI_OFFSET_COMP							
Reset	0	0	1	0	0	1	0	0

Indirect_Modem_LQI_OFFSET_COMP field descriptions

Field	Description
LQI_OFFSET_ COMP	LQI Offset Computation Programmable amount to offset RSSI based LQI value.

9.3.18 CCA Control (Indirect_Modem_CCA_CTRL)

Address: 0h base + 25h offset = 25h

Bit	7	6	5	4
Read Write	Reserved	AGC_FRZ_EN	CONT_RSSI_EN	LQI_RSSI_NOT_CORR
Reset	0	1	0	1

Bit	3	2	1	0
Read Write	CCA3_AND_NOT_OR	OWER_COMP_EN_ LQI	OWER_COMP_EN_ED	OWER_COMP_EN_ CCA1
Reset	1	1	1	1

Indirect_Modem_CCA_CTRL field descriptions

Field	Description
7 Reserved	This field is reserved.
6 AGC_FRZ_EN	AGC Freeze Enable 0 AGC operates continuously during reception. 1 Enable the AGC Freeze mode after SFD detection.
5 CONT_RSSI_EN	Continuous RSSI Averaging Enable 0 RSSI averaging stops 64 µs after preamble is detected (for LQI), or after the CCA or Energy Detect operation is completed, for Sequence C. 1 Enable continuous RSSI averaging.
4 LQI_RSSI_NOT_CORR	LQI Mode 0 LQI based on Correlation peaks. 1 LQI based on RSSI.
3 CCA3_AND_NOT_OR	Determines the way CCA3 is required to be detected. 0 CCA1 or CCA2. 1 CCA1 and CCA2.
2 OWER_COMP_EN_LQI	Enable offset compensation and slope correction to averaged RSSI value during LQI computation, when LQI_RSSI_NOT_CORR=1.
1 OWER_COMP_EN_ED	Enable offset compensation and slope correction to averaged RSSI value during Energy Detect mode.
0 OWER_COMP_EN_CCA1	Enable offset compensation and slope correction to averaged RSSI value during CCA1 mode.

9.3.19 Clear Channel Assessment 2 Threshold Peak Compare (Indirect_Modem_CCA2_CORR_PEAKS)

Address: 0h base + 26h offset = 26h

Bit	7	6	5	4	3	2	1	0
Read	Reserved	CCA2_MIN_NUM_CORR_TH			CCA2_NUM_CORR_PEAKS			
Write								
Reset	0	1	1	0	0	0	0	0

Indirect_Modem_CCA2_CORR_PEAKS field descriptions

Field	Description
7 Reserved	This field is reserved.

Table continues on the next page...

Indirect_Modem_CCA2_CORR_PEAKS field descriptions (continued)

Field	Description
6–4 CCA2_MIN_ NUM_CORR_TH	Programmable threshold to be compared against number of correlation peaks that exceeded cca2_corr_thresh for detecting CCA mode 2. Number of peaks detected = cca2_min_num_corr_th + 1; Example: If it is programmed to 3, CCA2 logic looks for at least 4 correlation peaks that crossed the threshold, to indicate channel is idle or busy.
CCA2_NUM_ CORR_PEAKS	Counts of number of peaks that crossed cca2_corr_thresh. Read-only.

9.3.20 Clear Channel Assessment 2 Threshold (Indirect_Modem_CCA2_THRESH)

Address: 0h base + 27h offset = 27h

Bit	7	6	5	4	3	2	1	0
Read Write	CCA2_THRESH							
Reset	1	0	0	0	0	0	1	0

Indirect_Modem_CCA2_THRESH field descriptions

Field	Description
CCA2_THRESH	Clear Channel Assessment 2 Threshold Programmable energy threshold register to detect CCA mode 2.

