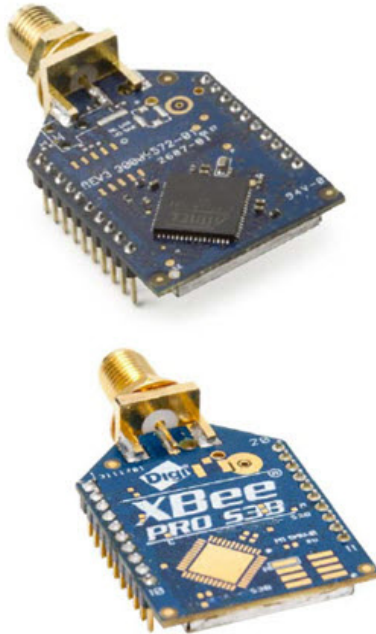


XBee-PRO® 900HP/XBee-PRO® XSC RF Modules

XBee-PRO RF Modules by Digi International

Models: XBEE-PRO S3, XBEE-PRO S3B

Hardware: S3, S3B



Digi International Inc.
11001 Bren Road East
Minnetonka, MN 55343
877 912-3444 or 952 912-3444
<http://www.digi.com>

90002173_B
September 26,

© 2012 Digi International, Inc. All rights reserved

No part of the contents of this manual may be transmitted or reproduced in any form or by any means without the written permission of Digi International, Inc.

XBee-PRO® is a registered trademark of Digi International, Inc.

| | | |
|--------------------|-----------------|--|
| Technical Support: | Phone: | (866) 765-9885 toll-free U.S.A. & Canada (801) 765-9885 Worldwide 8:00 am - 5:00 pm [U.S. Mountain Time] |
| | Online Support: | http://www.digi.com/support/eservice/login.jsp |
| | Email: | rf-experts@digi.com |

Contents

1. Preface: How to use this manual 5

2. RF Module Hardware 6

XBee-PRO S3B Hardware Description 6
Worldwide Acceptance 6

Specifications 7

Serial Communications Specifications 8

UART 8
SPI 8

GPIO Specifications 8

Hardware Specs for Programmable Variant 9

Mechanical Drawings 10

Pin Signals 11

Design Notes 11

Power Supply Design 11
Recommended Pin Connections 11
Board Layout 12

Module Operation for Programmable Variant 12

XBee Programmable Bootloader 14

Overview 14
Bootloader Software Specifics 14
Bootloader Menu Commands 18
Firmware Updates 19
Output File Configuration 19

3. RF Module Operation 21

Basic Operational Design 21

Serial Communications 21

UART Data Flow 21
SPI Communications 22
SPI Operation 23
Configuration 24
Data Format 25
SPI Parameters 25
Serial Buffers 26
UART Flow Control 26
Serial Interface Protocols 27

Modes of Operation 28

Description of Modes 28
Transmit Mode 28
Receive Mode 30
Command Mode 30
Sleep Mode 31

4. Networking Methods 32

MAC/PHY Basics 32

Related parameters: CM, HP, ID, PL, RR, MT 32

Addressing Basics 32

Related parameters: SH, SL, DH, DL, TO 32

Point to Point/Multipoint (P2MP) 33

Throughput 33

Repeater/Directed Broadcast 33

Related parameters: CE, NH, NN, BH 33

DigiMesh Networking 34

Related Command: MR 34
DigiMesh Feature Set 34
Data Transmission and Routing 34
Transmission Timeouts 35

5. Sleep Mode 37

Sleep Modes 37

Normal Mode (SM=0) 37
Asynchronous Pin Sleep Mode (SM=1) 37
Asynchronous Cyclic Sleep Mode (SM=4) 37
Asynchronous Cyclic Sleep with Pin Wake Up Mode (SM=5) 38
Synchronous Sleep Support Mode (SM=7) 38
Synchronous Cyclic Sleep Mode (SM=8) 38

Asynchronous Sleep Operation 38

Wake Timer 38

Indirect Messaging and Polling (P2MP Packets Only) 39

Indirect Messaging 39
Polling 39

Synchronous Sleep Operation (DigiMesh networks only) 39

Operation 39
Becoming a Sleep Coordinator 42
Configuration 43
Diagnostics 45

6. Command Reference Tables 46

7. API Operation 57

API Frame Format 57

API Serial Exchanges 59

AT Commands 59
Transmitting and Receiving RF Data 59
Remote AT Commands 59

Supporting the API 60

Frame Descriptions 60

AT Command 60
AT Command - Queue Parameter Value 61

Contents

| | |
|-------------------------------|----|
| TX Request | 61 |
| Explicit TX Request | 62 |
| Remote AT Command Request | 64 |
| AT Command Response | 65 |
| Modem Status | 65 |
| Transmit Status | 66 |
| Route Information Packet | 66 |
| Aggregate Addressing Update | 68 |
| RX Indicator | 69 |
| Explicit Rx Indicator | 70 |
| Node Identification Indicator | 70 |
| Remote Command Response | 72 |

8. Advanced Application Features 73

Remote Configuration Commands 73

| | |
|------------------------------------|----|
| Sending a Remote Command | 73 |
| Applying Changes on Remote Devices | 73 |
| Remote Command Responses | 73 |

Network Commissioning and Diagnostics 73

| | |
|--|----|
| Device Configuration | 73 |
| Network Link Establishment and Maintenance | 73 |
| Device Placement | 74 |
| Device Discovery | 75 |
| Link Reliability | 75 |
| Commissioning Pushbutton and Associate LED | 78 |

I/O Line Monitoring 80

| | |
|------------------------------|----|
| I/O Samples | 80 |
| Queried Sampling | 80 |
| Periodic I/O Sampling | 82 |
| Digital I/O Change Detection | 82 |

General Purpose Flash Memory 82

| | |
|--|----|
| Accessing General Purpose Flash Memory | 82 |
|--|----|

Over-the-Air Firmware Upgrades 88

| | |
|----------------------------------|----|
| Distributing the New Application | 89 |
| Verifying the New Application | 89 |
| Installing the Application | 89 |
| Things to Remember | 90 |

Appendix A: XSC Firmware 91

Specifications 93

Pin Signals 93

Electrical Characteristics 95

| | |
|-----------------------|----|
| Timing Specifications | 95 |
|-----------------------|----|

Mechanical Drawings 96

RF Module Operation 97

Serial Communications 97

| | |
|---------------------------|----|
| UART-Interfaced Data Flow | 97 |
| Serial Data | 97 |
| Flow Control | 98 |

Modes of Operation 99

| | |
|---------------|-----|
| Idle Mode | 99 |
| Transmit Mode | 99 |
| Sleep Mode | 101 |
| Command Mode | 103 |

RF Module Configuration 105

XBee Programming Examples 105

| | |
|-----------------|-----|
| AT Commands | 105 |
| Binary Commands | 106 |

Command Reference Table 106

Command Descriptions 107

RF Communication Modes 124

Addressing 125

| | |
|---------------------|-----|
| Address Recognition | 126 |
|---------------------|-----|

Basic Communications 126

| | |
|--------------------------|-----|
| Streaming Mode (Default) | 126 |
| Repeater Mode | 127 |

Acknowledged Communications 130

| | |
|-------------------|-----|
| Acknowledged Mode | 130 |
|-------------------|-----|

Appendix B: Agency Certifications for S3B Hardware 133

FCC (United States) Certification 133

| | |
|--------------------------|-----|
| Labeling Requirements | 133 |
| FCC Notices | 133 |
| Limited Modular Approval | 134 |
| FCC-approved Antennas | 134 |

IC (Industry Canada) Certification 134

Appendix C: Agency Certifications for Legacy S3/S3B Hardware 139

FCC (United States) Certification 139

| | |
|--------------------------|-----|
| Labeling Requirements | 139 |
| FCC Notices | 139 |
| Limited Modular Approval | 140 |
| FCC-approved Antennas | 140 |

IC (Industry Canada) Certification 140

Appendix D: Additional Information 145

Preface: How to use this manual

This combined manual contains documentation for two hardware platforms: the S3 and S3B. Existing S3 customers are strongly encouraged to migrate their systems and designs to the newer and superior S3B platform.

This manual also contains documentation for two RF protocols: XStream Compatible (XSC) and 900HP. The XSC firmware is provided for customers who need compatibility with existing networks which need to be 9XStream compatible. Customers which do not require this compatibility should not use the XSC firmware, but rather the newer 900HP firmware.

Documentation for the XSC firmware is contained in Appendix A. All other firmware documentation in the manual is not applicable to XSC firmware. Likewise documentation in Appendix A is not applicable to the 900HP firmware.

The following table describes how to use this manual based on the Digi part number for the module:

| Digi Part Numbers | FCC ID | Hardware Platform | Pre-installed Firmware | Firmware Available | Regulatory Information |
|--|--------------|-------------------|------------------------|--------------------|------------------------|
| XBP09-XC... | MCQ-XBEEEXSC | S3 | XSC | XSC | Appendix C |
| XBP9B-XC*T-001 (revision G and earlier) XBP9B-XC*T-002 (revision G and earlier) XBP9B-XC*T-021 (revision F and earlier) XBP9B-XC*T-022 (revision F and earlier) | MCQ-XBPS3B | S3B | XSC | XSC | Appendix C |
| XBP9B-XC*T-001 (revision H and later) XBP9B-XC*T-002 (revision H and later) XBP9B-XC*T-021 (revision G and later) XBP9B-XC*T-022 (revision G and later) all other part numbers beginning XBP9B-XC... | MCQ-XB900HP | S3B | XSC | XSC / 900HP | Appendix B |
| XBP9B-D... | MCQ-XB900HP | S3B | 900HP | XSC / 900HP | Appendix B |

1. RF Module Hardware

This manual describes the operation of the XBee-PRO® 900HP RF module, which consists of firmware loaded onto XBee-PRO S3B hardware.

XBee-PRO 900HP embedded RF modules provide wireless connectivity to end-point devices in mesh networks. Utilizing the XBee-PRO Feature Set, these modules are interoperable with other devices. With the XBee, users can have their network up-and-running in a matter of minutes without configuration or additional development.

XBee-PRO S3B Hardware Description

The XBee-PRO S3B radio module hardware consists of an Energy Micro EFM32G230F128 microcontroller, an Analog Devices ADF7023 radio transceiver, an RF power amplifier, and in the programmable version, a Freescale MC9S08QE32 microcontroller.

Worldwide Acceptance

FCC Certified (USA) - Refer to Appendix B for FCC Requirements.

Systems that include XBee-PRO Modules inherit Digi's FCC Certification

ISM (Industrial, Scientific & Medical) frequency band

Manufactured under **ISO 9001:2000** registered standards

XBee-PRO® (900 MHz) RF Modules are approved for use in **US** and **Canada**.

RoHS compliant



Specifications

Specifications of the XBee-PRO® 900HP/XBee-PRO® XSC RF Module

| Specification | XBee |
|--|---|
| Performance | |
| * Indoor/Urban Range | 10kbps: up to 2000 ft (610m) 200kbps: up to 1000 ft (305m) |
| * Outdoor RF line-of-sight Range | 10kbps: up to 9 miles (15.5km) 200kbps: up to 4 miles (6.5km) (with 2.1dB dipole antennas) |
| Transmit Power Output | 24 dBm (250 mW) (software selectable) |
| RF Data Rate (High) | 200 kbps |
| RF Data Rate (Low) | 10 kbps |
| Serial UART interface | CMOS Serial UART, baud rate stability of <1% |
| Serial Interface Data Rate (software selectable) | 9600-230400 baud |
| Receiver Sensitivity (typical) | -101 dBm, high data rate, -110 dBm, low data rate |
| Power Requirements | |
| Supply Voltage | 2.1 to 3.6 VDC** **Supply voltages of less than 3.0V may result in reduced performance. Output power and receiver sensitivity may be degraded. |
| Transmit Current | PL=4 : 215mA typical, (290mA max) PL=3 : 160mA typical PL=2 : 120mA typical PL=1 : 95mA typical PL=0 : 60mA typical |
| Idle / Receive Current | 29mA typical at 3.3V, (35mA max) |
| Sleep Current | 2.5 μ A (typical) |
| General | |
| **Operating Frequency Band | 902 to 928 MHz (software selectable channels) |
| Dimensions | 1.297" x 0.962" x 0.215 (3.29cm x 2.44cm x 0.546cm) Note: Dimensions do not include connector/antenna or pin lengths |
| Weight | 5 to 8 grams, depending on the antenna option |
| Operating Temperature | -40° to 85° C (industrial) |
| Antenna Options | Integrated wire, U. FL RF connector, Reverse-polarity SMA connector |
| Digital I/O | 15 I/O lines, |
| ADC | 4 10-bit analog inputs |
| Networking & Security | |
| Supported Network Topologies | Mesh, point-to-point, point-to-multipoint, peer-to-peer |
| Number of Channels, user selectable channels | 64 channels available |
| Addressing Options | PAN ID, Preamble ID, and 64-bit addresses |
| Encryption | 128 bit AES |
| Agency Approvals | |

Specifications of the XBee-PRO® 900HP/XBee-PRO® XSC RF Module

| Specification | XBee |
|---------------------------------|---------------|
| United States (FCC Part 15.247) | MCQ-XB900HP |
| Industry Canada (IC) | 1846A-XB900HP |
| Australia | C-Tick |
| Brazil | Pending |

* To determine your range, perform a range test under your operating conditions.

Serial Communications Specifications

XBee RF modules support both UART (Universal Asynchronous Receiver / Transmitter) and SPI (Serial Peripheral Interface) serial connections.

UART

UART Pin Assignments

| UART Pins | Module Pin Number |
|--------------------------------|-------------------|
| DOUT | 2 |
| DIN / CONFIG | 3 |
| $\overline{\text{CTS}}$ / DIO7 | 12 |
| $\overline{\text{RTS}}$ / DIO6 | 16 |

More information on UART operation is found in the UART section in Chapter 2.

SPI

SPI Pin Assignments

| SPI Pins | Module Pin Number |
|------------------|-------------------|
| SPI_SCLK / DIO18 | 18 |
| SPI_SSEL / DIO17 | 17 |
| SPI_MOSI / DIO16 | 11 |
| SPI_MISO / DIO15 | 4 |
| SPI_ATTN / DIO1 | 19 |

For more information on SPI operation, see the SPI section in Chapter 2.

GPIO Specifications

XBee RF modules have 15 GPIO (General Purpose Input / Output) ports available. The exact list will depend on the module configuration, as some GPIO pins are used for purposes such as serial communication.

See GPIO section for more information on configuring and using GPIO ports.

Electrical Specifications for GPIO Pins

| GPIO Electrical Specification | Value |
|----------------------------------|--|
| Voltage - Supply | 2.1 - 3.6 V, (3.0V or higher required for optimal performance) |
| Low Schmitt switching threshold | 0.3 x Vdd |
| High Schmitt switching threshold | 0.7 x Vdd |
| Input pull-up resistor value | 40 k Ω |
| Input pull-down resistor value | 40 k Ω |
| Output voltage for logic 0 | 0.05 x Vdd |
| Output voltage for logic 1 | 0.95 x Vdd |

Electrical Specifications for GPIO Pins

| GPIO Electrical Specification | Value |
|--------------------------------------|-------|
| Output source current | 2 mA |
| Output sink current | 2 mA |
| Total output current (for GPIO pins) | 48 mA |

Hardware Specs for Programmable Variant

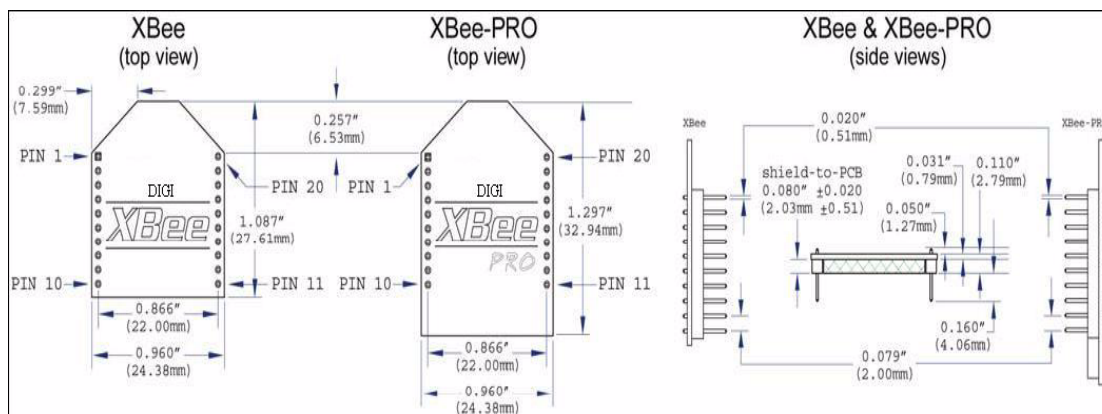
If the module has the programmable secondary processor, add the following table values to the specifications listed on page 7. For example, if the secondary processor is running at 20 MHz and the primary processor is in receive mode then the new current value will be $I_{\text{total}} = I_{r2} + I_{rx} = 14 \text{ mA} + 9 \text{ mA} = 23 \text{ mA}$, where I_{r2} is the runtime current of the secondary processor and I_{rx} is the receive current of the primary.

Specifications of the programmable secondary processor

| Optional Secondary Processor Specification | These numbers add to specifications (Add to RX, TX, and sleep currents depending on mode of operation) |
|---|--|
| Runtime current for 32k running at 20MHz | +14mA |
| Runtime current for 32k running at 1MHz | +1mA |
| Sleep current | +0.5µA typical |
| For additional specifications see Freescale Datasheet and Manual | MC9S08QE32 |
| Voltage requirement for secondary processor to operate at maximum clock frequency | 2.4 to 3.6VDC |
| Minimum Reset Pulse for Programmable | 100nS |
| Minimum Reset Pulse to Radio | 50nS |
| VREF Range | 1.8VDC to VCC |

Mechanical Drawings

Mechanical drawings of the XBee-PRO 900HP RF Modules (antenna options not shown). All dimensions are in inches.



Pin Signals

Pin Assignments for XBee Modules

(Low-asserted signals are distinguished with a horizontal line above signal name.)

| Pin # | Name | Direction | Default State | Description |
|-------|--------------------|-----------|---------------|---|
| 1 | VCC | | | Power Supply |
| 2 | DOUT/DIO13 | Both | Output | GPIO / UART Data out |
| 3 | DIN/nConfig/DIO14 | Both | Input | GPIO / UART Data In |
| 4 | DIO12/SPI_MISO | Both | Output | GPIO / SPI slave out |
| 5 | nRESET | Input | | Module Reset. Drive low to reset the module. This is also an output with an open drain configuration with an internal 20 K ohm pull-up (never drive to logic high, as the module may be driving it low). The minimum pulse width is 1 mS. |
| 6 | DIO10/PWM0 | Both | | GPIO / RX Signal Strength Indicator |
| 7 | DIO11/PWM1 | Both | | GPIO / Pulse Width Modulator |
| 8 | reserved | | Disabled | Do Not Connect |
| 9 | nDTR/SLEEP_RQ/DIO8 | Both | Input | GPIO / Pin Sleep Control Line (DTR on the dev board) |
| 10 | GND | | | Ground |
| 11 | DIO4/AD4/SPI_MOSI | Both | | GPIO/SPI slave In |
| 12 | nCTS/DIO7 | Both | Output | GPIO / Clear-to-Send Flow Control |
| 13 | On_nSLEEP/DIO9 | Output | Output | GPIO / Module Status Indicator |
| 14 | VREF | Input | | Internally used for programmable secondary processor. For compatibility with other XBee modules, we recommend connecting this pin to the voltage reference if Analog Sampling is desired. Otherwise, connect to GND. |
| 15 | Associate/DIO5 | Both | Output | GPIO / Associate Indicator |
| 16 | nRTS/DIO6 | Both | Input | GPIO / Request-to-Send Flow Control |
| 17 | AD3/DIO3/SPI_nSSEL | Both | | GPIO / Analog Input / SPI Slave Select |
| 18 | AD2/DIO2/SPI_CLK | Both | | GPIO / Analog Input / SPI Clock |
| 19 | AD1/DIO1/SPI_nATTN | Both | | GPIO / Analog Input / SPI Attention |
| 20 | AD0/DIO0 | Both | | GPIO / Analog Input |

- Signal Direction is specified with respect to the module
- See Design Notes section below for details on pin connections.

Design Notes

The XBee modules do not specifically require any external circuitry or specific connections for proper operation. However, there are some general design guidelines that are recommended for help in troubleshooting and building a robust design.

Power Supply Design

Poor power supply can lead to poor radio performance, especially if the supply voltage is not kept within tolerance or is excessively noisy. To help reduce noise, we recommend placing both a 1 μ F and 47pF capacitor as near to pin 1 on the PCB as possible. If using a switching regulator for your power supply, switching frequencies above 500kHz are preferred. Power supply ripple should be limited to a maximum 250mV peak to peak.

Note – For designs using the programmable modules, an additional 10 μ F decoupling cap is recommended near pin 1 of the module. The nearest proximity to pin 1 of the three caps should be in the following order: 47pF, 1 μ F followed by 10 μ F.