9.3.21 TMR PRESCALE (Indirect_Modem_TMR_PRESCALE)

Address: 0h base + 28h offset = 28h

Bit	7	6	5	4	3	2	1	0
Read Write	Reserved					TMR_PRESCALE		
Reset	0	0	0	0	0	0	1	1

Indirect_Modem_TMR_PRESCALE field descriptions

Field	Description
7–3 Reserved	Reserved This field is reserved.
TMR_ PRESCALE	Timer Prescaler Establishes the Event Timer lock rate(Maximum timer duration). 000 Reserved 001 Reserved 010 500 kHz(33.55 S)

Table continues on the next page...

Indirect_Modem_TMR_PRESCALE field descriptions (continued)

Field	Description
	011 250 kHZ(67.11 S)-default 100 125 kHZ(134.22 S) 101 62.5 kHZ(268.44 S) 110 31.25 kHZ(536.87 S) 111 15.625 kHZ(1073.74 S)

9.3.22 GPIO Data (Indirect_Modem_GPIO_DATA)

Address: 0h base + 2Ah offset = 2Ah

Bit	7	6	5	4	3	2	1	0
Read Write	GPIO_ DATA8	GPIO_ DATA7	GPIO_ DATA6	GPIO_ DATA5	GPIO_ DATA4	GPIO_ DATA3	GPIO_ DATA2	GPIO_ DATA1
Reset	0	0	0	0	0	0	0	0

Indirect_Modem_GPIO_DATA field descriptions

Field	Description
7 GPIO_DATA8	GPIO Data 8 Write data to this register determines the state of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}, when the corresponding GPIO_DIR bit for that pin is set to 1. Data read from this register returns the state of the GPIO pins, when the corresponding GPIO_DIR bit for that pin is set to 0
6 GPIO_DATA7	GPIO Data 7 Write data to this register determines the state of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}, when the corresponding GPIO_DIR bit for that pin is set to 1. Data read from this register returns the state of the GPIO pins, when the corresponding GPIO_DIR bit for that pin is set to 0
5 GPIO_DATA6	GPIO Data 6 Write data to this register determines the state of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}, when the corresponding GPIO_DIR bit for that pin is set to 1. Data read from this register returns the state of the GPIO pins, when the corresponding GPIO_DIR bit for that pin is set to 0
4 GPIO_DATA5	GPIO Data 5 Write data to this register determines the state of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}, when the corresponding GPIO_DIR bit for that pin is set to 1. Data read from this register returns the state of the GPIO pins, when the corresponding GPIO_DIR bit for that pin is set to 0
3 GPIO_DATA4	GPIO Data 4 Write data to this register determines the state of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}, when the corresponding GPIO_DIR bit for that pin is set to 1. Data read from this register returns the state of the GPIO pins, when the corresponding GPIO_DIR bit for that pin is set to 0
2 GPIO_DATA3	GPIO Data 3 Write data to this register determines the state of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}, when the corresponding GPIO_DIR bit for that pin is set to 1. Data read from this register returns the state of the GPIO pins, when the corresponding GPIO_DIR bit for that pin is set to 0

Table continues on the next page...

Indirect_Modem_GPIO_DATA field descriptions (continued)

Field	Description
1 GPIO_DATA2	GPIO Data 2 Write data to this register determines the state of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}, when the corresponding GPIO_DIR bit for that pin is set to 1. Data read from this register returns the state of the GPIO pins, when the corresponding GPIO_DIR bit for that pin is set to 0
0 GPIO_DATA1	GPIO Data 1 Write data to this register determines the state of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}, when the corresponding GPIO_DIR bit for that pin is set to 1. Data read from this register returns the state of the GPIO pins, when the corresponding GPIO_DIR bit for that pin is set to 0

9.3.23 GPIO Direction Control (Indirect_Modem_GPIO_DIR)

Address: 0h base + 2Bh offset = 2Bh

Bit	7	6	5	4	3	2	1	0
Read Write	GPIO_DIR8	GPIO_DIR7	GPIO_DIR6	GPIO_DIR5	GPIO_DIR4	GPIO_DIR3	GPIO_DIR2	GPIO_DIR1
Reset	0	0	0	0	0	0	0	0

Indirect_Modem_GPIO_DIR field descriptions

Field	Description
7 GPIO_DIR8	GPIO Direction Control 8 Determines the direction for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 Pin is in input mode. 1 Pin is in output mode.
6 GPIO_DIR7	GPIO Direction Control 7 Determines the direction for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 Pin is in input mode. 1 Pin is in output mode.
5 GPIO_DIR6	GPIO Direction Control 6 Determines the direction for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 Pin is in input mode. 1 Pin is in output mode.
4 GPIO_DIR5	GPIO Direction Control 5 Determines the direction for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 Pin is in input mode. 1 Pin is in output mode.
3 GPIO_DIR4	GPIO Direction Control 4

Table continues on the next page...

Indirect_Modem_GPIO_DIR field descriptions (continued)

Field	Description
	Determines the direction for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 Pin is in input mode. 1 Pin is in output mode.
2 GPIO_DIR3	GPIO Direction Control 3 Determines the direction for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 Pin is in input mode. 1 Pin is in output mode.
1 GPIO_DIR2	GPIO Direction Control 2 Determines the direction for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 Pin is in input mode. 1 Pin is in output mode.
0 GPIO_DIR1	GPIO Direction Control 1 Write data to this register determines the state of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}, when the corresponding GPIO_DIR bit for that pin is set to 1. Data read from this register returns the state of the GPIO pins, when the corresponding GPIO_DIR bit for that pin is set to 0 0 Pin is in input mode. 1 Pin is in output mode.

9.3.24 GPIO Pullup Enable (Indirect_Modem_GPIO_PUL_EN)

Address: 0h base + 2Ch offset = 2Ch

Bit	7	6	5	4
Read Write	GPIO_PUL_EN8	GPIO_PUL_EN7	GPIO_PUL_EN6	GPIO_PUL_EN5
Reset	1	1	1	1

Bit	3	2	1	0
Read Write	GPIO_PUL_EN14	GPIO_PUL_EN3	GPIO_PUL_EN2	GPIO_PUL_EN1
Reset	1	1	1	1

Indirect_Modem_GPIO_PUL_EN field descriptions

Field	Description
7 GPIO_PUL_EN8	GPIO Pullup Enable 8 Determines whether a pull device (pullup or pulldown) is engaged for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 No pullup or pulldown is engaged on the pin. 1 Pullup or pulldown is engaged (see GPIO_PUL_SEL register) on the pin

Table continues on the next page...

Indirect_Modem_GPIO_PUL_EN field descriptions (continued)

Field	Description
6 GPIO_PUL_EN7	GPIO Pullup Enable 7 Determines whether a pull device (pullup or pulldown) is engaged for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 No pullup or pulldown is engaged on the pin. 1 Pullup or pulldown is engaged (see GPIO_PUL_SEL register) on the pin
5 GPIO_PUL_EN6	GPIO Pullup Enable 6 Determines whether a pull device (pullup or pulldown) is engaged for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 No pullup or pulldown is engaged on the pin. 1 Pullup or pulldown is engaged (see GPIO_PUL_SEL register) on the pin
4 GPIO_PUL_EN5	GPIO Pullup Enable 5 Determines whether a pull device (pullup or pulldown) is engaged for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 No pullup or pulldown is engaged on the pin. 1 Pullup or pulldown is engaged (see GPIO_PUL_SEL register) on the pin
3 GPIO_PUL_ EN14	GPIO Pullup Enable 4 Determines whether a pull device (pullup or pulldown) is engaged for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 No pullup or pulldown is engaged on the pin. 1 Pullup or pulldown is engaged (see GPIO_PUL_SEL register) on the pin
2 GPIO_PUL_EN3	GPIO Pullup Enable 3 Determines whether a pull device (pullup or pulldown) is engaged for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 No pullup or pulldown is engaged on the pin. 1 Pullup or pulldown is engaged (see GPIO_PUL_SEL register) on the pin
1 GPIO_PUL_EN2	GPIO Pullup Enable 2 Determines whether a pull device (pullup or pulldown) is engaged for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 No pullup or pulldown is engaged on the pin. 1 Pullup or pulldown is engaged (see GPIO_PUL_SEL register) on the pin
0 GPIO_PUL_EN1	GPIO Pullup Enable 1 Determines whether a pull device (pullup or pulldown) is engaged for each of the GPIO pins {GPIO8, GPIO7, ... , GPIO1}. 0 No pullup or pulldown is engaged on the pin. 1 Pullup or pulldown is engaged (see GPIO_PUL_SEL register) on the pin