Recommended Pin Connections

The only required pin connections are VCC, GND, DOUT and DIN. To support serial firmware updates, VCC, GND, DOUT, DIN, RTS, and DTR should be connected.

All unused pins should be left disconnected. All inputs on the radio can be pulled high or low with 40k internal pull-up or pull-down resistors using the PR and PD software commands. No specific treatment is needed for unused outputs.

For applications that need to ensure the lowest sleep current, unconnected inputs should never be left floating. Use internal or external pull-up or pull-down resistors, or set the unused I/O lines to outputs.

Other pins may be connected to external circuitry for convenience of operation, including the Associate LED pin (pin 15) and the Commissioning pin (pin 20). An LED attached to the the associate LED pin will flash differently depending on the state of the module to the network, and a pushbutton attached to pin 20 can enable various join functions without having to send serial port commands. Please see the commissioning pushbutton and associate LED section in chapter 7 for more details. The source and sink capabilities are limited to 6mA on all I/O pins.

The VRef pin (pin 14) is only used on the programmable versions of these modules. For compatibility with other XBee modules, we recommend connecting this pin to a voltage reference if analog sampling is desired. Otherwise, connect to GND.

Board Layout

XBee modules are designed to be self sufficient and have minimal sensitivity to nearby processors, crystals or other PCB components. As with all PCB designs, Power and Ground traces should be thicker than signal traces and able to comfortably support the maximum current specifications. No other special PCB design considerations are required for integrating XBee radios except in the antenna section.

The choice of antenna and antenna location is very important for correct performance. XBees do not require additional ground planes on the host PCB. In general, antenna elements radiate perpendicular to the direction they point. Thus a vertical antenna emits across the horizon. Metal objects near the antenna cause reflections and may reduce the ability for an antenna to radiate efficiently. Metal objects between the transmitter and receiver can also block the radiation path or reduce the transmission distance, so external antennas should be positioned away from them as much as possible. Some objects that are often overlooked are metal poles, metal studs or beams in structures, concrete (it is usually reinforced with metal rods), metal enclosures, vehicles, elevators, ventilation ducts, refrigerators, microwave ovens, batteries, and tall electrolytic capacitors.

Module Operation for Programmable Variant

The modules with the programmable option have a secondary processor with 32k of flash and 2k of RAM. This allows module integrators to put custom code on the XBee module to fit their own unique needs. The DIN, DOUT, RTS, CTS, and RESET lines are intercepted by the secondary processor to allow it to be in control of the data transmitted and received. All other lines are in parallel and can be controlled by either the internal microcontroller or the MC9S08QE micro (see Block Diagram for details). The internal microcontroller by default has control of certain lines. These lines can be released by the internal microcontroller by sending the proper command(s) to disable the desired DIO line(s) (see XBee Command Reference Tables).

In order for the secondary processor to sample with ADCs, the XBee pin 14 (VREF) must be connected to a reference voltage.

Digi provides a bootloader that can take care of programming the processor over the air or through the serial interface. This means that over the air updates can be supported through an XMODEM protocol. The processor can also be programmed and debugged through a one wire interface BKGD (Pin 8).



XBee Programmable Bootloader

Overview

The XBee Programmable module is equipped with a Freescale MC9S08QE32 application processor. This application processor comes with a supplied bootloader. This section describes how to interface the customer's application code running on this processor to the XBee Programmable module's supplied bootloader.

The first section discusses how to initiate firmware updates using the supplied bootloader for wired and over-the-air updates.

Bootloader Software Specifics

Memory Layout

Figure 1 shows the memory map for the MC9S08QE32 application processor.

The supplied bootloader occupies the bottom pages of the flash from 0xF200 to 0xFFFF. Application code cannot write to this space.

The application code can exist in Flash from address 0x8400 to 0xF1BC. 1k of Flash from 0x8000 to 0x83FF is reserved for Non Volatile Application Data that will not be erased by the bootloader during a flash update.

A portion of RAM is accessible by both the application and the bootloader. Specifically, there is a shared data region used by both the application and the bootloader that is located at RAM address 0x200 to 0x215. Application code should not write anything to BLResetCause or AppResetCause unless informing the bootloader of the impending reset reason. The Application code should not clear BLResetCause unless it is handling the unexpected reset reason.

To prevent a malfunctioning application from running forever, the Bootloader increments BLResetCause after each watchdog or illegal instruction reset. If this register reaches above 0x10 the bootloader will stop running the application for a few minutes to allow an OTA or Local update to occur. If no update is initiated within the time period, BLResetCause is cleared and the application is started again. To prevent unexpected halting of the application, the application shall clear or decrement BLResetCause just before a pending reset. To disable this feature, the application shall clear BLResetCause at the start of the application.

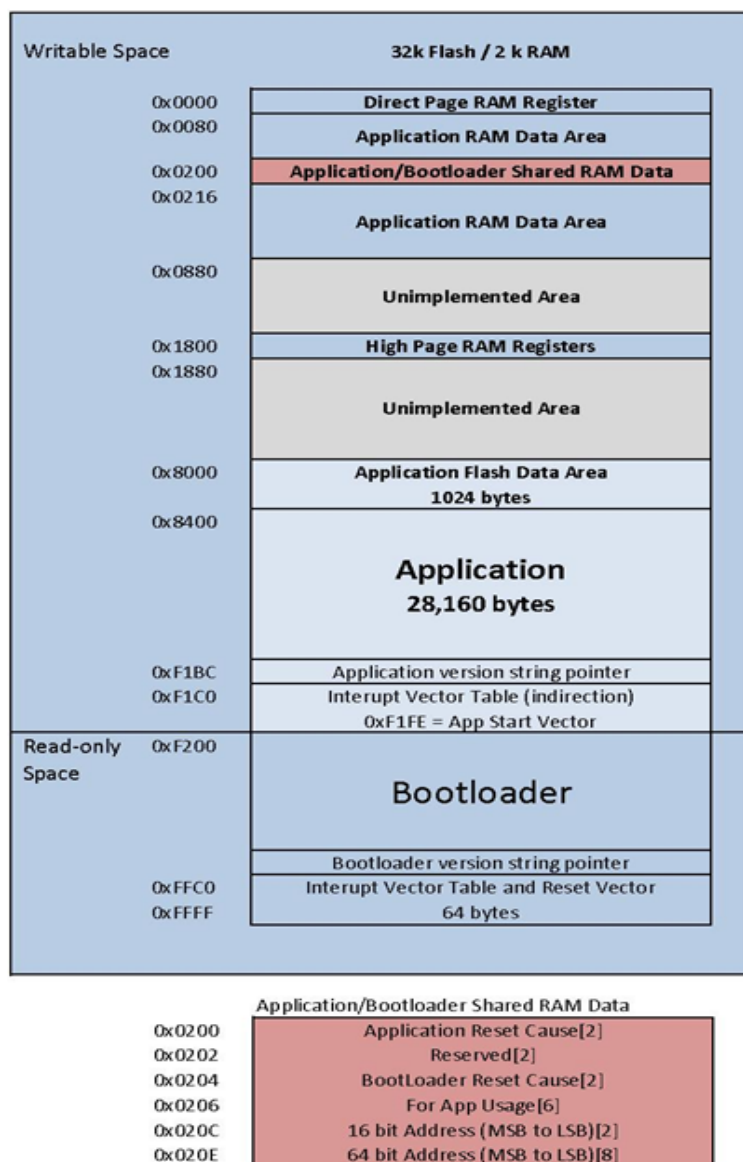


Figure 1: MC9S08QE32 Memory Map

Operation

Upon reset of any kind, the execution control begins with the bootloader.

If the reset cause is Power-On reset (POR), Pin reset (PIN), or Low Voltage Detect (LVD) reset (LVD) the bootloader will not jump to the application code if the override bits are set to RTS(D7)=1, DTR(D5)=0, and DIN(B0)=0. Otherwise, the bootloader writes the reset cause "NOTHING" to the shared data region, and jumps to the Application.

Reset causes are defined in the file *common.h* in an enumeration with the following definitions:

```
typedef enum {
    BL_CAUSE_NOTHING    = 0x0000, //PIN, LVD, POR
    BL_CAUSE_NOTHING_COUNT = 0x0001, //BL_Reset_Cause counter
    // Bootloader increments cause every reset
    BL_CAUSE_BAD_APP    = 0x0010, //Bootloader considers APP invalid
} BL_RESET_CAUSES;

typedef enum {
    APP_CAUSE_NOTHING      = 0x0000,
    APP_CAUSE_USE001       = 0x0001,
    // 0x0000 to 0x00FF are considered valid for APP use.
    APP_CAUSE_USE255       = 0x00FF,
    APP_CAUSE_FIRMWARE_UPDATE = 0x5981,
    APP_CAUSE_BYPASS_MODE    = 0x4682,
    APP_CAUSE_BOOTLOADER_MENU = 0x6A18,
} APP_RESET_CAUSES;
```

Otherwise, if the reset cause is a "watchdog" or other reset, the bootloader checks the shared memory region for the APP_RESET_CAUSE. If the reset cause is:

- 1."APP_CAUSE_NOTHING" or 0x0000 to 0x00FF, the bootloader increments the BL_RESET_CAUSES, verifies that it is still less than BL_CAUSE_BAD_APP, and jumps back to the application. If the Application does not clear the BL_RESET_CAUSE, it can prevent an infinite loop of running a bad application that continues to perform illegal instructions or watchdog resets.
- 2."APP_CAUSE_FIRMWARE_UPDATE", the bootloader has been instructed to update the application "over-the-air" from a specific 64-bit address. In this case, the bootloader will attempt to initiate an Xmodem transfer from the 64-bit address located in shared RAM.
- 3."APP_CAUSE_BYPASS_MODE", the bootloader executes bypass mode. This mode passes the local UART data directly to the internal microcontroller allowing for direct communication with the internal microcontroller.

The only way to exit bypass mode is to reset or power cycle the module.

If none of the above is true, the bootloader will enter "Command mode". In this mode, users can initiate firmware downloads both wired and over-the-air, check application/bootloader version strings, and enter Bypass mode.

Application version string

Figure 1 shows an "Application version string pointer" area in application flash which holds the pointer to where the application version string resides. The application's linker command file ultimately determines where this string is placed in application flash.

It is preferable that the application version string be located at address 0x8400 for MC9S08QE32 parts. The application string can be any characters terminated by the NULL character (0x00). There is not a strict limit on the number of characters in the string, but for practical purposes should be kept under 100 bytes including the terminating NULL character. During an update the bootloader erases the entire application from 0x8400 on. The last page has the vector table specifically the redirected reset vector. The version string pointer and reset vector are used to determine if the application is valid.

Application Interrupt Vector table and Linker Command File

Since the bootloader flash region is read-only, the interrupt vector table is redirected to the region 0xF1C0 to 0xF1FD so that application developers can use hardware interrupts. Note that in order for Application interrupts to function properly, the Application's linker command file (*.prm extension) must be modified appropriately to allow the linker to place the developers code in the correct place in memory. For example, the developer desires to use the serial communications port SCI1 receive interrupt. The developer would add the following line to the Codewarrior linker command file for the project:

```
VECTOR ADDRESS 0x0000F1E0 vSci1Rx
```

This will inform the linker that the interrupt function "vSci1Rx()" should be placed at address 0x0000F1E0. Next, the developer should add a file to their project "vector_table.c" that creates an array of function pointers to the ISR routines used by the application.

```
extern void _Startup(void);/* _Startup located in Start08.c */
extern void vSci1Rx(void);/* sci1 rx isr */
extern short iWriteToSci1(unsigned char *);
void vDummyIsr(void);
#pragma CONST_SEG VECTORS
void (* const vector_table[])(void) = /* Relocated Interrupt vector table */{
    vDummyIsr,/* Int.no. 0 Vtpm3ovf (at F1C0)Unassigned */
    vDummyIsr, /* Int.no. 1 Vtpm3ch5 (at F1C2) Unassigned */
    vDummyIsr, /* Int.no. 2 Vtpm3ch4 (at F1C4) Unassigned */
    vDummyIsr, /* Int.no. 3 Vtpm3ch3 (at F1C6) Unassigned */
    vDummyIsr, /* Int.no. 4 Vtpm3ch2 (at F1C8) Unassigned */
    vDummyIsr, /* Int.no. 5 Vtpm3ch1 (at F1CA) Unassigned */
    vDummyIsr, /* Int.no. 6 Vtpm3ch0 (at F1CC) Unassigned */
    vDummyIsr, /* Int.no. 7 Vrtc (at F1CE) Unassigned */
    vDummyIsr, /* Int.no. 8 Vsci2tx (at F1D0) Unassigned */
    vDummyIsr, /* Int.no. 9 Vsci2rx (at F1D2) Unassigned */
    vDummyIsr, /* Int.no. 10 Vsci2err (at F1D4) Unassigned */
    vDummyIsr, /* Int.no. 11 Vacmpx (at F1D6) Unassigned */
    vDummyIsr, /* Int.no. 12 Vadc (at F1D8) Unassigned */
    vDummyIsr, /* Int.no. 13 Vkeyboard (at F1DA) Unassigned */
    vDummyIsr, /* Int.no. 14 Viic (at F1DC) Unassigned */
    vDummyIsr, /* Int.no. 15 Vsci1tx (at F1DE) Unassigned */
    vSci1Rx, /* Int.no. 16 Vsci1rx (at F1E0) SCI1RX */
    vDummyIsr, /* Int.no. 17 Vsci1err (at F1E2) Unassigned */
    vDummyIsr, /* Int.no. 18 Vspi (at F1E4) Unassigned */
    vDummyIsr, /* Int.no. 19 VReserved12 (at F1E6) Unassigned */
    vDummyIsr, /* Int.no. 20 Vtpm2ovf (at F1E8) Unassigned */
    vDummyIsr, /* Int.no. 21 Vtpm2ch2 (at F1EA) Unassigned */
    vDummyIsr, /* Int.no. 22 Vtpm2ch1 (at F1EC) Unassigned */
    vDummyIsr, /* Int.no. 23 Vtpm2ch0 (at F1EE) Unassigned */
    vDummyIsr, /* Int.no. 24 Vtpm1ovf (at F1F0) Unassigned */
    vDummyIsr, /* Int.no. 25 Vtpm1ch2 (at F1F2) Unassigned */
    vDummyIsr, /* Int.no. 26 Vtpm1ch1 (at F1F4) Unassigned */
    vDummyIsr, /* Int.no. 27 Vtpm1ch0 (at F1F6) Unassigned */
}
```

```

vDummyIsr, /* Int.no. 28 Vld (at F1F8)    Unassigned */
vDummyIsr, /* Int.no. 29 Virq (at F1FA)   Unassigned */
vDummyIsr, /* Int.no. 30 Vswi (at F1FC)   Unassigned */
_Startup  /* Int.no. 31 Vreset (at F1FE)  Reset vector */

};

void vDummyIsr(void){
    for(;;){
        if(iWriteToSci1("STUCK IN UNASSIGNED ISR\n\r>"));
    }
}

```

The interrupt routines themselves can be defined in separate files. The "vDummyIsr" function is used in conjunction with "iWritetoSci1" for debugging purposes.

Bootloader Menu Commands

The bootloader accepts commands from both the local UART and OTA. All OTA commands sent must be Unicast with only 1 byte in the payload for each command. A response will be returned to the sender. All Broadcast and multiple byte OTA packets are dropped to help prevent general OTA traffic from being interpreted as a command to the bootloader while in the menu.

Bypass Mode - "B"

The bootloader provides a "bypass" mode of operation that essentially connects the freescale mcu to the internal microcontroller's serial UART. This allows direct communication to the internal microcontroller's radio for the purpose of firmware and radio configuration changes. Once in bypass mode, the X-CTU utility can change modem configuration and/or update module's firmware. Bypass mode automatically handles any baud rate up to 115.2kbps. Note that this command is unavailable when module is accessed remotely.

Update Firmware - "F"

The "F" command initiates a firmware download for both wired and over-the-air configurations. Depending on the source of the command (received via Over the Air or local UART), the download will proceed via wired or over-the-air respectively.

Adjust Timeout for Update Firmware - "T"

The "T" command changes the timeout before sending a NAK by $\text{Base-Time} * 2^{(T)}$. The Base-Time for the local UART is different than the Base-Time for Over the Air. During a firmware update, the bootloader will automatically increase the Timeout if repeat packets are received or multiple NAKs for the same packet without success occur.

Application Version String - "A"

The "A" command provides the version of the currently loaded application. If no application is present, "Unkown" will be returned.

Bootloader Version String - "V"

The "V" command provides the version of the currently loaded bootloader. The version will return a string in the format BLFFF-HHH-XYZ_DDD where FFF represents the Flash size in kilo bytes, HHH is the hardware, XYZ is the version, and DDD is the preferred XMODEM packet size for updates. Double the preferred packet size is also possible, but not guaranteed. For example "BL032-2B0-023_064" will take 64 byte CRC XMODEM payloads and may take 128 byte CRC XMODEM payloads also. In this case, both 64 and 128 payloads are handled, but the 64 byte payload is preferred for better Over the Air reliability.

Bootloader Version BL032-2x0-025_064 only operates at 9600 baud on the local UART as well as communications to the internal microcontroller. A newer version of the Bootloader BL032-2x0-033_064 or newer BL032-2B0-XXX_064 has changed the baud rate to 115200 between the Programmable and

the internal microcontroller. The internal module is also set to 115200 as the default baud rate. The default rate of the programmable local UART is also set to 115200, however, the local UART has an auto baud feature added to detect if the UART is at the wrong baud rate. If a single character is sent, it will automatically switch to 115200 or 9600 baud.

Firmware Updates

Wired Updates

A user can update their application using the bootloader in a wired configuration with the following steps:

- a. Plug XBee programmable module into a suitable serial port on a PC.
- b. Open a hyperterminal (or similar dumb terminal application) session with 115200 baud, no parity, and 8 data bits with one stop bit.
- c. Hit Enter to display the bootloader menu.
- d. Hit the "F" key to initiate a wired firmware update.
- e. A series of "C" characters Will be displayed within the hyperterminal window. At this point, select the "transfer->send file" menu item. Select the desired flat binary output file.
- f. Select "Xmodem" as the protocol.
- g. Click "Send" on the "Send File" dialog. The file will be downloaded to the XBee Programmable module. Upon a successful update, the bootloader will jump to the newly loaded application.

Over-The-Air updates

A user can update their application using the bootloader in an "over-the-air" configuration with the following steps...(This procedure assumes that the bootloader is running and not the application. The internal microcontroller baud rate of the programmable module must be set to 115200 baud. The bootloader only operates at 115200 baud between the Radio and programmable bootloader. The application must be programmed with some way to support returning to the bootloader in order to support Over the Air (OTA) updates without local intervention.)

- a. Open a hyperterminal session to the host module with no parity, no hardwareflow control, 8 data bits and 1 stop bit. (The host module does not have to operate at the same baud rate as the remote module.) For faster updates and less latency due to the UART, set the host module to a faster baud rate. (i.e. 115200)
- b. Enter 3 pluses "+++" to place the module in command mode. (or XCTU's "Modem Configuration" tab can be used to set the correct parameters)
- c. Set the Host Module destination address to the target module's 64 bit address that the host module will update (ATDH aabbccdd, ATDL eeffgghh, ATCN, where aabbccddeeffgghh is the hexadecimal 64 bit address of the target module).
- d. Hit Enter and the bootloader command menu will be displayed from the remote module. (Note that the option "B" doesn't exist for OTA)
- e. Hit the "F" key to cause the remote module to request the new firmware file over-the-air.
- f. The host module will begin receiving "C" characters indicating that the remote module is requesting an Xmodem CRC transfer. Using XCTU or another terminal program, Select "XMODEM" file transfer. Select the Binary file to upload/transfer. Click Send to start the transfer. At the conclusion of a successful transfer, the bootloader will jump to the newly loaded application.

Output File Configuration

BKGD Programming

P&E Micro provides a background debug tool that allows flashing applications on the MC9S08QE parts through their background debug mode port. By default, the Codewarrior tool produces an "ABS" output file for use in programming parts through the background debug interface. The programmable XBee from the factory has the BKGD debugging capability disabled. In order to debug, a bootloader

with the debug interface enabled needs to be loaded on the secondary processor or a stand-alone app needs to be loaded.

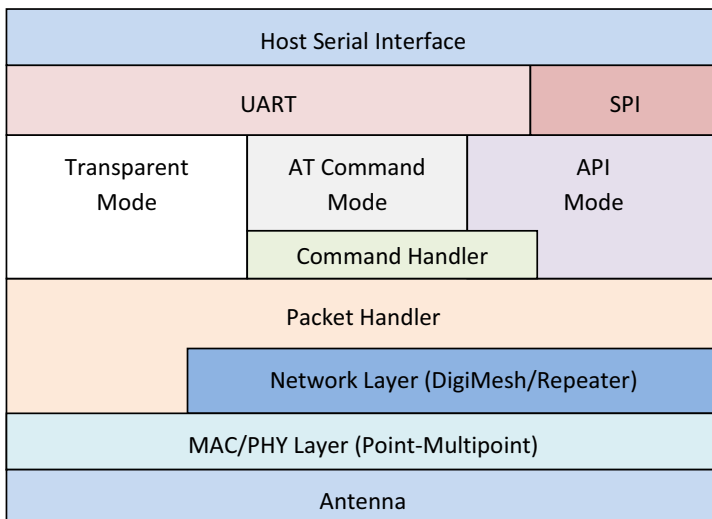
Bootloader updates

The supplied bootloader requires files in a "flat binary" format which differs from the default ABS file produced. The Codewarrior tool also produces a S19 output file. In order to successfully flash new applications, the S19 file must be converted into the flat binary format. Utilities are available on the web that will convert S19 output to "BIN" outputs. Often times, the "BIN" file conversion will pad the addresses from 0x0000 to the code space with the same number. (Often 0x00 or 0xFF) These extra bytes before the APP code starts will need to be deleted from the bin file before the file can be transferred to the bootloader.