9.3.25 GPIO Pullup Select (Indirect_Modem_GPIO_SEL)

Address: 0h base + 2Dh offset = 2Dh

Bit	7	6	5	4	3	2	1	0
Read Write	GPIO_PUL_SEL8	GPIO_PUL_SEL7	GPIO_PUL_SEL6	GPIO_PUL_SEL5	GPIO_PUL_SEL4	GPIO_PUL_SEL3	GPIO_PUL_SEL2	GPIO_PUL_SEL1
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_GPIO_SEL field descriptions

Field	Description
7 GPIO_PUL_SEL8	GPIO Pullup Select 8 For each pin {GPIO8, GPIO7, ... , GPIO1}, if GPIO_PUL[x]=1, this bit determines whether a pullup or pulldown is engaged. 0 Pulldown is engaged on the pin if GPIO_PUL_EN[x]=1. 1 Pullup is engaged on the pin if GPIO_PUL_EN[x]=1.
6 GPIO_PUL_SEL7	GPIO Pullup Select 7 For each pin {GPIO8, GPIO7, ... , GPIO1}, if GPIO_PUL[x]=1, this bit determines whether a pullup or pulldown is engaged. 0 Pulldown is engaged on the pin if GPIO_PUL_EN[x]=1. 1 Pullup is engaged on the pin if GPIO_PUL_EN[x]=1.
5 GPIO_PUL_SEL6	GPIO Pullup Select 6 For each pin {GPIO8, GPIO7, ... , GPIO1}, if GPIO_PUL[x]=1, this bit determines whether a pullup or pulldown is engaged. 0 Pulldown is engaged on the pin if GPIO_PUL_EN[x]=1. 1 Pullup is engaged on the pin if GPIO_PUL_EN[x]=1.
4 GPIO_PUL_SEL5	GPIO Pullup Select 5 For each pin {GPIO8, GPIO7, ... , GPIO1}, if GPIO_PUL[x]=1, this bit determines whether a pullup or pulldown is engaged. 0 Pulldown is engaged on the pin if GPIO_PUL_EN[x]=1. 1 Pullup is engaged on the pin if GPIO_PUL_EN[x]=1.
3 GPIO_PUL_SEL4	GPIO Pullup Select 4 For each pin {GPIO8, GPIO7, ... , GPIO1}, if GPIO_PUL[x]=1, this bit determines whether a pullup or pulldown is engaged. 0 Pulldown is engaged on the pin if GPIO_PUL_EN[x]=1. 1 Pullup is engaged on the pin if GPIO_PUL_EN[x]=1.
2 GPIO_PUL_SEL3	GPIO Pullup Select 3 For each pin {GPIO8, GPIO7, ... , GPIO1}, if GPIO_PUL[x]=1, this bit determines whether a pullup or pulldown is engaged.

Table continues on the next page...

Indirect_Modem_GPIO_SEL field descriptions (continued)

Field	Description
	0 Pulldown is engaged on the pin if GPIO_PUL_EN[x]=1. 1 Pullup is engaged on the pin if GPIO_PUL_EN[x]=1.
1 GPIO_PUL_SEL2	GPIO Pullup Select 2 For each pin {GPIO8, GPIO7, ... , GPIO1}, if GPIO_PUL[x]=1, this bit determines whether a pullup or pulldown is engaged. 0 Pulldown is engaged on the pin if GPIO_PUL_EN[x]=1. 1 Pullup is engaged on the pin if GPIO_PUL_EN[x]=1.
0 GPIO_PUL_SEL1	GPIO Pullup Select 1 For each pin {GPIO8, GPIO7, ... , GPIO1}, if GPIO_PUL[x]=1, this bit determines whether a pullup or pulldown is engaged. 0 Pulldown is engaged on the pin if GPIO_PUL_EN[x]=1. 1 Pullup is engaged on the pin if GPIO_PUL_EN[x]=1.