2. RF Module Operation

Basic Operational Design

The XBee-PRO® 900HP RF Module uses a multi-layered firmware base to order the flow of data, dependent on the hardware and software configuration chosen by the user. This configuration block diagram is shown below, with the host serial interface as the physical starting point, and the antenna as the physical endpoint for the transferred data. As long as a block is able to touch another block, the two interfaces can interact. For example, if the module is using SPI mode, Transparent Mode is not available. See below:



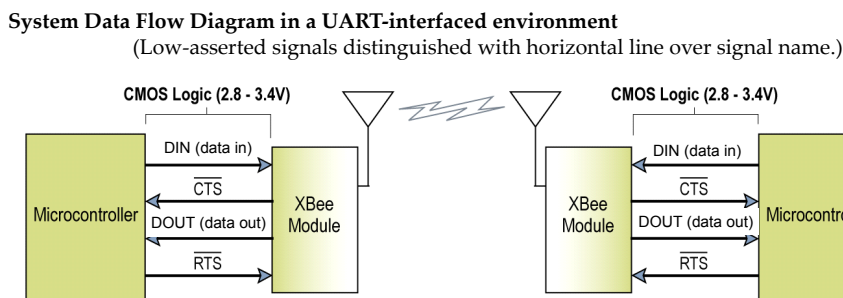
The command handler is the code that processes commands from AT Command Mode or API Mode (see AT Commands section). The command handler can also process commands from remote radios (see Remote AT Commands section).

Serial Communications

XBee RF Modules interface to a host device through a serial port. Through its serial port, the module can communicate with any logic and voltage compatible UART, through a level translator to any serial device (for example, through a RS-232 or USB interface board), or through a Serial Peripheral Interface (SPI), which is a synchronous interface to be described later.

UART Data Flow

Devices that have a UART interface can connect directly to the pins of the RF module as shown in the figure below.

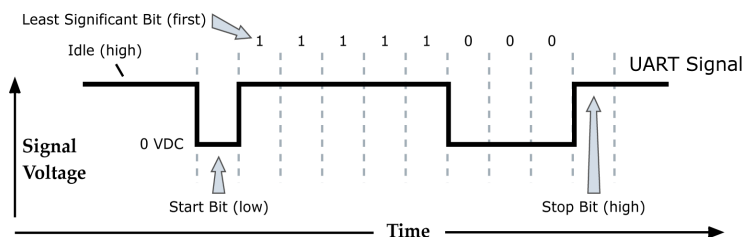


Serial Data

Data enters the module UART through the DIN (pin 3) as an asynchronous serial signal. The signal should idle high when no data is being transmitted.

Each data byte consists of a start bit (low), 8 data bits (least significant bit first) and a stop bit (high). The following figure illustrates the serial bit pattern of data passing through the module.

UART data packet 0x1F (decimal number "31") as transmitted through the RF module
Example Data Format is 8-N-1 (bits - parity - # of stop bits)



Serial communications depend on the two UARTs (the microcontroller's and the RF module's) to be configured with compatible settings (baud rate, parity, start bits, stop bits, data bits).

The UART baud rate, parity, and stop bits settings on the XBee module can be configured with the BD, NB, and SB commands respectively. See the command table in chapter 10 for details.

SPI Communications

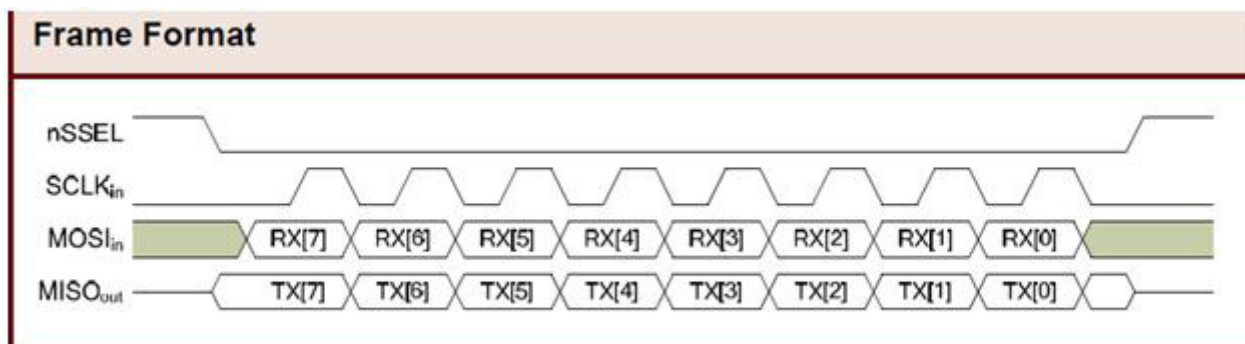
The XBee modules support SPI communications in slave mode. Slave mode receives the clock signal and data from the master and returns data to the master. The SPI port uses the following signals on the XBee:

- SPI_MOSI (Master Out, Slave In) - inputs serial data from the master
- SPI_MISO (Master In, Slave Out) - outputs serial data to the master
- SPI_SCLK (Serial Clock) - clocks data transfers on MOSI and MISO
- SPI_SSEL (Slave Select) - enables serial communication with the slave
- SPI_ATTN (Attention) - alerts the master that slave has data queued to send. The XBee module will assert this pin as soon as data is available to send to the SPI master and it will remain asserted until the SPI master has clocked out all available data.

In this mode, the following apply:

- SPI Clock rates up to 3.5 MHz are possible.
- Data is MSB first
- Frame Format mode 0 is used. This means CPOL=0 (idle clock is low) and CPHA=0 (data is sampled on the clock's leading edge). Mode 0 is diagramed below.
- SPI port is setup for API mode and is equivalent to AP=1.

Frame Format for SPI Communications



SPI Operation

This section specifies how SPI is implemented on the XBee, what the SPI signals are, and how full duplex operations work.

XBee Implementation of SPI

The module operates as a SPI slave only. This means that an external master will provide the clock and will decide when to send. The XBee-PRO 900HP supports an external clock rate of up to 3.5 Mbps.

Data is transmitted and received with most significant bit first using SPI mode 0. This means the CPOL and CPHA are both 0. Mode 0 was chosen because it's the typical default for most microcontrollers and would simplify configuration of the master. Further information on Mode 0 is not included in this manual, but is well-documented on the internet.

SPI Signals

The official specification for SPI includes the four signals **SPI_MISO**, **SPI_MOSI**, **SPI_CLK**, and **SPI_SSEL**. Using only these four signals, the master cannot know when the slave needs to send and the SPI slave cannot transmit unless enabled by the master. For this reason, the **SPI_ATTN** signal is available in the design. This allows the module to alert the SPI master that it has data to send. In turn, the SPI master is expected to assert **SPI_SSEL** and start **SPI_CLK**, unless these signals are already asserted and active respectively. This, in turn, allows the XBee module to send data to the master.

The table below names the SPI signals and specifies their pinouts. It also describes the operation of each pin:

| Signal Name | Pin Number | Applicable AT Command | Description |
|---|------------|-----------------------|---|
| SPI_MISO (Master In, Slave out) | 4 | ATP2 | When SPI_SSEL is asserted (low) and SPI_CLK is active, the module outputs the data on this line at the SPI_CLK rate. When SPI_SSEL is de-asserted (high), this output should be tri-stated such that another slave device can drive the line. |
| SPI_MOSI (Master out, Slave in) | 11 | ATD4 | The SPI master outputs data on this line at the SPI_CLK rate after it selects the desired slave. When the module is configured for SPI operations, this pin is an input. |
| SPI_SSEL (Slave Select) (Master out, Slave in) | 17 | ATD3 | The SPI master outputs a low signal on this line to select the desired slave. When the module is configured for SPI operations, this pin is an input. |
| SPI_CLK (Clock) (Master out, Slave in) | 18 | ATD2 | The SPI master outputs a clock on this pin, and the rate must not exceed the maximum allowed, 3.5 Mbps. When the module is configured for SPI operations, this pin is an input. |
| SPI_ATTN (Attention) (Master in, Slave out) | 19 | ATD1 | The module asserts this pin low when it has data to send to the SPI master. When this pin is configured for SPI operations, it is an output (not tri-stated). |

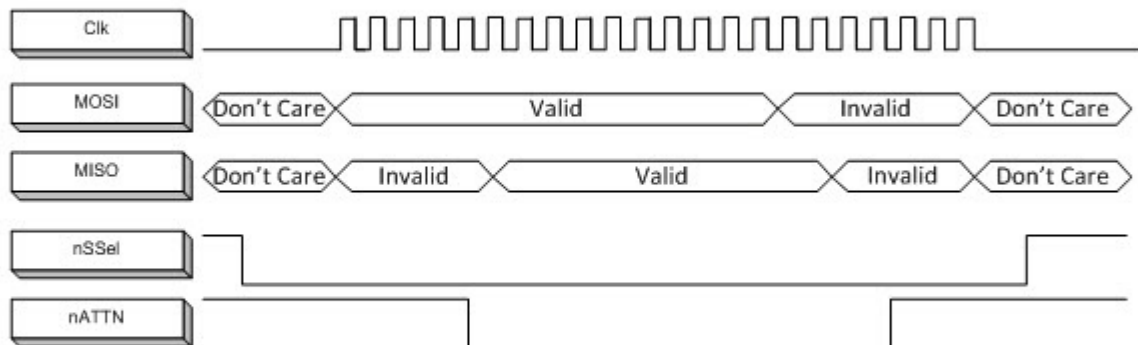
Note: By default, the inputs have pull-up resistors enabled. See the PR command to disable the pull-up resistors. When the SPI pins are not connected but the pins are configured for SPI operation, then the pull-ups are needed for proper UART operation.

Full Duplex Operation

SPI on XBee requires usage of API mode (without escaping) to packetize data. However, by design, SPI is a full duplex protocol, even when data is only available in one direction. This means that whenever data is received, it will also transmit, and that data will normally be invalid. Likewise, whenever data is transmitted, invalid data will probably be received. The means of determining whether or not received data is invalid is by packetizing the data with API packets.

SPI allows for valid data from the slave to begin before, at the same time, or after valid data begins from the master. When the master is sending data to the slave and the slave has valid data to send in the middle of receiving data from the master, this allows a true full duplex operation where data is valid in both directions for a period of time. Not only must the master and the slave both be able to keep up with the full duplex operation, but both sides must honor the protocol as specified.

An example follows to more fully illustrate the SPI interface while valid data is being sent in both directions.



Low Power Operation

In general, sleep modes work the same on SPI as they do on UART. However, due to the addition of SPI mode, there is the option of another sleep pin, as described below:

By default, DIO8 (SLEEP_REQUEST) is configured as a peripheral and is used for pin sleep to awaken and to sleep the radio. This applies regardless of the selected serial interface (UART or SPI).

However, if SLEEP_REQUEST is not configured as a peripheral and SPI_SSEL is configured as a peripheral, then pin sleep is controlled by SPI_SSEL rather than by SLEEP_REQUEST. Asserting SPI_SSEL by driving it low either awakens the radio or keeps it awake. Negating SPI_SSEL by driving it high puts the radio to sleep.

Using SPI_SSEL for two purposes (to control sleep and to indicate that the SPI master has selected a particular slave device) has the advantage of requiring one less physical pin connection to implement pin sleep on SPI. It has the disadvantage of putting the radio to sleep whenever the SPI master negates SPI_SSEL (meaning time will be lost waiting for the device to wake), even if that wasn't the intent. Therefore, if the user has full control of SPI_SSEL so that it can control pin sleep, whether or not data needs to be transmitted, then sharing the pin may be a good option in order to make the SLEEP_REQUEST pin available for another purpose. If the radio is one of multiple slaves on the SPI, then the radio would sleep while the SPI master talks to the other slave, but this is acceptable in most cases.

If neither pin is configured as a peripheral, then the radio stays awake, being unable to sleep in SM1 mode.

Configuration

The three considerations for configuration are:

- How is the serial port selected? (I.e. Should the UART or the SPI port be used?)
- If the SPI port is used, what should be the format of the data in order to avoid processing invalid characters while transmitting?
- What SPI options need to be configured?

Serial Port Selection

In the default configuration the UART and SPI ports will both be configured for serial port operation.

If both interfaces are configured, serial data will go out the UART until the SPI_SSEL signal is asserted. Thereafter, all serial communications will operate on the SPI interface.

If only the UART is enabled, then only the UART will be used, and SPI_ $\overline{\text{SSEL}}$ will be ignored. If only the SPI is enabled, then only the SPI will be used.

If neither serial port is enabled, the module will not support serial operations and all communications must occur over the air. All data that would normally go to the serial port is discarded.

Forcing UART Operation

In the rare case that a module has been configured with only the SPI enabled and no SPI master is available to access the SPI slave port, the module may be recovered to UART operation by holding DIN / $\overline{\text{CONFIG}}$ low at reset time. As always, DIN/ $\overline{\text{CONFIG}}$ forces a default configuration on the UART at 9600 baud and it will bring up the module in command mode on the UART port. Appropriate commands can then be sent to the module to configure it for UART operation. If those parameters are written, then the module will come up with the UART enabled, as desired on the next reset.

SPI Port Selection

SPI mode can be forced by holding DOUT/DIO13 (pin 2) low while resetting the module until SPI_nATTN asserts. By this means, the XBee module will disable the UART and go straight into SPI communication mode. Once configuration is completed, a modem status frame is queued by the module to the SPI port which will cause the SPI_nATTN line to assert. The host can use this to determine that the SPI port has been configured properly. This method internally forces the configuration to provide full SPI support for the following parameters:

- D1 (note this parameter will only be changed if it is at a default of zero when method is invoked)
- D2
- D3
- D4
- P2.

As long as a WR command is not issued, these configuration values will revert back to previous values after a power on reset. If a WR command is issued while in SPI mode, these same parameters will be written to flash. After a reset, parameters that were forced and then written to flash become the mode of operation. If the UART is disabled and the SPI is enabled in the written configuration, then the module will come up in SPI mode without forcing it by holding DOUT low. If both the UART and the SPI are enabled at the time of reset, then output will go to the UART until the host sends the first input. If that first input comes on the SPI port, then all subsequent output will go to the SPI port and the UART will be disabled. If the first input comes on the UART, then all subsequent output will go to the UART and the SPI will be disabled.

When the slave select (SPI_nSSEL) signal is asserted by the master, SPI transmit data is driven to the output pin SPI_MISO, and SPI data is received from the input pin SPI_MOSI. The SPI_nSSEL pin has to be asserted to enable the transmit serializer to drive data to the output signal SPI_MISO. A falling edge on SPI_nSSEL causes the SPI_MISO line to be tri-stated such that another slave device can drive it, if so desired.

If the output buffer is empty, the SPI serializer transmits the last valid bit repeatedly, which may be either high or low. Otherwise, the module formats all output in API mode 1 format, as described in chapter 7. The attached host is expected to ignore all data that is not part of a formatted API frame.

Data Format

The SPI will only operate in API mode 1. Neither transparent mode nor API mode 2 (which escapes control characters) will be supported. This means that the AP configuration only applies to the UART and will be ignored while using the SPI.

SPI Parameters

Most host processors with SPI hardware allow the bit order, clock phase and polarity to be set. For communication with all XBee radios the host processor must set these options as follows:

- Bit Order - send MSB first
- Clock Phase (CPHA) - sample data on first (leading) edge
- Clock Polarity (CPOL) - first (leading) edge rises

This is SPI Mode 0 and MSB first for all XBee radios. Mode 0 means that data is sampled on the leading edge and that the leading edge rises. MSB first means that bit 7 is the first bit of a byte sent over the interface.

Serial Buffers

To enable the UART port, DIN and DOUT must be configured as peripherals. To enable the SPI port, SPI_MISO, SPI_MOSI, SPI_SSEL, and SPI_CLK must be enabled as peripherals. If both ports are enabled then output will go to the UART until the first input on SPI.

When both the UART and SPI ports are enabled on power-up, all serial data will go out the UART. But, as soon as input occurs on either port, that port is selected as the active port and no input or output will be allowed on the other port until the next reset of the module.

If the configuration is changed so that only one port is configured, then that port will be the only one enabled or used. If the parameters are written with only one port enabled, then the port that is not enabled will not even be used temporarily after the next reset.

If both ports are disabled on reset, the UART will be used in spite of the wrong configuration so that at least one serial port will be operational.

Serial Receive Buffer

When serial data enters the RF module through the DIN Pin (or the MOSI pin), the data is stored in the serial receive buffer until it can be processed. Under certain conditions, the module may not be able to process data in the serial receive buffer immediately. If large amounts of serial data are sent to the module such that the serial receive buffer would overflow, then the new data will be discarded. If the UART is in use, this can be avoided by the host side honoring CTS flow control.

If the SPI is the serial port, no hardware flow control is available. It is the user's responsibility to ensure that receive buffer is not overflowed. One reliable strategy is to wait for a TX_STATUS response after each frame sent to ensure that the module has had time to process it.

Serial Transmit Buffer

When RF data is received, the data is moved into the serial transmit buffer and sent out the UART or SPI port. If the serial transmit buffer becomes full and system buffers are also full, then the entire RF data packet is dropped. Whenever data is received faster than it can be processed and transmitted out the serial port, there is a potential of dropping data.

UART Flow Control

The $\overline{\text{RTS}}$ and $\overline{\text{CTS}}$ module pins can be used to provide RTS and/or CTS flow control. CTS flow control provides an indication to the host to stop sending serial data to the module. RTS flow control allows the host to signal the module to not send data in the serial transmit buffer out the UART. RTS and CTS flow control are enabled using the D6 and D7 commands. Please note that serial port flow control is not possible when using the SPI port.

CTS Flow Control

If CTS flow control is enabled (D7 command), when the serial receive buffer is 17 bytes away from being full, the module de-asserts $\overline{\text{CTS}}$ (sets it high) to signal to the host device to stop sending serial data. $\overline{\text{CTS}}$ is re-asserted after the serial receive buffer has 34 bytes of space. See FT for the buffer size.

RTS Flow Control

If RTS flow control is enabled (D6 command), data in the serial transmit buffer will not be sent out the DOUT pin as long as $\overline{\text{RTS}}$ is de-asserted (set high). The host device should not de-assert $\overline{\text{RTS}}$ for long periods of time to avoid filling the serial transmit buffer. If an RF data packet is received, and the serial transmit buffer does not have enough space for all of the data bytes, the entire RF data packet will be discarded.

The UART Data Present Indicator is a useful feature when using RTS flow control. When enabled, the DIO1 line asserts (low asserted) when UART data is queued to be transmitted from the module. See the D1 command in the Command Reference Tables for more information.

Note: If the XBee is sending data out the UART when $\overline{\text{RTS}}$ is de-asserted (set high), the XBee could send up to 5 characters out the UART or SPI port after $\overline{\text{RTS}}$ is de-asserted.

Serial Interface Protocols

The XBee modules support both transparent and API (Application Programming Interface) serial interfaces.

Transparent Operation - UART

When operating in transparent mode, the modules act as a serial line replacement. All UART data received through the DIN pin is queued up for RF transmission. When RF data is received, the data is sent out through the serial port. The module configuration parameters are configured using the AT command mode interface. Please note that transparent operation is not provided when using the SPI.

Data is buffered in the serial receive buffer until one of the following causes the data to be packetized and transmitted:

- No serial characters are received for the amount of time determined by the RO (Packetization Time-out) parameter. If RO = 0, packetization begins when a character is received.
- The Command Mode Sequence (GT + CC + GT) is received. Any character buffered in the serial receive buffer before the sequence is transmitted.
- The maximum number of characters that will fit in an RF packet is received. See the NP parameter.

API Operation

API operation is an alternative to transparent operation. The frame-based API extends the level to which a host application can interact with the networking capabilities of the module. When in API mode, all data entering and leaving the module is contained in frames that define operations or events within the module.

Transmit Data Frames (received through the serial port) include:

- RF Transmit Data Frame
- Command Frame (equivalent to AT commands)

Receive Data Frames (sent out the serial port) include:

- RF-received data frame
- Command response
- Event notifications such as reset, etc.

The API provides alternative means of configuring modules and routing data at the host application layer. A host application can send data frames to the module that contain address and payload information instead of using command mode to modify addresses. The module will send data frames to the application containing status packets; as well as source, and payload information from received data packets.