9.3.26 GPIO Drive Strength (Indirect_Modem_GPIO_DS)

Address: 0h base + 2Eh offset = 2Eh

Bit	7	6	5	4	3	2	1	0
Read Write	GPIO_DS8	GPIO_DS7	GPIO_DS6	GPIO_DS5	GPIO_DS4	GPIO_DS3	GPIO_DS2	GPIO_DS1
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_GPIO_DS field descriptions

Field	Description
7 GPIO_DS8	GPIO Drive Strength 8 For each pin {GPIO8, GPIO7, ... , GPIO1}: 0 Pin uses normal drive strength. 1 Pin uses high drive strength.
6 GPIO_DS7	GPIO Drive Strength 7 For each pin {GPIO8, GPIO7, ... , GPIO1}: 0 Pin uses normal drive strength. 1 Pin uses high drive strength.
5 GPIO_DS6	GPIO Drive Strength 6 For each pin {GPIO8, GPIO7, ... , GPIO1}: 0 Pin uses normal drive strength. 1 Pin uses high drive strength.
4 GPIO_DS5	GPIO Drive Strength 5

Table continues on the next page...

Indirect_Modem_GPIO_DS field descriptions (continued)

Field	Description
	For each pin {GPIO8, GPIO7, ... , GPIO1}: 0 Pin uses normal drive strength. 1 Pin uses high drive strength.
3 GPIO_DS4	GPIO Drive Strength 4 For each pin {GPIO8, GPIO7, ... , GPIO1}: 0 Pin uses normal drive strength. 1 Pin uses high drive strength.
2 GPIO_DS3	GPIO Drive Strength 3 For each pin {GPIO8, GPIO7, ... , GPIO1}: 0 Pin uses normal drive strength. 1 Pin uses high drive strength.
1 GPIO_DS2	GPIO Drive Strength 2 For each pin {GPIO8, GPIO7, ... , GPIO1}: 0 Pin uses normal drive strength. 1 Pin uses high drive strength.
0 GPIO_DS1	GPIO Drive Strength 1 For each pin {GPIO8, GPIO7, ... , GPIO1}: 0 Pin uses normal drive strength. 1 Pin uses high drive strength.

9.3.27 Antenna Control (Indirect_Modem_ANT_PAD_CTRL)

Address: 0h base + 30h offset = 30h

Bit	7	6	5	4	3	2	1	0
Read Write	ANTX_ POL3	ANTX_ POL2	ANTX_ POL1	ANTX_ POL0	ANTX_ CTRLMODE	ANTX_HZ	ANTX_EN	
Reset	0	0	0	0	1	0	0	0

Indirect_Modem_ANT_PAD_CTRL field descriptions

Field	Description
7 ANTX_POL3	Antenna Polarity Control 3 Control the polarity of the Antenna or Switch control pins. 0 Do not invert the RXSWITCH output. 1 Invert the RXSWITCH output.

Table continues on the next page...

Indirect_Modem_ANT_PAD_CTRL field descriptions (continued)