The API operation option facilitates many operations such as the examples cited below:

- > Transmitting data to multiple destinations without entering Command Mode
- > Receive success/failure status of each transmitted RF packet
- > Identify the source address of each received packet

A Comparison of Transparent and API Operation

The following table compares the advantages of transparent and API modes of operation:

| Transparent Operation Features | |
|--|---|
| Simple Interface | All received serial data is transmitted unless the module is in command mode. |
| Easy to support | It is easier for an application to support transparent operation and command mode |
| API Operation Features | |
| Easy to manage data transmissions to multiple destinations | Transmitting RF data to multiple remotes only requires changing the address in the API frame. This process is much faster than in transparent operation where the application must enter AT command mode, change the address, exit command mode, and then transmit data. Each API transmission can return a transmit status frame indicating the success or reason for failure. |
| Received data frames indicate the sender's address | All received RF data API frames indicate the source address. |
| Advanced addressing support | API transmit and receive frames can expose addressing fields including source and destination endpoints, cluster ID and profile ID. |

| Transparent Operation Features | |
|---------------------------------|--|
| Advanced networking diagnostics | API frames can provide indication of IO samples from remote devices, and node identification messages. |
| Remote Configuration | Set / read configuration commands can be sent to remote devices to configure them as needed using the API. |

As a general rule of thumb, API mode is recommended when a device:

- sends RF data to multiple destinations
- sends remote configuration commands to manage devices in the network
- receives RF data packets from multiple devices, and the application needs to know which device sent which packet
- must support multiple endpoints, cluster IDs, and/or profile IDs
- uses the Device Profile services.

API mode is required when:

- receiving I/O samples from remote devices
- using SPI for the serial port

If the above conditions do not apply (e.g. a sensor node, router, or a simple application), then transparent operation might be suitable. It is acceptable to use a mixture of devices running API mode and transparent mode in a network.

Modes of Operation

Description of Modes

When not transmitting data, the RF module is in Receive Mode. The module shifts into the other modes of operation under the following conditions:

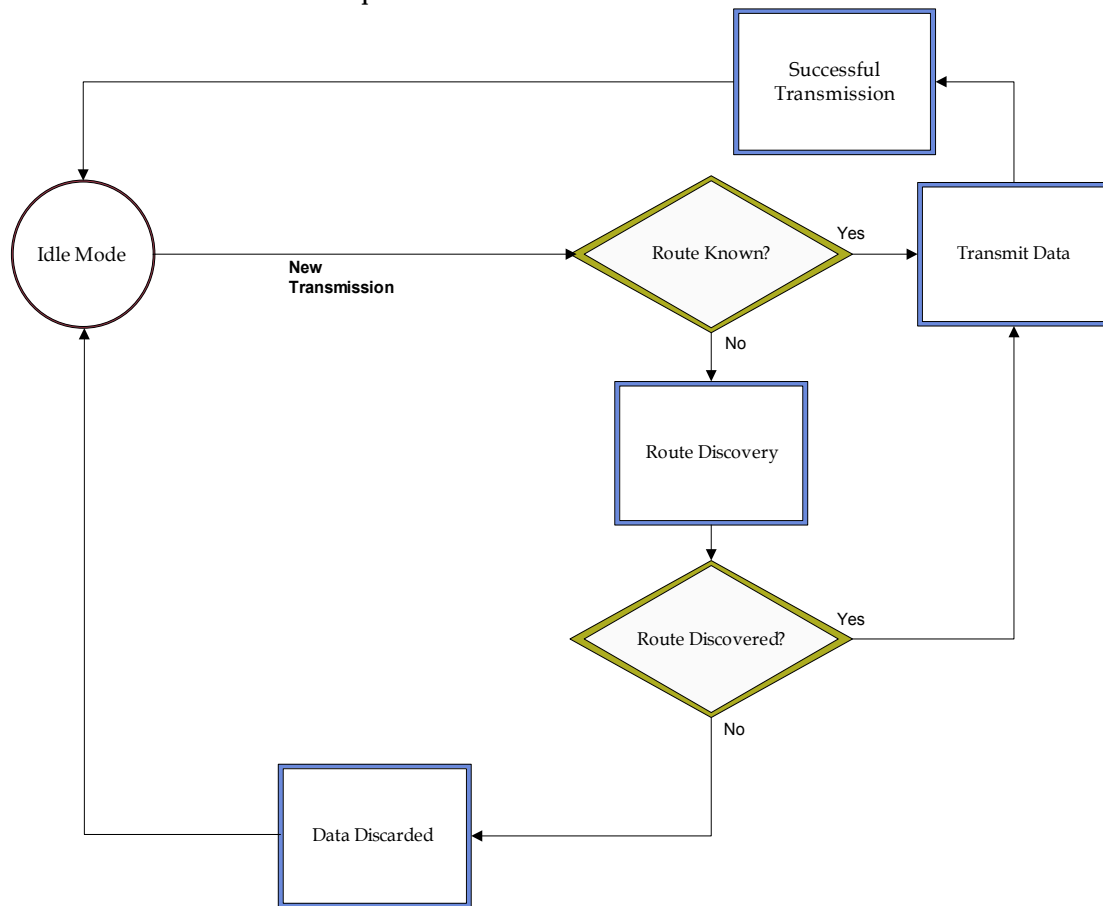
- Transmit Mode (Serial data in the serial receive buffer is ready to be packetized)
- Sleep Mode
- Command Mode (Command Mode Sequence is issued, not available when using the SPI port)

Transmit Mode

When serial data is received and is ready for packetization, the RF module will attempt to transmit the data. The destination address determines which node(s) will receive and send the data.

In the diagram below, route discovery applies only to DigiMesh transmissions. The data will be transmitted once a route is established. If route discovery fails to establish a route, the packet will be discarded.

Transmit Mode Sequence



When DigiMesh data is transmitted from one node to another, a network-level acknowledgement is transmitted back across the established route to the source node. This acknowledgement packet indicates to the source node that the data packet was received by the destination node. If a network acknowledgement is not received, the source node will re-transmit the data.

See Data Transmission and Routing in chapter 4 for more information.

Receive Mode

If a valid RF packet is received, the data is transferred to the serial transmit buffer. This is the default mode for the XBee radio.

Command Mode

To modify or read RF Module parameters, the module must first enter into Command Mode - a state in which incoming serial characters are interpreted as commands. The API Mode section in Chapter 7 describes an alternate means for configuring modules which is available with the SPI, as well as over the UART with code.

AT Command Mode

To Enter AT Command Mode:

Send the 3-character command sequence “+++” and observe guard times before and after the command characters. [Refer to the “Default AT Command Mode Sequence” below.]

Default AT Command Mode Sequence (for transition to Command Mode):

- No characters sent for one second [GT (Guard Times) parameter = 0x3E8]
- Input three plus characters (“+++”) within one second [CC (Command Sequence Character) parameter = 0x2B.]
- No characters sent for one second [GT (Guard Times) parameter = 0x3E8]

Once the AT command mode sequence has been issued, the module sends an “OK\r” out the UART pin. The “OK\r” characters can be delayed if the module has not finished transmitting received serial data.

When command mode has been entered, the command mode timer is started (CT command), and the module is able to receive AT commands on the UART port.

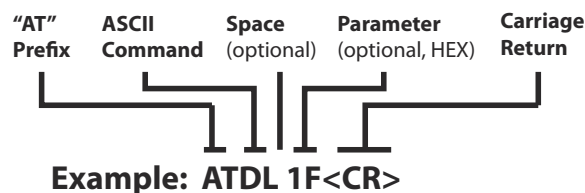
All of the parameter values in the sequence can be modified to reflect user preferences.

NOTE: Failure to enter AT Command Mode is most commonly due to baud rate mismatch. By default, the BD (Baud Rate) parameter = 3 (9600 bps).

To Send AT Commands:

Send AT commands and parameters using the syntax shown below.

Syntax for sending AT Commands



To read a parameter value stored in the RF module's register, omit the parameter field.

The preceding example would change the RF module Destination Address (Low) to “0x1F”. To store the new value to non-volatile (long term) memory, send the WR (Write) command. This allows modified parameter values to persist in the module's registry after a reset. Otherwise, parameters are restored to previously saved values after the module is reset.

Command Response

When a command is sent to the module, the module will parse and execute the command. Upon successful execution of a command, the module returns an "OK" message. If execution of a command results in an error, the module returns an "ERROR" message.

Applying Command Changes

Any changes made to the configuration command registers through AT commands will not take effect until the changes are applied. For example, sending the BD command to change the baud rate will not change the actual baud rate until changes are applied. Changes can be applied in one of the following ways:

- The AC (Apply Changes) command is issued.
- AT command mode is exited.

To Exit AT Command Mode:

1. Send the ATCN (Exit Command Mode) command (followed by a carriage return).
[OR]
2. If no valid AT Commands are received within the time specified by CT (Command Mode Timeout) Command, the RF module automatically returns to Idle Mode.

For an example of programming the RF module using AT Commands and descriptions of each configurable parameter, please see the Command Reference Table chapter.

Sleep Mode

Sleep modes allow the RF module to enter states of low power consumption when not in use. XBee RF modules support both pin sleep (sleep mode entered on pin transition) and cyclic sleep (module sleeps for a fixed time). XBee sleep modes are discussed in detail in chapter 5.

3. Networking Methods

This chapter will attempt to explain the basic layers and the three networking methods available on the XBee-PRO® 900HP RF modules, building from the simplest to the most complex.

MAC/PHY Basics

PHY is short for Physical Layer. It is responsible for managing the hardware that modulates and demodulates the RF bits.

MAC stands for Media Access Layer. The MAC layer is responsible for sending and receiving RF frames. As part of each packet, there is a MAC layer data header that has addressing information as well as packet options. This layer implements packet acknowledgements (ACKs), packet tracking to eliminate duplicates, etc.

When a radio is transmitting, it cannot receive packets. When a radio is not sleeping, it is either receiving or transmitting. There are no beacons or master/slave requirements in the design of the MAC/PHY.

This radio uses a patented method for scanning and finding a transmission. When a radio transmits, it sends out a repeated preamble pattern, a MAC header, optionally a network header, followed then by packet data. A receiving radio is able to scan all the channels to find a transmission during the preamble, then once it has locked into that it will attempt to receive the whole packet.

Related parameters: CM, HP, ID, PL, RR, MT

The Preamble ID (HP) can be changed to make it so a group of radios will not interfere with another group of radios in the same vicinity. The advantage of changing this parameter is that a receiving radio will not even lock into a transmission of a transmitting radio that does not have the same ID.

The Network ID (ID) can be changed to further keep radios from interfering with each other. This ID is matched after the preamble pattern has been matched, and the MAC header has been received. Networks are defined with a unique network identifier. For modules to communicate they must be configured with the same network identifier. The ID parameter allows multiple networks to co-exist on the same physical channel.

The Channel Mask (CM) parameter determines the channels that the radio will choose to communicate on. See CM in the command reference.

Power Level (PL) sets the TX power level. The power level can be reduced from the maximum to reduce current consumption or for testing. This comes at the expense of reduced radio range.

The RR parameter specifies the number of time a sending radio will attempt to get an ACK from a destination radio when sending a packet.

The MT parameter specifies the number of times that a broadcast packet is repeatedly transmitted. This adds redundancy that improves reliability.

Addressing Basics

Related parameters: SH, SL, DH, DL, TO

64-bit Addresses

Each radio is given a unique IEEE 64-bit address at the factory. This can be read with the SH and SL commands. This is the source address that is returned in API mode of the radio that sent a packet. At this time addresses are of the form: 0x0013A2XXXXXXXXXX. The first 6 digits are the Digi (MaxStream) OUI. The broadcast address is 0x000000000000FFFF.

Unicast

To transmit to a specific radio:

- When using transparent mode set DH:DL to the SH:SL of the destination radio.
- For API mode, set the SH:SL address in the 64-bit destination address.

Broadcast

To transmit to all radios:

- For transparent mode set DH:DL to 0x000000000000FFFF, and for API mode set the 64-bit destination address to 0x000000000000FFFF.
- The scope of the broadcast changes based on the delivery method chosen.

Delivery Method

There are three delivery methods supported by this radio:

- Point to multipoint. (0x40)
- Repeater (Directed broadcast). (0x80)
- DigiMesh. (0xC0)

The TO parameter is the default delivery method used by transparent mode. For API transmissions the TxOptions API field is used to specify the delivery method. When the TxOptions API field is set to 0, the value in the TO parameter will also be used by API transmissions.

The three delivery modes are described below:

Point to Point/Multipoint (P2MP)

This delivery mode does not use a network header, only the MAC header. All messages are always sent directly to the destination. There is no repeating of the packet by other nodes.

A P2MP unicast is only delivered directly to the destination radio, which must be in range of the sending radio. This radio uses patented technology that allows the destination radio to receive transmissions directed to it, even when there is a large amount of traffic. This works best when broadcast transmissions are kept to a minimum. A P2MP broadcast transmission is repeated MT+1 times by the sending node, but is not repeated by nodes which receive it, so like a unicast transmission, the receiving radio must be in range. All radios that receive a P2MP broadcast transmission will output the data through the serial port.

Throughput

10 kbps version, 115.2 kbps serial data rate

| Configuration | Data Throughput |
|---|-----------------|
| Point to point unicast, Encryption Disabled | 8.8 kbps |
| Point to point unicast, Encryption Enabled | 8.7 kbps |

200 kbps version, 115.2 kbps serial data rate

| Configuration | Data Throughput |
|---|-----------------|
| Point to point unicast, Encryption Disabled | 105.5 kbps |
| Point to point unicast, Encryption Enabled | 105.4 kbps |

Note: Data throughput measurements were made setting the serial interface rate to 115200 bps, and measuring the time to send 100,000 bytes from source to destination. During the test, no route discoveries or failures occurred.

Repeater/Directed Broadcast

Related parameters: CE, NH, NN, BH

Directed broadcast transmissions will be received and repeated by all routers in the network. Because ACKs are not used the originating node will send the broadcast multiple times. By default a broadcast transmission is sent four times. Essentially the extra transmissions become automatic retries without acknowledgments. This will result in all nodes repeating the transmission four times as well. Sending frequent broadcast transmissions can quickly reduce the available network bandwidth and as such should be used sparingly.

The MAC layer is the building block that is used to build repeater capability. Repeater mode is implemented with a network layer header that comes after the MAC layer header in each packet. In this network layer there is additional packet tracking to eliminate duplicate broadcasts. In this delivery method, unicasts and broadcast packets are both sent out as broadcasts that are always repeated. All repeated packets are sent to every radio. Broadcast data will be sent out the serial port of all radios that receive it.

When a unicast is sent, it specifies a destination address in the network header. Only the radio that has the matching destination address then will send it out the serial port. This is called a directed broadcast. Any node that has a CE parameter set to route will rebroadcast the packet if its broadcast hops (BH) or broadcast radius values have not been depleted. If a repeated broadcast has already been seen, the node will ignore it. The NH parameter sets the maximum number of hops that a broadcast will be repeated. This value is always used, unless a BH value is specified that is smaller.

By default the CE parameter is set to route all broadcasts. As such, all nodes that receive a repeated packet will repeat it. By changing the CE parameter, you can limit which nodes repeat packets, which can help dense networks from becoming overly congested while packets are being repeated.

Transmission timeout calculations for directed broadcast/repeater mode are the same as for DigiMesh, and can be found below in the DigiMesh section.

DigiMesh Networking

Related Command: MR

In the same manner as the repeater delivery method, DigiMesh builds on P2MP and repeater modes. In DigiMesh, broadcasts always use repeater delivery method, but unicasts use meshing technologies. In the DigiMesh network layer, there are additional network layer ACKs and NACKs. Mesh networking allows messages to be routed through several different nodes to a final destination. DigiMesh firmware allows manufacturers and system integrators to bolster their networks with the self-healing attributes of mesh networking. In the event that one RF connection between nodes is lost (due to power-loss, environmental obstructions, etc.) critical data can still reach its destination due to the mesh networking capabilities embedded inside the modules. Note that if you disable network ACKs, the network will never heal.

DigiMesh Feature Set

DigiMesh contains the following features

- **Self-healing**
Any node may enter or leave the network at any time without causing the network as a whole to fail.
- **Peer-to-peer architecture**
No hierarchy and no parent-child relationships are needed.
- **Quiet Protocol**
Routing overhead will be reduced by using a reactive protocol similar to AODV.
- **Route Discovery**
Rather than maintaining a network map, routes will be discovered and created only when needed.
- **Selective acknowledgements**
Only the destination node will reply to route requests.
- **Reliable delivery**
Reliable delivery of data is accomplished by means of acknowledgements.
- **Sleep Modes**
Low power sleep modes with synchronized wake are supported with variable sleep and wake times.

Data Transmission and Routing

Unicast Addressing

When transmitting while using DigiMesh Unicast communications, reliable delivery of data is accomplished using retries and acknowledgements. The number of mesh network retries is determined by the MR (Mesh Network Retries) parameter. RF data packets are sent up to MR + 1 times across the network route, and ACKs (acknowledgements) are transmitted by the receiving node upon receipt. If a network ACK is not received within the time it would take for a packet to traverse the network twice, a retransmission occurs. Note that when sending a DigiMesh Unicast that both MAC and NWK retries/acknowledgments are used.

MAC retries/acknowledgments are used for transmissions between adjacent nodes in the route. NWK retries/acknowledgments are used across the entire route.

To send Unicast messages, set the DH and DL on the transmitting module to match the corresponding SH and SL parameter values on the receiving module.

Routing

A module within a mesh network is able to determine reliable routes using a routing algorithm and table. The routing algorithm uses a reactive method derived from AODV (Ad-hoc On-demand Distance Vector). An associative routing table is used to map a destination node address with its next hop. By sending a message to the next hop address, either the message will reach its destination or be forwarded to an intermediate router which will route the message on to its destination. A message with a broadcast address is broadcast to all neighbors. All routers receiving the message will rebroadcast the message MT+1 times and eventually the message will reach all corners of the network. Packet tracking prevents a node from resending a broadcast message more than MT+1 times.

Route Discovery

If the source node doesn't have a route to the requested destination, the packet is queued to await a route discovery (RD) process. This process is also used when a route fails. A route fails when the source node uses up its network retries without ever receiving an ACK. This results in the source node initiating RD.

RD begins by the source node broadcasting a route request (RREQ). Any router that receives the RREQ that is not the ultimate destination is called an intermediate node.

Intermediate nodes may either drop or forward a RREQ, depending on whether the new RREQ has a better route back to the source node. If so, information from the RREQ is saved and the RREQ is updated and broadcast. When the ultimate destination receives the RREQ, it unicasts a route reply (RREP) back to the source node along the path of the RREQ. This is done regardless of route quality and regardless of how many times an RREQ has been seen before.

This allows the source node to receive multiple route replies. The source node selects the route with the best round trip route quality, which it will use for the queued packet and for subsequent packets with the same destination address.

Throughput

Throughput in a DigiMesh network can vary by a number of variables, including: number of hops, encryption enabled/disabled, sleeping end devices, failures/route discoveries. Our empirical testing showed the following throughput performance in a robust operating environment (low interference).

200 kbps version, 115.2 kbps serial data rate, 100 KB

| Configuration | Data Throughput |
|--|-----------------|
| Mesh unicast, 1 hop, Encryption Disabled | 91.0 kbps |
| Mesh unicast, 3 hop, Encryption Disabled | 32.5 kbps |
| Mesh unicast, 6 hop, Encryption Disabled | 16.7 kbps |
| Mesh unicast, 1 hop, Encryption Enabled | 89.3 kbps |
| Mesh unicast, 3 hop, Encryption Enabled | 32.2 kbps |
| Mesh unicast, 6 hop, Encryption Enabled | 16.1 kbps |

Note: Data throughput measurements were made setting the serial interface rate to 115200 bps, and measuring the time to send 100,000 bytes from source to destination. During the test, no route discoveries or failures occurred.

Transmission Timeouts

When a node receives an API TX Request (API configured modules) or an RO timeout occurs (modules configured for Transparent Mode) the time required to route the data to its destination depends on a number of configured parameters, whether the transmission is a unicast or a broadcast, and if the route to the destination address is known. Timeouts or timing information is provided for the following transmission types:

- Transmitting a broadcast
- Transmitting a unicast with a known route

- Transmitting a unicast with an unknown route
- Transmitting a unicast with a broken route.

Note: The timeouts in this section are theoretical timeouts and not precisely accurate. The application should pad the calculated maximum timeouts by a few hundred milliseconds. When using API mode, Tx Status API packets should be the primary method of determining if a transmission has completed.

Unicast One Hop Time

A building block of many of the calculations presented below is the unicastOneHopTime. As its name indicates, it represents the amount of time it takes to send a unicast transmission between two adjacent nodes. It is dependent upon the %H setting. It is defined as follows:

$\text{unicastOneHopTime} = \%H$

Transmitting a broadcast

A broadcast transmission must be relayed by all routers in the network. The maximum delay would be when the sender and receiver are on the opposite ends of the network. The NH and %H parameters define the maximum broadcast delay as follows:

$\text{BroadcastTxTime} = \text{NH} * \%H$

Transmitting a unicast with a known route

When a route to a destination node is known the transmission time is largely a function of the number of hops and retries. The timeout associated with a unicast assumes the maximum number of hops is necessary (as specified by NH). The timeout can be estimated in the following manner:

$\text{knownRouteUnicast} = 2 * \text{NH} * \text{MR} * \text{unicastOneHopTime}$

Transmitting a unicast with an unknown route

If the route to the destination is not known the transmitting module will begin by sending a route discovery. If the route discovery is successful and a route is found then the data is transmitted. The timeout associated with the entire operation can be estimated as follows:

$\text{unknownRouteUnicast} = \text{BroadcastTxTime} + \text{NH} * \text{unicastOneHopTime} + \text{knownRouteUnicast}$

Transmitting a unicast with a broken route

If the route to a destination node has changed since the last time a route discovery was completed a node will begin by attempting to send the data along the previous route. After it fails a route discovery will be initiated and, upon completion of the route discovery, the data will be transmitted along the new route. The timeout associated with the entire operation can be estimated as follows:

$\text{brokenRouteUnicast} = \text{BroadcastTxTime} + \text{NH} * \text{unicastOneHopTime} + 2 * \text{knownRouteUnicast}$

4. Sleep Mode

A number of low-power modes exist to enable modules to operate for extended periods of time on battery power. These sleep modes are enabled with the SM command. The sleep modes are characterized as either asynchronous (SM = 1, 4, 5) or synchronous (SM = 7,8). Asynchronous sleeping modes should not be used in a synchronous sleeping network, and vice versa.

Asynchronous sleep modes can be used to control the sleep state on a module by module basis. Modules operating in an asynchronous sleep mode should not be used to route data. Digi strongly encourages users to set asynchronous sleeping modules as non-routing nodes using the CE command. This will prevent the node from attempting to route data.

The synchronous sleep feature of DigiMesh makes it possible for all nodes in the network to synchronize their sleep and wake times. All synchronized cyclic sleep nodes enter and exit a low power state at the same time. This forms a cyclic sleeping network. Nodes synchronize by receiving a special RF packet called a sync message which is sent by a node acting as a sleep coordinator. A node in the network can become a coordinator through a process called nomination. The sleep coordinator will send one sync message at the beginning of each wake period. The sync message is sent as a broadcast and repeated by every node in the network. The sleep and wake times for the entire network can be changed by locally changing the settings on an individual node. The network will use the most recently set sleep settings.

Sleep Modes

Normal Mode (SM=0)

Normal mode is the default for a newly powered-on node. In this mode, a node will not sleep. Normal mode nodes should be mains-powered.

A normal mode module will synchronize to a sleeping network, but will not observe synchronization data routing rules (it will route data at any time, regardless of the wake state of the network). When synchronized, a normal node will relay sync messages generated by sleep-compatible nodes but will not generate sync messages. Once a normal node has synchronized with a sleeping network, it can be put into a sleep-compatible sleep mode at any time.

Asynchronous Pin Sleep Mode (SM=1)

Pin sleep allows the module to sleep and wake according to the state of the **Sleep_RQ** pin (pin 9). Pin sleep mode is enabled by setting the SM command to 1. When **Sleep_RQ** is asserted (high), the module will finish any transmit or receive operations and enter a low-power state. The module will wake from pin sleep when the **Sleep_RQ** pin is de-asserted (low). When indirect messaging polling is enabled (see the CE command), a poll will be sent upon waking to the module's parent node as described in the Indirect Messaging and Polling Section.

Asynchronous Cyclic Sleep Mode (SM=4)

Cyclic sleep allows the module to sleep for a specified time and wake for a short time to poll. Cyclic sleep mode is enabled by setting the SM command to 4. In cyclic sleep, the module sleeps for a specified time. If the XBee receives serial or RF data while awake, it will then extend the time before it returns to sleep by the amount specified by the ST command. Otherwise, it will enter sleep mode immediately. The **On_SLEEP** line is asserted (high) when the module wakes, and is de-asserted (low) when the module sleeps. If hardware flow control is enabled (D7 command), the **CTS** pin will assert (low) when the module wakes and can receive serial data, and de-assert (high) when the module sleeps. When indirect messaging polling is enabled (see the CE command), a poll will be sent upon waking to the module's parent node as described in the Indirect Messaging and Polling Section.

Asynchronous Cyclic Sleep with Pin Wake Up Mode (SM=5)

(SM=5) is similar to both the (SM=1) and (SM=4) modes. When the SLEEP_REQUEST pin is asserted the module will enter a cyclic sleep mode similar to (SM=4). When the SLEEP_REQUEST pin is de-asserted the module will immediately wake up. The module will not sleep when the SLEEP_REQUEST pin is de-asserted.

When indirect messaging polling is enabled (see the CE command) upon waking a poll will be sent to the module's parent node as described in the Indirect Messaging and Polling Section. Polls will also be regularly sent to the parent while the module is held awake.

Synchronous Sleep Support Mode (SM=7)

A node in synchronous sleep support mode will synchronize itself with a sleeping network but will not itself sleep. At any time, the node will respond to new nodes which are attempting to join the sleeping network with a sync message. A sleep support node will only transmit normal data when the other nodes in the sleeping network are awake. Sleep support nodes are especially useful when used as preferred sleep coordinator nodes and as aids in adding new nodes to a sleeping network.

Note: Because sleep support nodes do not sleep, they should be mains powered.

Synchronous Cyclic Sleep Mode (SM=8)

A node in synchronous cyclic sleep mode sleeps for a programmed time, wakes in unison with other nodes, exchanges data and sync messages, and then returns to sleep. While asleep, it cannot receive RF messages or read commands from the UART port. Generally, sleep and wake times are specified by the SP and ST respectively of the network's sleep coordinator. These parameters are only used at start up until the node is synchronized with the network. When a module has synchronized with the network, its sleep and wake times can be queried with the OS and OW commands respectively. If D9 = 1 (**ON_SLEEP** enabled) on a cyclic sleep node, the **ON_SLEEP** line will assert when the module is awake and de-assert when the module is asleep. **CTS** is also de-asserted while asleep (D7 = 1). A newly-powered unsynchronized sleeping node will poll for a synchronized message and then sleep for the period specified by SP, repeating this cycle until it becomes synchronized by receiving a sync message. Once a sync message is received, the node will synchronize itself with the network.

Note: All nodes in a synchronous sleep network should be configured to operate in either Synchronous Sleep Support Mode or Synchronous Cyclic Sleep Mode. Asynchronous sleeping nodes are not compatible with synchronous sleep nodes.

Asynchronous Sleep Operation

Wake Timer

In cyclic sleep mode (SM=4 or SM=5), if serial or RF data is received, the module will start a sleep timer (time until sleep). Any data received serially or by RF link will reset the timer. The timer duration can be set using the ST command. The module returns to sleep when the sleep timer expires.

Indirect Messaging and Polling (P2MP Packets Only)

The messaging mode command (CE) can be used to enable indirect messaging and polling. This enables reliable communication with asynchronous sleeping devices.

Indirect Messaging

Indirect messaging is a communication mode designed for communicating with asynchronous sleeping devices. A module can enable indirect messaging by making itself an indirect messaging coordinator with the CE command. An indirect messaging coordinator does not immediately transmit a P2MP unicast when it is received over the serial port. Instead the module holds onto the data until it is requested via a poll. On receiving a poll the indirect messaging coordinator will send a queued data packet (if available) to the requestor.

Because it is possible for polling device to be eliminated, a mechanism is in place to purge unrequested data packets. If the coordinator holds an indirect data packet for an indirect messaging poller for more than 2.5 times its SP value, then the packet is purged. Users are encouraged to set the SP of the coordinator to the same value as the highest SP time that exists among the pollers in the network. If the coordinator is in API mode, a TxStatus message is generated for a purged data packet with a status of 0x75 (INDIRECT_MESSAGE_UNREQUESTED).

An indirect messaging coordinator will queue up as many data packets as it has buffers available. After the coordinator has used all of its available buffers, it will hold transmission requests unprocessed on the serial input queue. After the serial input queue is full, $\overline{\text{CTS}}$ will be de-asserted (if hardware flow control is enabled). Obviously, after receiving a poll or purging data from the indirect messaging queue the buffers become available again.

Indirect messaging has no effect on P2MP broadcasts, directed broadcasts, repeater packets, or DigiMesh packets. These messages are sent immediately when received over the serial port and are not put on the indirect messaging queue

Polling

Polling is the automatic process by which a node can request data from an indirect messaging coordinator. Polling can be enabled on a device by configuring it as an indirect messaging poller with the CE command and setting its DH:DL registers to match the SH:SL registers of the module which will function as the Indirect Messaging Coordinator. When polling is enabled, the module will send a P2MP poll request regularly to the address specified by the DH:DL registers. When a P2MP unicast is sent to the destination specified by the DH:DL of an polling module, the data will also function as a poll.

When a polling device is also an asynchronous sleeping device, then that device will send a poll shortly after waking from sleep. After that first poll is sent, the module will send polls in the normal manner described above until it returns to sleep.

The 200K data rate product will send polls at least every 100ms when awake. The 10K data rate product will send polls at least every 300ms when awake.

Synchronous Sleep Operation (DigiMesh networks only)

The Sleeping Router feature of DigiMesh makes it possible for all nodes in the network to synchronize their sleep and wake times. All synchronized cyclic sleep nodes enter and exit a low power state at the same time. This forms a cyclic sleeping network. Nodes synchronize by receiving a special RF packet called a sync message which is sent by a node acting as a sleep coordinator. A node in the network can become a sleep coordinator through a process called nomination. The sleep coordinator will send one sync message at the beginning of each wake period. The sync message is sent as a broadcast and repeated by every node in the network. The sleep and wake times for the entire network can be changed by locally changing the settings on an individual node. The network will use the most recently set sleep settings.

Operation

One node in a sleeping network acts as the sleeping coordinator. The process by which a node becomes a sleep coordinator is described later in this document. During normal operations, at the beginning of a wake

cycle the sleep coordinator will send a sync message as a broadcast to all nodes in the network. This message contains synchronization information and the wake and sleep times for the current cycle. All cyclic sleep nodes receiving a sync message will remain awake for the wake time and then sleep for the sleep period specified.

The sleep coordinator will send one sync message at the beginning of each cycle with the currently configured wake and sleep times. All router nodes which receive this sync message will relay the message to the rest of the network. If the sleep coordinator does not hear a re-broadcast of the sync message by one of its immediate neighbors then it will re-send the message one additional time. It should be noted that if SP or ST are changed, the network will not apply the new settings until the beginning of the next wake time. See the Changing Sleep Parameters section below for more information.

A sleeping router network is robust enough that an individual node can go several cycles without receiving a sync message (due to RF interference, for example). As a node misses sync messages, the time available for transmitting messages in the wake time is reduced to maintain synchronization accuracy. By default, a module will also reduce its active sleep time progressively as sync messages are missed.

Synchronization Messages

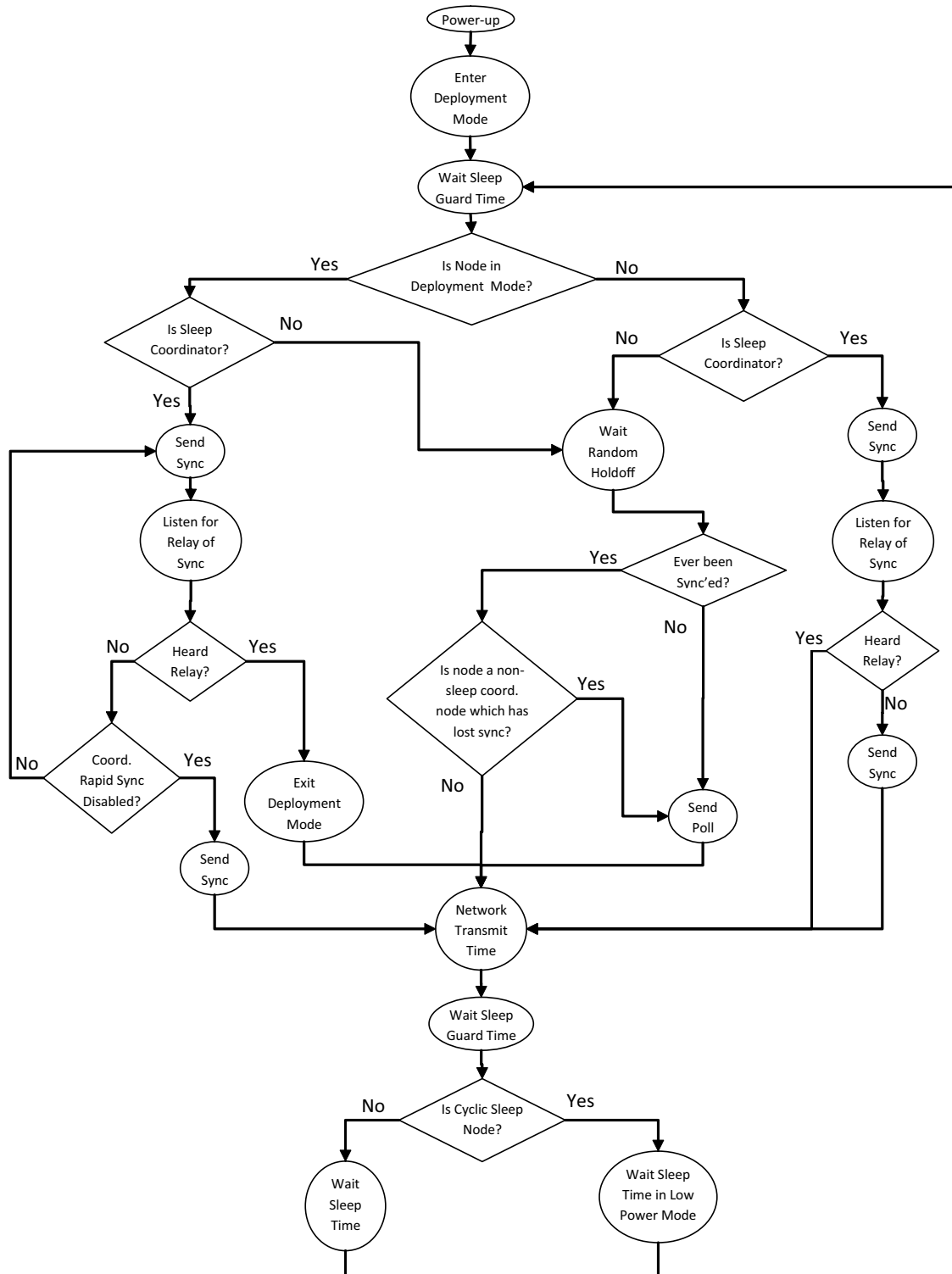
A sleep coordinator will regularly send sync messages to keep the network in sync. Nodes which have not been synchronized or, in some cases, which have lost sync will also send messages requesting sync information.

Deployment mode is used by sleep compatible nodes when they are first powered up and the sync message has not been relayed. A sleep coordinator in deployment mode will rapidly send sync messages until it receives a relay of one of those messages. This allows a network to be deployed more effectively and allows a sleep coordinator which is accidentally or intentionally reset to rapidly re-synchronize with the rest of the network. If a node which has exited deployment mode receives a sync message from a sleep coordinator which is in deployment mode, the sync will be rejected and a corrective sync will be sent to the sleep coordinator. Deployment mode can be disabled using the sleep options command (SO).

A sleep coordinator which is not in deployment mode or which has had deployment mode disabled will send a sync message at the beginning of the wake cycle. The sleep coordinator will then listen for a neighboring node to relay the sync. If the relay is not heard, the sync coordinator will send the sync one additional time.

A node which is not acting as a sleep coordinator which has never been synchronized will send a message requesting sync information at the beginning of its wake cycle. Synchronized nodes which receive one of these messages will respond with a synchronization packet. Nodes which are configured as non-sleep coordinators (using the SO command) which have gone six or more cycles without hearing a sync will also send a message requesting sync at the beginning of their wake period.

The following diagram illustrates the synchronization behavior of sleep compatible modules:



Becoming a Sleep Coordinator

A node can become a sleep coordinator in one of four ways:

Preferred Sleep Coordinator Option

A node can be specified to always act as a sleep coordinator. This is done by setting the preferred sleep coordinator bit (bit 0) in the sleep operations parameter (SO) to 1. A node with the sleep coordinator bit set will always send a sync message at the beginning of a wake cycle. For this reason, it is imperative that no more than one node in the network has this bit set. Although it is not necessary to specify a preferred sleep coordinator, it is often useful to select a node for this purpose to improve network performance. A node which is centrally located in the network can serve as a good sleep coordinator to minimize the number of hops a sync message must take to get across the network. A sleep support node and/or a node which is mains powered may be a good candidate.

The preferred sleep coordinator bit should be used with caution. The advantages of using the option become weaknesses when used on a node that is not positioned or configured properly. The preferred sleep coordinator option can also be used when setting up a network for the first time. When starting a network, a node can be configured as a sleep coordinator so it will begin sending sleep messages. After the network is set up, the preferred sleep coordinator bit can be disabled.

Nomination and Election

Nomination is an optional process that can occur on a node in the event that contact with the network sleep coordinator is lost. By default, this behavior is disabled. This behavior can be enabled with the sleep options command (SO). This process will automatically occur in the event that contact with the previous sleep coordinator is lost. Any sleep compatible node which has this behavior enabled is eligible to become the sleep coordinator for the network. If a sleep compatible node has missed three or more sync messages and is not configured as a non-sleep coordinator (presumably because the sleep coordinator has been disabled) it may become a sleep coordinator. Depending on the platform and other configured options, such a node will eventually nominate itself after a number of cycles without a sync. A nominated node will begin acting as the new network sleep coordinator. It is possible for multiple nodes to nominate themselves as the sleep coordinator. If this occurs, an election will take place to establish seniority among the multiple sleep coordinators. Seniority is determined by four factors (in order of priority):

1. Newer sleep parameters: a node using newer sleep parameters (SP/ST) is considered senior to a node using older sleep parameters. (See the Changing Sleep Parameters section below.)
2. Preferred Sleep Coordinator: a node acting as a preferred sleep coordinator is senior to other nodes.
3. Sleep Support Node: sleep support nodes are senior to cyclic sleep nodes. (This behavior can be modified using the SO parameter.)
4. Serial number: in the event that the above factors do not resolve seniority, the node with the higher serial number is considered senior.

Commissioning Button

The commissioning button can be used to select a module to act as the sleep coordinator. If the commissioning button functionality has been enabled, a node can be immediately nominated as a sleep coordinator by pressing the commissioning button twice or by issuing the CB2 command. A node nominated in this manner is still subject to the election process described above. A node configured as a non-sleep coordinator will ignore commissioning button nomination requests.

Changing Sleep Parameters

Any sleep compatible node in the network which does not have the non-sleep coordinator sleep option set can be used to make changes to the network's sleep and wake times. If a node's SP and/or ST are changed to values different from those that the network is using, that node will become the sleep coordinator. That node will begin sending sync messages with the new sleep parameters at the beginning of the next wake cycle.

Note #1: For normal operations, a module will use the sleep and wake parameters it gets from the sleep sync message, not the ones specified in its SP and ST parameters. The SP and ST parameters are not updated with the values of the sync message. The current network sleep and wake times used by the node can be queried using the OS and OW commands.

Note #2: Changing network parameters can cause a node to become a sleep coordinator and change the sleep settings of the network. The following commands can cause this to occur: NH, NN, NQ, and MR. In most applications, these network parameters should only be configured during deployment.

Sleep Guard Times

To compensate for variations in the timekeeping hardware of the various modules in a sleeping router network, sleep guard times are allocated at the beginning and end of the wake time. The size of the sleep guard time varies based on the sleep and wake times selected and the number of cycles that have elapsed since the last sync message was received. The sleep guard time guarantees that a destination radio will be awake when a transmission is sent. As more and more consecutive sync messages are missed, the sleep guard time increases in duration and decreases the available transmission time.

Auto-Early Wake-Up Sleep Option

Similarly to the sleep guard time, the auto early wake-up option decreases the sleep period based on the number of sync messages missed. This option comes at the expense of battery life. Auto-early wake-up sleep can be disabled using the sleep options (SO) command.

Configuration

Selecting Sleep Parameters

Choosing proper sleep parameters is vital to creating a robust sleep-enabled network with a desirable battery life. To select sleep parameters that will be good for most applications, follow these steps:

1. **Choose NH.** Based on the placement of the nodes in your network, select appropriate values for the Network Hops (NH) parameter.

Note: the default value of NH has been optimized to work for the majority of deployments. In most cases, we suggest that the parameter not be modified from its default value. Decreasing its parameters for small networks can improve battery life, but care should be taken so that the value is not made too small.

2. **Determine the Sync Message Propagation Time (SMPT).** This is the maximum amount of time it takes for a sleep synchronization message to propagate to every node in the network. This number is the BroadcastTxTime described in the "Transmission Timeouts" section of Chapter 3.

3. **Select desired duty cycle.** The ratio of sleep time to wake time is the factor that has the greatest effect on the RF module's power consumption. Battery life can be estimated based on the following factors: sleep period, wake time, sleep current, RX current, TX current, and battery capacity.