Field	Description
6 ANTX_POL2	Antenna Polarity Control 2 Control the polarity of the Antenna or Switch control pins. 0 Do not invert the TXSWITCH output. 1 Invert the TXSWITCH output.
5 ANTX_POL1	Antenna Polarity Control 1 Control the polarity of the Antenna or Switch control pins. 0 Do not invert the ANT_B output. 1 Invert the ANT_B output.
4 ANTX_POL0	Antenna Polarity Control 0 Control the polarity of the Antenna or Switch control pins. 0 Do not invert the ANT_A output. 1 Invert the ANT_A output.
3 ANTX_ CTRLMODE	Antenna Control Mode Sets the single/dual mode. 0 Single mode: ANT_A=ANTX TX_SWITCH=TXON and RX_SWITCH=(RXON OR TXON). 1 Dual mode: ANT_A=ANTX AND (RXON OR TXON) ANT_B=NOT (ANTX) AND (RXON OR TXON) TX_SWITCH=TXON and RX_SWITCH=RXON.
2 ANTX_HZ	Antenna High Impedance Antenna controls high impedance. 0 ANT_A, ANT_B, RX_SWITCH, and TX_SWITCH are actively driven outputs. 1 Set ANT_A, ANT_B, RX_SWITCH and TX_SWITCH in high impedance.
ANTX_EN	Antenna Controls Enable Enable ANT_A, ANT_B, RX_SWITCH, and TX_SWITCH. 00 All disabled. 01 Only RX_SWITCH/TX_SWITCH enabled. 10 Only ANT_A/ANT_B enabled. 11 All enabled.

9.3.28 Miscellaneous Pad Control (Indirect_Modem_MISC_PAD_CTRL)

Address: 0h base + 31h offset = 31h

Bit	7	6	5	4
Read Write	Reserved			
Reset	0	0	0	1

Bit	3	2	1	0
Read Write	MISO_HIZ_EN	IRQ_B_OD	NON_GPIO_DS	ANTX_CURR
Reset	1	0	1	0

Indirect_Modem_MISC_PAD_CTRL field descriptions

Field	Description
7–4 Reserved	This field is reserved.
3 MISO_HIZ_EN	Determines the output state of the radio's R_MISO output when R_SSEL_B is deasserted (high). 0 R_MISO output is driven low by the SPI slave. 1 R_MISO output is tristated (default). An external pullup on R_MISO must be employed if MISO_HIZ_EN=1.
2 IRQ_B_OD	IRQ_B Open Drain 0 IRQ_B is an actively-driven output. 1 IRQ_B is open-drain (external pullup required).
1 NON_GPIO_DS	Determines the drive strength for the all of the following pins: {IRQ_B, R_MISO, DTM0 ... DTM6}. 0 All the listed pins use normal drive strength. 1 All the listed pins use high drive strength.
0 ANTX_CURR	Determines the drive strength for the all of the following pins: {ANT_A, ANT_B, RX_SWITCH, TX_SWITCH}. 0 All the listed pins use normal drive strength. 1 All the listed pins use high drive strength.

9.3.29 RX_BYTE_COUNT (Indirect_Modem_RX_BYTE_COUNT)

Address: 0h base + 35h offset = 35h

Bit	7	6	5	4	3	2	1	0
Read	RX_BYTE_COUNT							
Write								
Reset	0	0	0	0	0	0	0	0

Indirect_Modem_RX_BYTE_COUNT field descriptions

Field	Description
RX_BYTE_ COUNT	During packet reception, this read-only register is a real-time indicator of the number of bytes that have been received. This register will read 0 until SFD and PHR have been received. It will read 1 after the first byte of Frame Control Field has been received, etc.

9.3.30 RX_WTR_MARK (Indirect_Modem_RX_WTR_MARK)

Address: 0h base + 36h offset = 36h

Bit	7	6	5	4	3	2	1	0
Read Write	RX_WTR_MARK							
Reset	1	1	1	1	1	1	1	1

Indirect_Modem_RX_WTR_MARK field descriptions

Field	Description
RX_WTR_MARK	Receive byte count (octets) needed to trigger a RXWTRMRKIRQ interrupt . A setting of 0 generates an interrupt at end of the Frame Length field (first byte after SFD). A setting of 1 generates an interrupt after the first byte of Frame Control Field, etc.