4. **Choose sleep period and wake time.** The wake time needs to be long enough to transmit the desired data as well as the sync message. The ST parameter will automatically adjust upwards to its minimum value when other AT commands are changed that will affect it (SP, and NH). Use a value larger than this minimum. If a module misses successive sync messages, it reduces its available transmit time to compensate for possible clock drift. Budget a large enough ST time to allow for a few sync messages to be missed and still have time for normal data transmissions.

Starting a Sleeping Network

By default, all new nodes operate in normal (non-sleep) mode. To start a sleeping network, follow these steps:

1. Enable the preferred sleep coordinator option on one of the nodes, and set its SM to a sleep compatible mode (7 or 8) with its SP and ST set to a quick cycle time. The purpose of a quick cycle time is to allow commands to be sent quickly through the network during commissioning.
2. Next, power on the new nodes within range of the sleep coordinator. The nodes will quickly receive a sync message and synchronize themselves to the short cycle SP and ST.
3. Configure the new nodes in their desired sleep mode as cyclic sleeping nodes or sleep support nodes.
4. Set the SP and ST values on the sleep coordinator to the desired values for the deployed network.
5. Wait a cycle for the sleeping nodes to sync themselves to the new SP and ST values.
6. Disable the preferred sleep coordinator option bit on the sleep coordinator (unless a preferred sleep coordinator is desired).

7. Deploy the nodes to their positions.

Alternatively, nodes can be set up with their sleep pre-configured and written to flash (using the WR command) prior to deployment. If this is the case, the commissioning button and associate LED can be used to aid in deployment:

1. If a preferred sleep coordinator is going to be used in the network, deploy it first. If there will be no preferred sleep coordinator, select a node for deployment, power it on and press the commissioning button twice. This will cause the node to begin emitting sync messages.

Verify that the first node is emitting sync messages by watching its associate LED. A slow blink indicates that the node is acting as a sleep coordinator.

2. Next, power on nodes in range of the sleep coordinator or other nodes which have synchronized with the network. If the synchronized node is asleep, it can be woken by pressing the commissioning button once.

3. Wait a cycle for the new node to sync itself.

4. Verify that the node syncs with the network. The associate LED will blink when the module is awake and synchronized.

5. Continue this process until all nodes have been deployed.

Adding a New Node to an Existing Network

To add a new node to the network, the node must receive a sync message from a node already in the network. On power-up, an unsynchronized sleep compatible node will periodically send a broadcast requesting a sync message and then sleep for its SP period. Any node in the network that receives this message will respond with a sync. Because the network can be asleep for extended periods of time, and as such cannot respond to requests for sync messages, there are methods that can be used to sync a new node while the network is asleep.

1. Power the new node on within range of a sleep support node. Sleep support nodes are always awake and will be able to respond to sync requests promptly.

2. A sleeping cyclic sleep node in the network can be woken by the commissioning button. Place the new node in range of the existing cyclic sleep node and wake the existing node by holding down the commissioning button for 2 seconds, or until the node wakes. The existing node stays awake for 30 seconds and will respond to sync requests while it is awake.

If you do not use one of these two methods, you must wait for the network to wake up before adding the new node. The new node should be placed in range of the network with a sleep/wake cycle that is shorter than the wake period of the network. The new node will periodically send sync requests until the network wakes up and it receives a sync message.

Changing Sleep Parameters

Changes to the sleep and wake cycle of the network can be made by selecting any node in the network and changing the SP and/or ST of the node to values different than those the network is currently using. If using a preferred sleep coordinator or if it is known which node is acting as the sleep coordinator, it is suggested that this node be used to make changes to network settings. If the network sleep coordinator is not known, any node that does not have the non-sleep coordinator sleep option bit set (see the SO command) can be used.

When changes are made to a node's sleep parameters, that node will become the network's sleep coordinator (unless it has the non-sleep coordinator option selected) and will send a sync message with the new sleep settings to the entire network at the beginning of the next wake cycle. The network will immediately begin using the new sleep parameters after this sync is sent.

Changing sleep parameters increases the chances that nodes will lose sync. If a node does not receive the sync message with the new sleep settings, it will continue to operate on its old settings. To minimize the risk of a node losing sync and to facilitate the re-syncing of a node that does lose sync, the following precautions can be taken:

1. Whenever possible, avoid changing sleep parameters.

2. Enable the missed sync early wake up sleep option (SO). This command is used to tell a node to wake up progressively earlier based on the number of cycles it has gone without receiving a sync. This will increase the probability that the un-synced node will be awake when the network wakes up and sends the sync message.

Note: using this sleep option increases reliability but may decrease battery life. Nodes using this sleep option which miss sync messages will have an increased wake time and decreased sleep time during cycles in which the sync message is missed. This will reduce battery conservation.

3. When changing between two sets of sleep settings, choose settings so that the wake periods of the two sleep settings will happen at the same time. In other words, try to satisfy the following equation: $(SP1 + ST1) = N * (SP2 + ST2)$, where $SP1/ST1$ and $SP2/ST2$ are the desired sleep settings and N is an integer.

Rejoining Nodes Which Have Lost Sync

Mesh networks get their robustness from taking advantage of routing redundancies which may be available in a network. It is recommended to architect the network with redundant mesh nodes to increase robustness. If a scenario exists such that the only route connecting a subnet to the rest of the network depends on a single node, and that node fails -- or the wireless link fails due to changing environmental conditions (catastrophic failure condition), then multiple subnets may arise while using the same wake and sleep intervals. When this occurs the first task is to repair, replace, and strengthen the weak link with new and/or redundant modules to fix the problem and prevent it from occurring in the future.

When the default DigiMesh sleep parameters are used, separated subnets will not drift out of phase with each other. Subnets can drift out of phase with each other if the network is configured in one of the following ways:

- If multiple modules in the network have had the non-sleep coordinator sleep option bit disabled and are thus eligible to be nominated as a sleep coordinator.
- If the modules in the network are not using the auto early wake-up sleep option.

If a network has multiple subnets that have drifted out of phase with each other, get the subnets back in phase with the following steps:

1. Place a sleep support node in range of both subnets.
2. Select a node in the subnet that you want the other subnet to sync up with. Use this node to slightly change the sleep cycle settings of the network (increment ST , for example).
3. Wait for the subnet's next wake cycle. During this cycle, the node selected to change the sleep cycle parameters will send the new settings to the entire subnet it is in range of, including the sleep support node which is in range of the other subnet.
4. Wait for the out of sync subnet to wake up and send a sync. When the sleep support node receives this sync, it will reject it and send a sync to the subnet with the new sleep settings.
5. The subnets will now be in sync. The sleep support node can be removed. If desired, the sleep cycle settings can be changed back to what they were.

In the case that only a few nodes need to be replaced, this method can also be used:

1. Reset the out of sync node and set its sleep mode to cyclic sleep ($SM = 8$). Set it up to have a short sleep cycle.
2. Place the node in range of a sleep support node or wake a sleeping node with the commissioning button.
3. The out of sync node will receive a sync from the node which is synchronized to the network and sync to the network sleep settings.

Diagnostics

The following are useful in some applications when managing a sleeping router network:

Query current sleep cycle: the OS and OW commands can be used to query the current operational sleep and wake times a module is currently using.

Sleep Status: the SS command can be used to query useful information regarding the sleep status of the module. This command can be used to query if the node is currently acting as a network sleep coordinator, as well as other useful diagnostics.

Missed Sync Messages Command: the MS command can be used to query the number of cycles that have elapsed since the module last received a sync message.

Sleep Status API messages: when enabled with the SO command, a module configured in API mode will output modem status frames immediately after a module wakes up and just prior to a module going to sleep.

5. Command Reference Tables

Special

Table 5-01. Special Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-----------------|---------|
| AC | Apply Changes. Immediately applies new settings without exiting command mode. | -- | -- |
| FR | Software Reset. Reset module. Responds immediately with an "OK" then performs a reset 100ms later. | -- | -- |
| RE | Restore Defaults. Restore module parameters to factory defaults. | -- | -- |
| WR | Write. Write parameter values to non-volatile memory so that parameter modifications persist through subsequent resets. Note: Once WR is issued, no additional characters should be sent to the module until after the "OK\r" response is received. | -- | -- |

MAC/PHY Level

Table 5-02. MAC/PHY-level Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|--|----------------------------------|---|
| AF | <p>Available Frequencies. This read only command can be queried to return a bitfield of the frequencies that are available in the module's region of operation.</p> <p>This command returns a bitfield. Each bit corresponds to a physical channel. Channels are spaced 400 kHz apart:</p> <p>Bit 0 – 902.400 MHz Bit 1 – 902.800 MHz . . Bit 31 – 914.800 MHz . . Bit 63 – 927.600 MHz</p> | 0x1FFFFFFF – 0x00FFFFFFFFFFFFFFF | <p>USA/Canada: 0x00FFFFFFF FFFFFFFFF (channels 0 – 63)</p> <p>Australia: 0x00FFFFFF FE0000000 (channels 33 – 63)</p> <p>Brazil: 0x00FFFFFF FE0000FFF (channels 0-11, 33 – 63)</p> |
| CM | <p>Channel Mask. The channel mask command allows channels to be selectively enabled or disabled. This is useful to avoid using frequencies that experience unacceptable levels of RF interference.</p> <p>This command is a bitfield. Each bit in the bitfield corresponds to a frequency as defined in the Available Frequencies (AF) command. When a bit in the Channel Mask and the corresponding bit in the Available Frequencies are both set to 1 then that physical channel may be chosen by the module as an active channel for communication.</p> <p>A minimum of 25 channels must be made available for the module to communicate on. The module will choose the 25 lowest enabled frequencies as its active channels if more than 25 are enabled.</p> <p>All modules in a network must use an identical set of active channels. Separate networks which are in physical range of each other should use different Preamble Patterns (HP) and/or Network ID's (ID) to avoid receiving data from the other network.</p> <p>The user may find the Energy Detect (ED) command especially useful when choosing what channels to enable or disable.</p> <p>Note that channel 19 (910.000MHz) is disabled by default. This channel has approximately 2dBm worse receiver sensitivity than other channels. It is suggested that this channel not be used.</p> | 0x1FFFFFFF – 0x00FFFFFFFFFFFFFFF | 0xFFFFFFFF FFF7FFFF |
| HP | Preamble ID. The preamble ID for which module communicates. Only modules with matching preamble IDs can communicate with each other. Different preamble IDs minimize interference between multiple sets of modules operating in the same vicinity. When receiving a packet this is checked before the network ID, as it is encoded in the preamble, and the network ID is encoded in the MAC header. | 0-7 | 0 |

| AT Command | Name and Description | Parameter Range | Default |
|------------|--|--|---------|
| ID | Network ID. The user network identifier. Nodes must have the same network identifier to communicate. Changes to ID can be written to non-volatile memory using the WR command. Only modules with matching IDs can communicate with each other. When receiving a packet this is checked after the preamble ID. If using OEM network IDs, 0xFFFF will use the factory value. | 0-0x7FFF | 0x7FFF |
| MT | Broadcast Multi-Transmit. The number of additional MAC-level broadcast transmissions. All broadcast packets are transmitted MT+1 times to ensure it is received. | 0-5 | 3 |
| PL | Power Level. Set/Read the power level at which the RF module transmits conducted power. Power level 4 is calibrated and the other power levels are approximate. . | 0 = +7 dBm, (5 mW) 1 = +15 dBm, (32 mW) 2 = +18 dBm, (63 mW) 3 = +21 dBm, (125 mW) 4 = +24 dBm, (250 mW) | 4 |
| RR | Unicast Mac Retries. The maximum number of MAC level packet delivery attempts for unicasts. If RR is non-zero packets sent from the radio will request an acknowledgement, and can be resent up to RR times if no acknowledgements are received. | 0-0xF | 0x10 |
| ED | Energy Detect. Start an Energy Detect scan. This parameter is the time in milliseconds to scan all channels. The module will loop through all the channels until the time elapses. The maximal energy on each channel is returned, and each value is followed by a comma with the list ending with a carriage return. The values returned reflect the detected energy level in units of -dBm. | 0-0xFF | 0x10 |

Diagnostics

Table 5-03. Diagnostics Commands - MAC Statistics and Timeouts

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-----------------------|---------|
| BC | Bytes Transmitted. The number of RF bytes transmitted. This count is incremented for every PHY level byte transmitted. The purpose of this count is to estimate battery life by tracking time doing transmissions. This number rolls over to zero from 0xFFFF. The counter can be reset to any 16-bit value by appending a hexadecimal parameter to the command. | 0-0xFFFF | 0 |
| DB | Received Signal Strength. This command reports the received signal strength of the last received RF data packet. The DB command only indicates the signal strength of the last hop. It does not provide an accurate quality measurement for a multihop link. The DB command value is measured in -dBm. For example if DB returns 0x60, then the RSSI of the last packet received was -96dBm. | 0-0xFF [read-only] | 0 |
| ER | Received Error Count. This count is incremented whenever a packet is received which contained integrity errors of some sort. Once the number reaches 0xFFFF, further events will not be counted. The counter can be reset to any 16-bit value by appending a hexadecimal parameter to the command. | 0-0xFFFF | 0 |
| GD | Good Packets Received. This count is incremented whenever a good frame with a valid MAC header is received on the RF interface. Once the number reaches 0xFFFF, further events will not be counted. The counter can be reset to any 16-bit value by appending a hexadecimal parameter to the command. | 0-0xFFFF | 0 |
| EA | MAC ACK Timeouts. This count is incremented whenever a MAC ACK timeout occurs on a MAC level unicast. Once the number reaches 0xFFFF further events will not be counted. The counter can be reset to any 16-bit value by appending a hexadecimal parameter to the command. | 0-0xFFFF | 0 |
| TR | Transmission Errors. This count is incremented whenever a MAC transmission attempt exhausts all MAC retries without ever receiving a MAC acknowledgement message from the destination node. Once the number reaches 0xFFFF, further events will not be counted. The counter can be reset to any 16-bit value by appending a hexadecimal parameter to the command. | 0-0xFFFF | 0 |
| UA | MAC Unicast Transmission Count. This count is incremented whenever a MAC unicast transmission occurs for which an ACK is requested. Once the number reaches 0xFFFF further events will not be counted. The counter can be reset to any 16-bit value by appending a hexadecimal parameter to the command. | 0-0xFFFF | 0 |
| %H | MAC Unicast One Hop Time. The MAC unicast one hop timeout in milliseconds. Changing MAC parameters can change this value. | [read-only] | 0xCF |
| %8 | MAC Broadcast One Hop Time. The MAC broadcast one hop timeout in milliseconds. Changing MAC parameters can change this value. | [read-only] | 0x1BE |

Network**Table 5-04. Network Commands - DigiMesh and Repeater**

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-----------------|---------|
| CE | <p>Node Messaging Options. The module's routing and messaging mode bit field. A routing module will repeat broadcasts. Indirect Messaging Coordinators will not transmit point-to-multipoint unicasts until they are requested by an Indirect Messaging Poller. Setting a radio as an Indirect Messaging Poller will cause it to regularly send polls to its Indirect Messaging Coordinator. Nodes can also be configured to route, or not route, multi-hop packets.</p> <p>Bit 0 - Indirect Messaging Coordinator enable All point-multipoint unicasts will be held until requested by a polling end device.</p> <p>Bit 1 - Disable routing on this node When set, this node will not propagate broadcasts or become an intermediate node in a DigiMesh route. This node will not function as a repeater.</p> <p>Bit 2 - Indirect Messaging Polling enable Periodically send requests for messages held by the node's coordinator.</p> <p>Bit 0 and bit 2 cannot be set at the same time.</p> | 0-6 | 0 |
| BH | <p>Broadcast Hops. The transmission hops for broadcast data transmissions. Set to 0 for maximum radius. If BH is set greater than NH then the value of NH is used. Supported in both variants.</p> | 0-0x20 | 0 |
| NH | <p>Network Hops The maximum number of hops expected to be seen in a network route. This value doesn't limit the number of hops allowed, but it is used to calculate timeouts waiting for network acknowledgements. Supported in both variants.</p> | 0-0x20 | |
| NN | <p>Network Delay Slots. Set or read the maximum random number of network delay slots before rebroadcasting a network packet.</p> | 0 to 0x05 | 3 |
| MR | <p>Mesh Unicast Retries The maximum number of network packet delivery attempts. If MR is non-zero, packets sent will request a network acknowledgement, and can be resent up to MR+1 times if no acknowledgements are received. We recommend setting this value to 1. If this parameter is set to 0, then network ACKs are disabled. Routes can be found initially, but will never be repaired if a route fails. Supported in the 200k variant only.</p> | 0 to 7 | 1 |

Addressing**Table 5-05. Addressing Commands**

| AT Command | Name and Description | | | Parameter Range | Default |
|------------|--|--|---|--|------------------------|
| SH | Serial Number High. The upper 32 bits of the module's unique IEEE 64-bit MAC address. | | | 0-0xFFFFFFFF [read-only] | Factory |
| SL | Serial Number Low. The lower 32 bits of the module's unique IEEE 64-bit MAC address. | | | 0-0xFFFFFFFF [read-only] | Factory |
| DH | Destination Address High. The upper 32 bits of the 64-bit destination address. When combined with DL, it defines the destination address used for transmission in transparent mode. | | | 0-0xFFFFFFFF | 0 |
| DL | Destination Address Low. The lower 32 bits of the 64-bit destination address. When combined with DH, DL defines the destination address used for transmission in transparent mode. | | | 0-0xFFFFFFFF | 0x0000FFFF |
| TO | Transmit Options. This command defines transmission options for all packets originating from this radio. These options can be overridden on a packet-by-packet basis by using the TxOptions field of the API TxRequest frames. | | | | |
| | Bit | Meaning | Description | | |
| | 6, 7 | Delivery method | b'00 - <invalid option>. b'01 - Point-Multipoint b'10 - Repeater mode (directed broadcast of packets) b'11 - DigiMesh (not available on 10k product) | Bits 6 & 7 cannot be set to DigiMesh on the 10k build. | 0x40 (10k product) |
| | 5 | Reserved | <set this bit to 0> | Bits 4 & 5 must be set to 0 | |
| | 4 | Reserved | <set this bit to 0> | | 0xC0 (200k product) |
| | 3 | Trace Route | Enable a Trace Route on all DigiMesh API packets | Bits 1, 2, & 3 cannot be set on the 10k build | |
| | 2 | NACK | Enable a NACK messages on all DigiMesh API packets | | |
| | 1 | Disable RD | Disable Route Discovery on all DigiMesh unicasts | | |
| | 0 | Disable ACK | Disable acknowledgments on all unicasts | | |
| | | Example #1: Setting TO to 0x80 would cause all transmissions to be sent using repeater mode. | | | |
| | Example #2: Setting TO to 0xC1 would cause all transmissions to be sent using DigiMesh, with network acknowledgments disabled. | | | | |

Table 5-05. Addressing Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-------------------------------|-------------------|
| NI | Node Identifier. A string identifier for this module. The string accepts only printable ASCII data. In AT Command Mode, the string can not start with a space. A carriage return or comma ends the command. Command will automatically end when maximum bytes for the string have been entered. This string is returned as part of the ATND (Network Discover) command. This identifier is also used with the ATDN (Destination Node) command. | up to 20 byte ASCII string | a space character |
| NT | Node Discover Timeout. The amount of time a node will spend discovering other nodes when ND or DN is issued. This value is used to randomize the responses to alleviate network congestion. | 0x20 - 0x2EE0 [x 100 msec] | 0x82 (130d) |
| NO | Node Discovery Options. The options value for the network discovery command. The options bitfield value can change the behavior of the ND (network discovery) command and/or change what optional values are returned in any received ND responses or API node identification frames. Options include: 0x01 = Append DD value (to ND responses or API node identification frames) 0x02 = Local device sends ND or FN response frame when ND is issued. 0x04 = Append RSSI (of the last hop for DigiMesh networks) to ND or FN responses or API node identification frames. | 0-0x07 [bitfield] | 0 |
| CI | Cluster ID. The application layer cluster ID value. This value will be used as the cluster ID for all data transmissions. The default value 0x11 (Transparent data cluster ID) | 0-0xFFFF | 0x11 |
| DE | Destination Endpoint. The application layer destination ID value. This value will be used as the destination endpoint for all data transmissions. The default value (0xE8) is the Digi data endpoint. | 0-0xFF | 0xE8 |
| SE | Source Endpoint. The application layer source endpoint value. This value will be used as the source endpoint for all data transmissions. The default value 0xE8 (Data endpoint) is the Digi data endpoint | 0-0xFF | 0xE8 |

Addressing Discovery/Configuration

Table 5-06. Addressing Discovery/Configuration Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|--|----------------------|---------|
| AG | Aggregator Support. The AG command sends a broadcast through the network that has the following effects on nodes which receive the broadcast: <ul style="list-style-type: none"> - The receiving node will establish a DigiMesh route back to the originating node, provided there is space in the routing table. - The DH and DL of the receiving node will be updated to the address of the originating node if the AG parameter matches the current DH/DL of the receiving node. - For API-enabled modules on which DH and DL are updated, an Aggregate Addressing Update frame will be sent out the serial port. <p>Note that the AG command is only available on products that support DigiMesh.</p> | Any 64-bit number | n/a |
| DN | Discover Node. Resolves an NI (Node Identifier) string to a physical address (case sensitive). The following events occur after the destination node is discovered: <p><AT Firmware></p> <ol style="list-style-type: none"> 1. DL & DH are set to the extended (64-bit) address of the module with the matching NI (Node Identifier) string. 2. OK (or ERROR)\r is returned. 3. Command Mode is exited to allow immediate communication <p><API Firmware></p> <p>0xFFFFE and 64-bit extended addresses are returned in an API Command Response frame.</p> <p>If there is no response from a module within (NT * 100) milliseconds or a parameter is not specified (left blank), the command is terminated and an "ERROR" message is returned. In the case of an ERROR, Command Mode is not exited.</p> | 20 byte ascii string | |

Table 5-06. Addressing Discovery/Configuration Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-----------------|---------|
| ND | <p>Network Discover. Discovers and reports all RF modules found. The following information is reported for each module discovered.</p> <p>MY<CR> (always 0xFFFE) SH<CR> SL<CR> NI<CR> (Variable length) PARENT_NETWORK ADDRESS<CR> (2 Bytes) (always 0xFFFE) DEVICE_TYPE<CR> (1 Byte: 0=Coord, 1=Router, 2=End Device) STATUS<CR> (1 Byte: Reserved) PROFILE_ID<CR> (2 Bytes) MANUFACTURER_ID<CR> (2 Bytes) DIGI DEVICE TYPE<CR> (4 Bytes. Optionally included based on NO settings.) RSSI OF LAST HOP<DR> (1 Byte. Optionally included based on NO settings.) <CR></p> <p>After (NT * 100) milliseconds, the command ends by returning a <CR>. ND also accepts a Node Identifier (NI) as a parameter (optional). In this case, only a module that matches the supplied identifier will respond.</p> <p>If the ND command is sent through a local API frame, each response is returned as a separate Local or Remote AT Command Response API packet, respectively. The data consists of the above listed bytes without the carriage return delimiters. The NI string will end in a "0x00" null character.</p> | n/a | n/a |
| FN | <p>Find Neighbors. Discovers and reports all RF modules found within immediate RF range. The following information is reported for each module discovered.</p> <p>MY<CR> (always 0xFFFE) SH<CR> SL<CR> NI<CR> (Variable length) PARENT_NETWORK ADDRESS<CR> (2 Bytes) (always 0xFFFE) DEVICE_TYPE<CR> (1 Byte: 0=Coord, 1=Router, 2=End Device) STATUS<CR> (1 Byte: Reserved) PROFILE_ID<CR> (2 Bytes) MANUFACTURER_ID<CR> (2 Bytes) DIGI DEVICE TYPE<CR> (4 Bytes. Optionally included based on NO settings.) RSSI OF LAST HOP<DR> (1 Byte. Optionally included based on NO settings.) <CR></p> <p>If the FN command is issued in command mode, after (NT*100) ms + overhead time, the command ends by returning a <CR>.</p> <p>If the FN command is sent through a local API frame, each response is returned as a separate Local or Remote AT Command Response API packet, respectively. The data consists of the above listed bytes without the carriage return delimiters. The NI string will end in a "0x00" null character.</p> | n/a | n/a |

Security

Table 5-07. Security Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|--|-----------------|---------|
| EE | Security Enable Enables or disables 128-bit AES encryption. This command parameter must be set the same on all devices for communication to work. | 0-1 | 0 |
| KY | AES Encryption Key Sets the 16 byte network security key value. This command is write-only; it cannot be read. Attempts to read KY will return an OK status. This command parameter must be set the same on all devices for communication to work. This value is passed in as hex characters when setting from AT command mode, and as binary bytes when set in AT! mode. | 128-bit value | n/a |

Serial Interfacing**Table 5-08. Serial Interfacing Commands**

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|---------------------------------|-----------------|
| BD | Baud rate. The UART baud rate (speed for data transfer between radio modem and host). Values from 0-8 select preset standard rates. Values at 0x39 and above select the actual baud rate. Providing the host supports it. Baud rates can go as high as 7Mbps. The values from 0 to 8 are interpreted as follows: 0 - 1,200bps 3 - 9,600bps 6 - 57,600bps 1 - 2,400bps 4 - 19,200bps 7 - 115,200bps 2 - 4,800bps 5 - 38,400bps 8 - 230,400bps | 0 to 8, and 0x100 to 0x6ACFC0 | 0x03 (9600 bps) |
| NB | Parity. Set or read parity settings for UART communications. The values from 0 to 2 are interpreted as follows: 0 No parity 1 Even parity 2 Odd parity | 0-2 | 0 (No parity) |
| SB | Stop Bits. The number of stop bits for the UART. 0 - One stop bit 1 - Two stop bits | 0-1 | 0 |
| RO | Packetization Timeout. The number of UART character times of inter-character silence required before packetization in transparent mode. Set (RO=0) to transmit characters as they arrive instead of buffering them into one RF packet. | 0 - 0xFF [x character times] | 3 |
| FT | Flow Control Threshold. The UART flow control threshold. De-assert CTS and/or send XOFF when FT bytes are in the UART receive buffer. Re-assert CTS when less than FT - 16 bytes are in the UART receive buffer. | 0x11 - 0x16F | 0x13F |
| AP | API mode. The UART API mode. The following settings are allowed: 0 Transparent mode, API mode is off. All UART input and output is raw data and packets are delineated using the RO and RB parameters. 1 API mode without escapes is on. All UART input and output data is packetized in the API format. 2 API mode is on with escaped sequences inserted to allow for control characters (XON, XOFF, escape, and the 0x7e delimiter to be passed as data.) | 0- 2 | 0 |
| AO | API Options. The API data frame output format for received frames. This parameter applies to both the UART and SPI interfaces. 0 API RX Indicator (0x90) 1 API Explicit RX Indicator (0x91) | 0, 1 | 0 |

I/O Settings**Table 5-09. I/O Settings and Commands**

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-----------------|---------|
| CB | Commissioning Pushbutton. This command can be used to simulate commissioning button presses in software. The parameter value should be set to the number of button presses to be simulated. For example, sending the ATCB1 command will execute the action associated with 1 commissioning button press. | 0-4 | n/a |
| D0 | DIO0 / AD0 Configuration (Pin 20). 0 = Disabled 1 = Commissioning button 2 = ADC 3 = Digital input 4 = Digital output low 5 = Digital output high | 0 - 5 | 1 |
| D1 | DIO1 / AD1 Configuration (Pin 19). 0 = Disabled 1 = SPI Attention 2 = ADC 3 = Digital input 4 = Digital output low 5 = Digital output high 6 = Uart Data Present Indicator | 0-6 | 0 |
| D2 | DIO2 / AD2 Configuration (Pin 18). 0 = Disabled 1 = SPI Clock 2 = ADC 3 = Digital input 4 = Digital output low 5 = Digital output high | 0-5 | 0 |

Table 5-09. I/O Settings and Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|--|-----------------|---------|
| D3 | DIO3 / AD3 Configuration (Pin 17). 0 = Disabled 1 = SPI Slave Select 2 = ADC 3 = Digital input 4 = Digital output low 5 = Digital output high | 0-5 | 0 |
| D4 | DIO4 Configuration (Pin 11). 0 = Disabled 1 = SPI_MOSI 3 = Digital input 4 = Digital output low 5 = Digital output high | 0, 1, 3-5 | 0 |
| D5 | DIO5 / ASSOCIATE_INDICATOR Configuration (Pin 15). 0 = Disabled 1 = Associated Indicator 3 = Digital input 4 = Digital output low 5 = Digital output high | 0, 1, 3-5 | 1 |
| D6 | DIO6 / RTS Configuration (Pin 16). 0 = Disabled 1 = RTS flow control 3 = Digital input 4 = Digital output low 5 = Digital output high | 0, 1, 3-5 | 0 |
| D7 | DIO7 / CTS Configuration (Pin 12). 0 = Disabled 1 = CTS flow control 3 = Digital input 4 = Digital output low 5 = Digital output high 6 = RS-485 Tx enable, low TX (0V on transmit, high when idle) 7 = RS-485 Tx enable, high TX (high on transmit, 0V when idle) | 0, 1, 3-7 | 1 |
| D8 | DIO8 / SLEEP_REQUEST Configuration (Pin 9). 0 = Disabled 1 = Sleep request 3 = Digital input 4 = Digital output low 5 = Digital output high | 0, 1, 3-5 | 1 |
| D9 | DIO9 / ON/SLEEP Configuration. (Pin 13) 0 = Disabled 1 = ON/SLEEP output 3 = Digital input 4 = Digital output low 5 = Digital output high | 0, 1, 3-5 | 1 |
| P0 | DIO10 / RSSI / PWM0 Configuration (Pin 6). 0 = Disabled 1 = RSSI PWM0 output 2 = PWM0 output 3 = Digital input 4 = Digital output low 5 = Digital output high | 0-5 | 1 |
| P1 | DIO11 / PWM1 Configuration (Pin 7). 0 = Disabled 1 = 32.768 kHz clock output 2 = PWM1 output 3 = Digital input 4 = Digital output low 5 = Digital output high | 0, 2-5 | 0 |

Table 5-09. I/O Settings and Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|---------------------|---------------------|
| P2 | DIO12 Configuration (Pin 4). 0 = Disabled 1 = SPI_MISO 3 = Digital input 4 = Digital output low 5 = Digital output high | 0, 1, 3-5 | 0 |
| P3 | DIO13 / DOUT Configuration (Pin 2). 0 = Disabled 1 = UART DOUT output | 0, 1 | 1 |
| P4 | DIO14 / DIN Configuration (Pin 3). 0 = Disabled 1 = UART DIN output | 0, 1 | 1 |
| PD | Pull Direction. The resistor pull direction bit field for corresponding I/O lines that are set in the PR command. 0 = pull down 1 = pull up | 0-0x7FFF | 0 |
| PR | Pull-up Resistor. The bit field that configures the internal pull-up resistor status for the I/O lines. "1" specifies the pull-up/down resistor is enabled. "0" specifies no pullup/down. Bits: 0 - DIO4 / AD4 / SPI_MOSI 1 - DIO3 / AD3 / SPI_SSSEL 2 - DIO2 / AD2 / SPI_SCLK 3 - DIO1 / AD1 / SPI_ATTEN 4 - DIO0 / AD0 5 - DIO6 / RTS 6 - SLEEP_REQUEST 7 - DIN / CONFIG 8 - DIO5 / AD5 / ASSOCIATE 9 - On/SLEEP 10 - DIO12 / SPI_MISO 11 - DIO10 / PWM0 / RSSI 12 - DIO11 / PWM1 13 - DIO7/CTS 14 - PWM0 / DOUT | 0 - 0x7FFF | 0x7FFF |
| M0 | PWM0 Duty Cycle. The duty cycle of the PWM0 line. The line should be configured as a PWM output using the P0 command. | 0-0x3FF | 0 |
| M1 | PWM1 Duty Cycle. The duty cycle of the PWM1 line. The line should be configured as a PWM output using the P1 command. | 0-0x3FF | 0 |
| LT | Assoc LED Blink Time. The Associate LED blink time. If the Associate LED functionality is enabled (D5 command), this value determines the on and off blink times for the LED. If LT=0, the default blink rate will be used (500ms sleep coordinator, 250ms otherwise). For all other LT values, LT is measured in 10ms | 0x14-0xFF [x 10 ms] | 0 |
| RP | RSSI PWM Timer. Time RSSI signal will be output after last transmission. When RP = 0xFF, output will always be on. | 0 - 0xFF [x 100 ms] | 0x28 (4 seconds) |

I/O Sampling

Table 5-010. I/O Sampling Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-----------------|---------|
| AV | Analog Voltage Reference. The analog voltage reference that is used for A/D sampling. 0 = 1.25 V reference 1 = 2.5 V reference | 0, 1 | 0 |

Table 5-010. I/O Sampling Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-----------------|---------|
| IC | DIO Change Detection. The digital I/O pins to monitor for changes in the I/O state. IC works with the individual pin configuration commands (D0-D9, P0-P2). If a pin is enabled as a digital input/output, the IC command can be used to force an immediate I/O sample transmission when the DIO state changes. IC is a bitmask that can be used to enable or disable edge detection on individual channels. Unused bits should be set to 0. Bit (I/O pin): 0 (DIO0) 1 (DIO1) 2 (DIO2) 3 (DIO3) 4 (DIO4) 5 (DIO5) 6 (DIO6) 7 (DIO7) 8 (DIO8) 9 (DIO9) 10 (DIO10) 11 (DIO11) 12 (DIO12) | 0-0xFFFF | 0 |
| IF | Sleep Sample Rate. The number of sleep cycles that must elapse between periodic I/O samples. This allows I/O samples to be taken only during some wake cycles. During those cycles I/O samples are taken at the rate specified by IR. | 1-0xFF | 1 |
| IR | IO Sample Rate. The I/O sample rate to enable periodic sampling. For periodic sampling to be enabled, IR must be set to a non-zero value, and at least one module pin must have analog or digital I/O functionality enabled (see D0-D9, P0-P2 commands). The sample rate is measured in milliseconds. | 0 - 0xFFFF (ms) | 0 |
| IS | Force Sample. Forces a read of all enabled digital and analog input lines. | n/a | n/a |
| %V | Supply Voltage. The supply voltage of the module in millivolts. | -- | -- |

Sleep

Sleep Commands

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|---|-----------|
| SM | Sleep Mode. The sleep mode of the module. 0 - Normal 1 - Pin sleep. In this mode, the sleep/wake state of the module is controlled by the SLEEP_REQUEST line. 4 - Asynchronous cyclic sleep. In this mode, the module periodically sleeps and wakes based on the SP and ST commands. 5 - Asynchronous cyclic sleep with pin wake-up. In this mode, the module acts in the same way as asynchronous cyclic sleep when SLEEP_RQ is asserted. When SLEEP_RQ is not asserted the module remains awake.. 7 - Sleep support mode. 8 - Synchronous cyclic sleep mode. | 0, 1, 4, 5, 7, 8 | 0 |
| SO | Sleep Options. The sleep options of the module. This command is a bitmask. For synchronous sleep modules, the following sleep options are defined: bit 0 = Preferred sleep coordinator bit 1 = Non-sleep coordinator bit 2 = Enable API sleep status messages bit 3 = Disable early wake-up bit 4 = Enable node type equality bit 5 = Disable lone coordinator sync repeat For asynchronous sleep modules, the following sleep options are defined: bit 8 = Always wake for ST time | Any of the available sleep option bits can be set or cleared. Bit 0 and bit 1 cannot be set at the same time. | 0x02 |
| SN | Number of Sleep Periods. The number of sleep periods value. This command controls the number of sleep periods that must elapse between assertions of the ON_SLEEP line during the wake time of asynchronous cyclic sleep. During cycles when the ON_SLEEP line is not asserted, the module will wake up and check for any serial or RF data. If any such data is received, then the ON_SLEEP line will be asserted and the module will fully wake up. Otherwise, the module will return to sleep after checking. This command does not work with synchronous sleep. | 1 - 0xFFFF | 1 |
| SP | Sleep Period. The sleep period of the module. This command defines the amount of time the module will sleep per cycle. For a node operating as an Indirect Messaging Coordinator, this command defines the amount of time that it will hold an indirect message for an Indirect Messaging Poller. The coordinator will hold the message for (2.5*SP). | 1 - 1440000 (x 10 ms) | 2 seconds |

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|------------------|-------------------|
| ST | Wake Time. The wake period of the module. For asynchronous sleep modules, this command defines the amount of time that the module will stay awake after receiving RF or serial data. For synchronous sleep modules, this command defines the amount of time that the module will stay awake when operating in cyclic sleep mode. This value will be adjusted upwards automatically if it is too small to function properly based on other settings. | 0x45-0x36EE80 | 0x7D0 (2 seconds) |
| WH | Wake Host. The wake host timer value. If the wake host timer is set to a non-zero value, this timer specifies a time (in millisecond units) that the device should allow after waking from sleep before sending data out the UART or transmitting an I/O sample. If serial characters are received, the WH timer is stopped immediately. When in synchronous sleep, the device will shorten its sleep period by the value specified by the WH command to ensure that it is prepared to communicate when the network wakes up. When in this sleep mode, the device will always stay awake for the WH time plus the amount of time it takes to transmit a one-hop unicast to another node. | 0-0xFFFF (x 1ms) | 0 |

Sleep Diagnostics

Table 5-011. Diagnostics - Sleep Status Timing

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-----------------|---------|
| SS | Sleep Status. The SS command can be used to query a number of Boolean values describing the status of the module. Bit 0: This bit will be true when the network is in its wake state. Bit 1: This bit will be true if the node is currently acting as a network sleep coordinator. Bit 2: This bit will be true if the node has ever received a valid sync message since the time it was powered on. Bit 3: This bit will be true if the node has received a sync message in the current wake cycle. Bit 4: This bit will be true if the user has altered the sleep settings on the module so that the node will nominate itself and send a sync message with the new settings at the beginning of the next wake cycle. Bit 5: This bit will be true if the user has requested that the node nominate itself as the sleep coordinator (using the commissioning button or the CB2 command). Bit 6 = This bit will be true if the node is currently in deployment mode. All other bits: Reserved - All non-documented bits can be any value and should be ignored. | [read-only] | 0x40 |
| OS | Operational Sleep Period. The sleep period that the node is currently using. This number will oftentimes be different from the SP parameter if the node has synchronized with a sleeping router network. Units of 10mSec | [read-only] | 0x12C |
| OW | Operational Wake Period. The wake time that the node is currently using. This number will oftentimes be different from the ST parameter if the node has synchronized with a sleeping router network. Units of 1 ms | [read-only] | 0xBB8 |
| MS | Number of Missed Syncs. The number of wake cycles that have elapsed since the last sync message was received. Supported in the 80k firmware variant only. | [read-only] | 0 |
| SQ | Missed Sync Count. Count of the number of syncs that have been missed. This value can be reset by setting ATSQ to 0. When the value reaches 0xFFFF it will not be incremented anymore. | 0-0xFFFF | 0 |

AT Command Options

Table 5-012. AT Command Options

| AT Command | Name and Description | Parameter Range | Default |
|------------|---|-----------------|---------------|
| CC | Command Character. Set or read the character to be used between guard times of the AT Command Mode Sequence. The AT Command Mode Sequence causes the radio modem to enter Command Mode (from Idle Mode). | 0 - 0xFF | 0x2B |
| CN | Exit Command Mode. Explicitly exit the module from AT Command Mode. | n/a | n/a |
| CT | Command Mode Timeout. Set/Read the period of inactivity (no valid commands received) after which the RF module automatically exits AT Command Mode and returns to Idle Mode. | 2-0x1770 | 0x64 (100d) |
| GT | Guard Times. Set required period of silence before and after the Command Sequence Characters of the AT Command Mode Sequence (GT + CC + GT). The period of silence is used to prevent inadvertent entrance into AT Command Mode. | 0 to 0xFFFF | 0x3E8 (1000d) |

Firmware Commands**Table 5-013. Firmware Version/Information**

| AT Command | Name and Description | Parameter Range | Default |
|-------------------|---|----------------------------|----------------|
| VL | Version Long. Shows detailed version information including application build date and time. | [read-only] | n/a |
| VR | Firmware Version. Read firmware version of the module. | 0 - 0xFFFFFFFF [read-only] | Firmware-set |
| HV | Hardware Version. Read hardware version of the module. | 0 - 0xFFFF [read-only] | Factory-set |
| HS | Hardware Series. The module hardware series number. For example, if the module is version S8B, this will return 0x801. | 0-0xFFFF | Factory-set |
| DD | Device Type Identifier. Stores a device type value. This value can be used to differentiate multiple XBee-based products. | 0-0xFFFFFFFF [read only] | 0xC0000 |
| NP | Maximum RF Payload Bytes. This value returns the maximum number of RF payload bytes that can be sent in a unicast transmission based on the current configurations. | 0-0xFFFF [read-only] | 0x100 |
| CK | Configuration CRC. The CRC of the current settings. The purpose of this command is to allow the detection of an unexpected configuration change on a device. After a firmware update, this command may return a different value. | | |

6. API Operation

As an alternative to Transparent Operation, API (Application Programming Interface) Operations are available. API operation requires that communication with the module be done through a structured interface (data is communicated in frames in a defined order). The API specifies how commands, command responses and module status messages are sent and received from the module using a serial data frame.

Please note that Digi may add new frame types to future versions of firmware, so please build into your software interface the ability to filter out additional API frames with unknown Frame Types.