9.3.31 TXDELAY (Indirect_Modem_TXDELAY)

Address: 0h base + 38h offset = 38h

Bit	7	6	5	4	3	2	1	0
Read Write	Reserved		TXDELAY					
Reset	0	0	0	0	0	0	0	0

Indirect_Modem_TXDELAY field descriptions

Field	Description
7–6 Reserved	This field is reserved. Reserved
TXDELAY	Provides a fine-tune adjustment of the time delay between post-CCA Rx warm-down and the beginning of Tx warm-up for an Tx(non-Ack) packet. TXDELAY register will apply in both SLOTTED and UNSLOTTED modes, but only to T sequences (e.g., T, TR, and T(R)), not TxAck operations. This is a two's complement value. The minimum permissible value is -19(0x2D). Values less than -19 will lead to unexpected results. Resolution = 2µs. Range = +/- 62µs. Max TXDELAY = 0x1F Min TXDELAY = 0x2D

9.3.32 ACKDELAY (Indirect_Modem_ACKDELAY)

Address: 0h base + 39h offset = 39h

Bit	7	6	5	4	3	2	1	0
Read Write	Reserved		ACKDELAY					
Reset	0	0	1	1	1	1	0	1

Indirect_Modem_ACKDELAY field descriptions

Field	Description
7–6 Reserved	This field is reserved. Reserved
ACKDELAY	Provides a fine-tune adjustment of the time delay between Rx warmdown and the beginning of Tx warmup for an Tx Acknowledge packet. ACKDELAY register will apply to both SLOTTED and UNSLOTTED TxAck, but only to TxAck (not T sequences). This is a two's complement value. The minimum permissible value is -19 (0x2D). Values less than -19 will lead to unexpected results. Values less than -19 will lead to unexpected results. Resolution = 2µs. Range = +/- 62µs. Max ACKDELAY = 0x1F Min ACKDELAY = 0x2D

9.3.33 Antenna AGC and FAD Control (Indirect_Modem_ANT_AGC_CTRL)

Address: 0h base + 51h offset = 51h

Bit	7	6	5	4	3	2	1	0
Read Write	SNF_EN	AGC_EN	AGC_LEVEL		Reserved		ANTX	FAD_EN
Reset	0	1	0	0	0	0	0	0

Indirect_Modem_ANT_AGC_CTRL field descriptions

Field	Description
7 SNF_EN	Sniff Enable Enable the sniff function that allows the symbol demodulator to be started when the RSSI level has crossed the defined SNF threshold.
6 AGC_EN	AGC Enable Enable the AGC function. 0 The AGC function is disabled. 1 The AGC function is enabled.
5–4 AGC_LEVEL	AGC Level If AGC_EN=0, forces the attenuation mode AGC_LEVEL_IN[1:0]. For reads:

Table continues on the next page...

Indirect_Modem_ANT_AGC_CTRL field descriptions (continued)

Field	Description
	AGC_EN=1: gives the AGC level chosen. AGC_EN=0: reads back what was last written to the register. 00 FE full gain, CHF full gain. 01 FE full gain, CHF low gain. 10 Reserved 11 FE low gain, CHF low gain.
3–2 Reserved	This field is reserved.
1 ANTX	ANTX bit selects the initial antenna state when FAD_EN=1, or allows software direct control over the antenna selection at all times when FAD_EN=0. 0 Select ANT_B 1 Select ANT_A
0 FAD_EN	FAD Enable Enable the FAD function. 0 FAD function is disabled. 1 FAD function is enabled.

9.3.34 LPPS_CTRL (Indirect_Modem_LPPS_CTRL)

Address: 0h base + 56h offset = 56h

Bit	7	6	5	4
Read Write	Reserved			LPPS_BUFMIX_EN
Reset	0	0	0	1

Bit	3	2	1	0
Read Write	LPPS_LIM_EN	LPPS_RSSI_EN	LPPS_LNA_EN	LPPS_EN
Reset	1	1	1	0

Indirect_Modem_LPPS_CTRL field descriptions

Field	Description
7–5 Reserved	This field is reserved.
4 LPPS_BUFMIX_EN	VCO buffer enable during LPPS 0 Normal operation. 1 During preamble search in LPPS mode, the VCO buffer is turned on/off with a 50% DC (Duty Cycle).
3 LPPS_LIM_EN	LIM enable during LPPS 0 Normal operation. 1 During preamble search in LPPS mode, the limiter is turned on/off with a 50% DC (Duty Cycle).

Table continues on the next page...

Indirect_Modem_LPPS_CTRL field descriptions (continued)

Field	Description
2 LPPS_RSSI_EN	RSSI enable during LPPS 0 Normal operation. 1 During preamble search in LPPS mode, the RSSI is turned on/off with a 50% DC (Duty Cycle).
1 LPPS_LNA_EN	LNA enable during LPPS 0 Normal operation. 1 During preamble search in LPPS mode, the LNA is turned on/off with a 50% DC (Duty Cycle).
0 LPPS_EN	Master Enable for Low-Power Preamble Search (LPPS) mode 0 Normal operation. 1 Enable Low-Power Preamble Search mode. Note: LPPS mode should not be engaged if Fast Antenna Diversity mode is in use (FAN_EN=1).

9.3.35 RSSI (Indirect_Modem_RSSI)

Address: 0h base + 5Bh offset = 5Bh

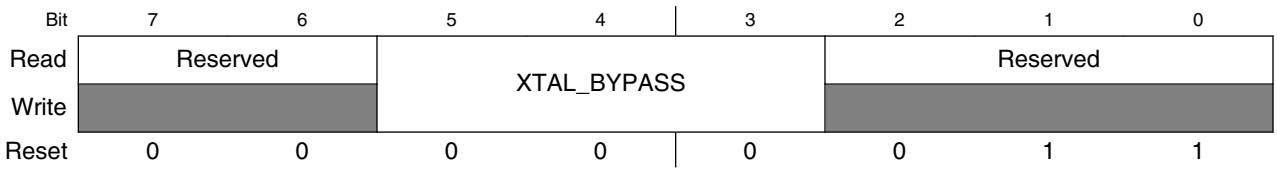
Bit	7	6	5	4	3	2	1	0
Read	RSSI_ Level_7	RSSI_ Level_6	RSSI_ Level_5	RSSI_ Level_4	RSSI_ Level_3	RSSI_ Level_2	RSSI_ Level_1	RSSI_ Level_0
Write								
Reset	0	0	0	0	0	0	0	0

Indirect_Modem_RSSI field descriptions

Field	Description
7 RSSI_Level_7	RSSI Level, refreshed every 125 µs.
6 RSSI_Level_6	RSSI Level, refreshed every 125 µs.
5 RSSI_Level_5	RSSI Level, refreshed every 125 µs.
4 RSSI_Level_4	RSSI Level, refreshed every 125 µs.
3 RSSI_Level_3	RSSI Level, refreshed every 125 µs.
2 RSSI_Level_2	RSSI Level, refreshed every 125 µs.
1 RSSI_Level_1	RSSI Level, refreshed every 125 µs.
0 RSSI_Level_0	RSSI Level, refreshed every 125 µs.

9.3.36 XTAL Control (Indirect_Modem_XTAL_CTRL)

Address: 0h base + 6Eh offset = 6Eh



Indirect_Modem_XTAL_CTRL field descriptions

Field	Description
7–6 Reserved	This field is reserved.
5–3 XTAL_BYPASS	XTAL Bypass Reset value = 000 Bypass value = 111 XTAL32m bypass for external injection. Note: XTALEN must be 0 when XTAL_BYPASS=1. See PWR_MODES register for XTALEN.
Reserved	This field is reserved.

Appendix A
Release Notes for Revision 3

A.1 General changes throughout

- Updated missing reset values of Modem and Indirect Modem registers.
- Added the following registers to the Indirect Modem memory map: TMR_PRESCALE, RX_BYTE_COUNT, RX_WTR_MARK, TXDELAY, ACKDELAY.
- Updated register bit map structure of XTAL Control (Indirect Modem_XTAL_CTRL) register.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number MCR20ARM
Revision 3, July 2016