API Frame Format

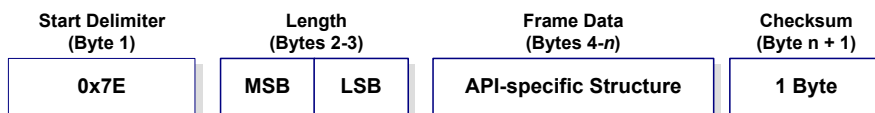
Two API modes are supported and both can be enabled using the AP (API Enable) command. Use the following AP parameter values to configure the module to operate in a particular mode:

- AP = 1: API Operation
- AP = 2: API Operation (with escaped characters--possible on UART only)

API Operation (AP parameter = 1)

When this API mode is enabled (AP = 1), the serial data frame structure is defined as follows:

Figure 6-01. Serial Data Frame Structure:



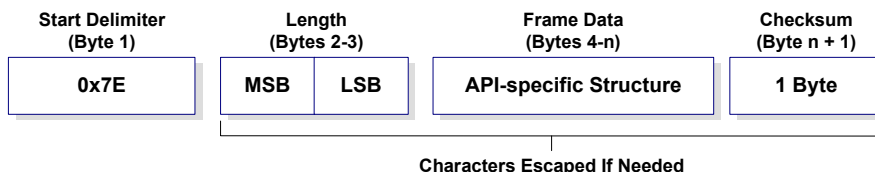
MSB = Most Significant Byte, LSB = Least Significant Byte

Any data received prior to the start delimiter is silently discarded. If the frame is not received correctly or if the checksum fails, the module will reply with a module status frame indicating the nature of the failure.

API Operation - with Escape Characters (AP parameter = 2)

When this API mode is enabled (AP = 2), the UART data frame structure is defined as follows:

UART Data Frame Structure - with escape control characters:



MSB = Most Significant Byte, LSB = Least Significant Byte

Escape characters. When sending or receiving a UART data frame, specific data values must be escaped (flagged) so they do not interfere with the data frame sequencing. To escape an interfering data byte, insert 0x7D and follow it with the byte to be escaped XOR'd with 0x20.

Data bytes that need to be escaped:

- 0x7E – Frame Delimiter
- 0x7D – Escape
- 0x11 – XON
- 0x13 – XOFF

Example – Raw serial data frame (before escaping interfering bytes):

0x7E 0x00 0x02 0x23 0x11 0xCB

0x11 needs to be escaped which results in the following frame:

0x7E 0x00 0x02 0x23 0x7D 0x31 0xCB

Note: In the above example, the length of the raw data (excluding the checksum) is 0x0002 and the checksum of the non-escaped data (excluding frame delimiter and length) is calculated as:

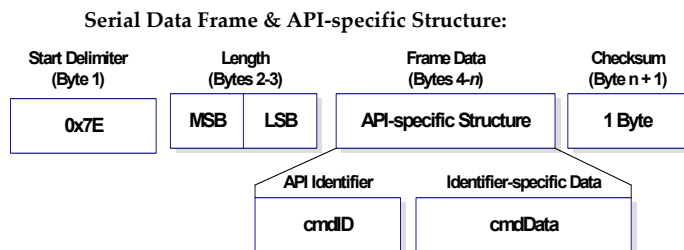
$0xFF - (0x23 + 0x11) = (0xFF - 0x34) = 0xCB$.

Length

The length field has two-byte value that specifies the number of bytes that will be contained in the frame data field. It does not include the checksum field.

Frame Data

Frame data of the serial data frame forms an API-specific structure as follows:



The cmdID frame (API-identifier) indicates which API messages will be contained in the cmdData frame (Identifier-specific data). Note that multi-byte values are sent big endian. The XBee modules support the following API frames:

API Frame Names and Values Sent to the Module

| API Frame Names | API ID |
|------------------------------------|--------|
| AT Command | 0x08 |
| AT Command - Queue Parameter Value | 0x09 |
| TX Request | 0x10 |
| Explicit TX Request | 0x11 |
| Remote Command Request | 0x17 |

API Frame Names and Values Received from the Module

| API Frame Names | API ID |
|--------------------------------------|--------|
| AT Command Response | 0x88 |
| Modem Status | 0x8A |
| Transmit Status | 0x8B |
| RX Indicator (AO=0) | 0x90 |
| Explicit Rx Indicator (AO=1) | 0x91 |
| Node Identification Indicator (AO=0) | 0x95 |
| Remote Command Response | 0x97 |

Note that requests are less than 0x80, and responses are always 0x80 or higher.

Checksum

To test data integrity, a checksum is calculated and verified on non-escaped data.

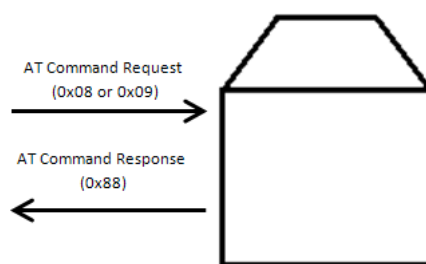
To calculate: Not including frame delimiters and length, add all bytes keeping only the lowest 8 bits of the result and subtract the result from 0xFF.

To verify: Add all bytes (include checksum, but not the delimiter and length). If the checksum is correct, the sum will equal 0xFF.

API Serial Exchanges

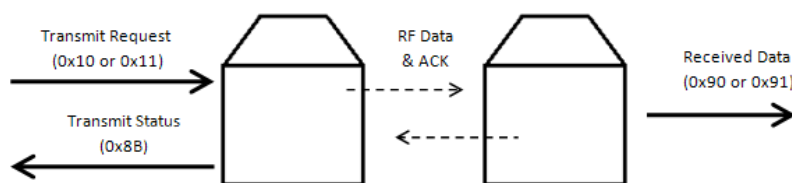
AT Commands

The following image shows the API frame exchange that takes place at the serial interface when sending an AT command request to read or set a module parameter. The response can be disabled by setting the frame ID to 0 in the request.



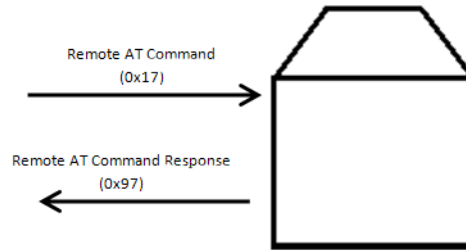
Transmitting and Receiving RF Data

The following image shows the API exchanges that take place at the serial interface when sending RF data to another device. The transmit status frame is always sent at the end of a data transmission unless the frame ID is set to 0 in the TX request. If the packet cannot be delivered to the destination, the transmit status frame will indicate the cause of failure. The received data frame (0x90 or 0x91) is set by the AP command.



Remote AT Commands

The following image shows the API frame exchanges that take place at the serial interface when sending a remote AT command. A remote command response frame is not sent out the serial interface if the remote device does not receive the remote command.



Supporting the API

Applications that support the API should make provisions to deal with new API frames that may be introduced in future releases. For example, a section of code on a host microprocessor that handles received serial API frames (sent out the module's DOUT pin) might look like this:

```
void XBee_HandleRxAPIFrame(_apiFrameUnion *papiFrame){
    switch(papiFrame->api_id){
        case RX_RF_DATA_FRAME:
            //process received RF data frame
            break;

        case RX_IO_SAMPLE_FRAME:
            //process IO sample frame
            break;

        case NODE_IDENTIFICATION_FRAME:
            //process node identification frame
            break;

        default:
            //Discard any other API frame types that are not being used
            break;
    }
}
```

Frame Descriptions

The following sections illustrate the types of frames encountered while using the API.

AT Command

Frame Type: 0x08

Used to query or set module parameters on the local device. This API command applies changes after executing the command. (Changes made to module parameters take effect once changes are applied.) The API example below illustrates an API frame when modifying the NH parameter value of the module

| | Frame Fields | | Offset | Example | Description |
|---------------|---------------------|----------------------------|--------|----------|--|
| API packet | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x04 | |
| | Frame-specific Data | Frame Type | 3 | 0x08 | |
| | | Frame ID | 4 | 0x52 | Identifies this command for correlation to a later response frame (0x88) to this command. If set to 0, no response frame will be sent. |
| | | AT Command | 5 | 0x4E (N) | Command Name - Two ASCII characters that identify the AT Command. |
| | | | 6 | 0x48 (H) | |
| | | Parameter Value (optional) | | | If present, indicates the requested parameter value to set the given register. If no characters present, register is queried. |
| | Checksum | | 8 | 0x0F | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

The above example illustrates an AT command when querying an NH value.

AT Command - Queue Parameter Value

Frame Type: 0x09

This API type allows module parameters to be queried or set. In contrast to the "AT Command" API type, new parameter values are queued and not applied until either the "AT Command" (0x08) API type or the AC (Apply Changes) command is issued. Register queries (reading parameter values) are returned immediately.

Example: Send a command to change the baud rate (BD) to 115200 baud, but don't apply changes yet. (Module will continue to operate at the previous baud rate until changes are applied.)

| | Frame Fields | | Offset | Example | Description |
|---------------|---------------------|---------------------------------------|--------|----------|---|
| API packet | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x05 | |
| | Frame-specific Data | Frame Type | 3 | 0x09 | |
| | | Frame ID | 4 | 0x01 | Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent. |
| | | AT Command | 5 | 0x42 (B) | Command Name - Two ASCII characters that identify the AT Command. |
| | | | 6 | 0x44 (D) | |
| | | Parameter Value (ATBD7 = 115200 baud) | | 0x07 | If present, indicates the requested parameter value to set the given register. If no characters present, register is queried. |
| | Checksum | | 8 | 0x68 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Note: In this example, the parameter could have been sent as a zero-padded 2-byte or 4-byte value.

TX Request

Frame Type: 0x10

A TX Request API frame causes the module to send data as an RF packet to the specified destination.

The 64-bit destination address should be set to 0x000000000000FFFF for a broadcast transmission (to all devices). For unicast transmissions the 64 bit address field should be set to the address of the desired destination node. The reserved field should be set to 0xFFFFE.

This example shows if escaping is disabled (AP=1).

| Frame Fields | | Offset | Example | Description |
|--|----------------------------|--------|---------|--|
| A P P l i c a t i o n s | Start Delimiter | 0 | 0x7E | |
| | Length | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | LSB 2 | 0x16 | |
| | Frame Type | 3 | 0x10 | |
| | Frame ID | 4 | 0x01 | Identifies this command for correlation to a later response frame (0x90) to this command. If set to 0, no response frame will be sent.. |
| | | 5 | 0x00 | |
| | 64-bit Destination Address | 6 | 0x13 | Set to the 64-bit address of the destination device. The following address is also supported: 0x000000000000FFFF - Broadcast address |
| | | 7 | 0xA2 | |
| | | 8 | 0x00 | |
| | | 9 | 0x40 | |
| | | 10 | 0x0A | |
| | | 11 | 0x01 | |
| | | 12 | 0x27 | |
| | Reserved | 13 | 0xFF | Set to 0xFFFE. |
| | | 14 | 0xFE | |
| | Broadcast Radius | 15 | 0x00 | Sets maximum number of hops a broadcast transmission can occur. If set to 0, the broadcast radius will be set to the maximum hops value. |
| | Transmit Options | 16 | 0x00 | If the Transmit Options Bitfield is 0, then the TO parameter will be used. Bitfield: bit 0: Disable ACK bit 1: Disable Route Discovery bit 2: Enable Unicast NACK messages. bit 3: Enable Unicast Trace Route messages. bits 6,7: b'01 - Point-Multipoint b'10 - Repeater mode (directed broadcast) b'11 - DigiMesh (not available on 10k product) All other bits must be set to 0. |
| | | 17 | 0x54 | |
| | | 18 | 0x78 | |
| | | 19 | 0x44 | |
| | | 20 | 0x61 | |
| | | 21 | 0x74 | |
| | | 22 | 0x61 | |
| | | 23 | 0x30 | |
| | RF Data | 24 | 0x41 | Data that is sent to the destination device |
| | | 25 | 0x13 | |
| | Checksum | 25 | 0x13 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: The example above shows how to send a transmission to a module where escaping is disabled (AP=1) with destination address 0x0013A200 40014011, payload "TxData0A". If escaping is enabled (AP=2), the frame should look like:

0x7E 0x00 0x16 0x10 0x01 0x00 0x7D 0x33 0xA2 0x00 0x40 0x0A 0x01 0x27

0xFF 0xFE 0x00 0x00 0x54 0x78 0x44 0x61 0x74 0x61 0x30 0x41 0x7D 0x33

The checksum is calculated (on all non-escaped bytes) as [0xFF - (sum of all bytes from API frame type through data payload)].

Explicit TX Request

Frame Type: 0x11

Allows application layer fields (endpoint and cluster ID) to be specified for a data transmission. Similar to the TX Request, but also requires application layer addressing fields to be specified (endpoints, cluster ID, profile

ID). An Explicit TX Request API frame causes the module to send data as an RF packet to the specified destination, using the specified source and destination endpoints, cluster ID, and profile ID.

The 64-bit destination address should be set to 0x000000000000FFFF for a broadcast transmission (to all devices). For unicast transmissions the 64 bit address field should be set to the address of the desired destination node. The reserved field should be set to 0xFFFE.

The broadcast radius can be set from 0 up to NH to 0xFF. If the broadcast radius exceeds the value of NH then the value of NH will be used as the radius. This parameter is only used for broadcast transmissions.

The maximum number of payload bytes can be read with the NP command.

| Frame Fields | | Offset | Example | Description |
|---------------|----------------------------|--------|---------|---|
| API Packet | Start Delimiter | 0 | 0x7E | |
| | Length | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | LSB 2 | 0x1A | |
| | Frame Type | 3 | 0x11 | |
| | Frame ID | 4 | 0x01 | Identifies this command for correlation to a later response frame (0x91) to this command. If set to 0, no response frame will be sent. |
| | 64-bit Destination Address | MSB 5 | 0x00 | Set to the 64-bit address of the destination device. The following address is also supported: 0x000000000000FFFF - Broadcast address |
| | | 6 | 0x13 | |
| | | 7 | 0xA2 | |
| | | 8 | 0x00 | |
| | | 9 | 0x01 | |
| | | 10 | 0x23 | |
| | | 11 | 0x84 | |
| | | LSB 12 | 0x00 | |
| | Reserved | 13 | 0xFF | Set to 0xFFFE. |
| | | 14 | 0xFE | |
| | Source Endpoint | 15 | 0xA0 | Source endpoint for the transmission. |
| | Destination Endpoint | 16 | 0xA1 | Destination endpoint for the transmission. |
| | Cluster ID | 17 | 0x15 | Cluster ID used in the transmission |
| | | 18 | 0x54 | |
| | Profile ID | 19 | 0xC1 | Profile ID used in the transmission |
| | | 20 | 0x05 | |
| | Broadcast Radius | 21 | 0x00 | Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value. |
| | Transmit Options | 22 | 0x00 | If the Transmit Options Bitfield is 0, then the TO parameter will be used. Bitfield: bit 0: Disable ACK bit 1: Don't attempt route Discovery. bit 2: Enable Unicast NACK messages. bit 3: Enable Unicast Trace Route messages. All other bits must be set to 0. |
| | | 23 | 0x54 | |
| | Data Payload | 24 | 0x78 | |
| | | 25 | 0x44 | |
| | | 26 | 0x61 | |
| | | 27 | 0x74 | |
| | | 28 | 0x61 | |
| | | 29 | 0xDD | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |
| | Checksum | | | |

Example: The above example sends a data transmission to a radio with a 64 bit address of 0x0013A20001238400 using a source endpoint of 0xA0, destination endpoint 0xA1, cluster ID = 0x1554, and profile ID 0xC105. Payload will be "TxData".

Remote AT Command Request

Frame Type: 0x17

Used to query or set module parameters on a remote device. For parameter changes on the remote device to take effect, changes must be applied, either by setting the apply changes options bit, or by sending an AC command to the remote.

| | Frame Fields | Offset | Example | Description |
|---|----------------------------|--------|----------------------|---|
| A P P l i c a t i o n | Start Delimiter | 0 | 0x7E | |
| | Length | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | LSB 2 | 0x10 | |
| | Frame Type | 3 | 0x17 | |
| | Frame ID | 4 | 0x01 | Identifies this command for correlation to a later response frame (0x97) to this command. If set to 0, no response frame will be sent. |
| | 64-bit Destination Address | MSB 5 | 0x00 | Set to the 64-bit address of the destination device. The following address is also supported: 0x000000000000FFFF - Broadcast address |
| | | 6 | 0x13 | |
| | | 7 | 0xA2 | |
| | | 8 | 0x00 | |
| | | 9 | 0x40 | |
| | | 10 | 0x40 | |
| | | 11 | 0x11 | |
| | | LSB 12 | 0x22 | |
| | Reserved | 13 | 0xFF | Set to 0xFFFE. |
| | | 14 | 0xFE | |
| | Remote Command Options | 15 | 0x02 (apply changes) | 0x02 - Apply changes on remote. (If not set, AC command must be sent before changes will take effect.) All other bits must be set to 0. |
| | AT Command | 16 | 0x42 (B) | Name of the command |
| | | 17 | 0x48 (H) | |
| | Command Parameter | 18 | 0x01 | If present, indicates the requested parameter value to set the given register. If no characters present, the register is queried. |
| | Checksum | 18 | 0xF5 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: The above example sends a remote command to change the broadcast hops register on a remote device to 1 (broadcasts go to 1-hop neighbors only), and apply changes so the new configuration value immediately takes effect. In this example, the 64-bit address of the remote is 0x0013A200 40401122.

AT Command Response

Frame Type: 0x88

In response to an AT Command message, the module will send an AT Command Response message. Some commands will send back multiple frames (for example, the ND (Node Discover) command).

| | Frame Fields | | Offset | Example | Description |
|---------------|---------------------|----------------|--------|------------|--|
| API Packet | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x05 | |
| | Frame-specific Data | Frame Type | 3 | 0x88 | |
| | | Frame ID | 4 | 0x01 | Identifies the serial interface data frame being reported. Note: If Frame ID = 0 in the associated request frame then no response frame will be delivered.. |
| | | AT Command | 5 | 'B' = 0x42 | Command Name - Two ASCII characters that identify the AT Command. |
| | | | 6 | 'D' = 0x44 | |
| | | Command Status | 7 | 0x00 | The least significant nibble indicates the command status: 0 = OK 1 = ERROR 2 = Invalid Command 3 = Invalid Parameter |
| | | Command Data | | | Register data in binary format. If the register was set, then this field is not returned, as in this example. |
| | Checksum | | 8 | 0xF0 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: Suppose the BD parameter is changed on the local device with a frame ID of 0x01. If successful (parameter was valid), the above response would be received.

Modem Status

Frame Type: (0x8A)

RF module status messages are sent from the module in response to specific conditions.

Example: The following API frame is returned when an API device powers up.

| | Frame Fields | | Offset | Example | Description |
|---------------|---------------------|------------|--------|---------|--|
| API Packet | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x02 | |
| | Frame-specific Data | Frame Type | 3 | 0x8A | |
| | | Status | 4 | 0x00 | 0x00 = Hardware reset 0x01 = Watchdog timer reset 0x0B = Network Woke Up 0x0C = Network Went To Sleep |
| | Checksum | | 5 | 0x75 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Transmit Status

Frame Type: 0x8B

When a TX Request is completed, the module sends a TX Status message. This message will indicate if the packet was transmitted successfully or if there was a failure.

| A P P L I C A T I O N | Frame Fields | | Offset | Example | Description |
|---|---------------------|----------------------|--------|---------|---|
| | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x07 | |
| | Frame-specific Data | Frame Type | 3 | 0x8B | |
| | | Frame ID | 4 | 0x47 | Identifies the serial interface data frame being reported. Note: If Frame ID = 0 in the associated request frame then no response frame will be delivered.. |
| | | Reserved | 5 | 0xFF | Reserved. |
| | | | 6 | 0xFE | |
| | | Transmit Retry Count | 7 | 0x00 | The number of application transmission retries that took place. |
| | | Delivery Status | 8 | 0x00 | 0x00 = Success 0x01 = MAC ACK Failure 0x21 = Network ACK Failure 0x25 = Route Not Found 0x74 = Payload too large. 0x75 = Indirect message unrequested. |
| | | Discovery Status | 9 | 0x02 | 0x00 = No Discovery Overhead 0x02 = Route Discovery |
| | Checksum | | 10 | 0x2E | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: In the above example, a unicast data transmission was sent successfully to a destination device using a frame ID of 0x47.)

Route Information Packet

Frame type: 0x8D

A Route Information Packet can be output for DigiMesh unicast transmissions on which the NACK enable or the Trace Route enable TX option is enabled.

| | Frame Fields | | Offset | Example | Description |
|---|---------------------|---------------------|--------|---------|---|
| A P P l i c a t i o n | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x2A | |
| | | Frame Type | 3 | 0x8D | |
| | | Source Event | 4 | 0x12 | 0x11 = NACK, 0x12 = Trace Route |
| | | Length | 5 | 0x2B | Number of bytes that follow (excluding checksum). If length increases, then new items have been added to the end of the list (for future revisions). |
| | | Timestamp | MSB 6 | 0x9C | System timer value on the node generating the Route Information Packet. |
| | | | 7 | 0x93 | |
| | | | 8 | 0x81 | |
| | | | LSB 9 | 0x7F | |
| | | ACK Timeout Count | 10 | 0x00 | The number of MAC ACK timeouts that occurred. |
| | | Reserved | 11 | 0x00 | Reserved |
| | | Reserved | 12 | 0x00 | Reserved |
| | Frame-specific Data | Destination Address | MSB 13 | 0x00 | Address of the final destination node of this network level transmission. |
| | | | 14 | 0x13 | |
| | | | 15 | 0xA2 | |
| | | | 16 | 0x00 | |
| | | | 17 | 0x40 | |
| | | | 18 | 0x52 | |
| | | | 19 | 0xAA | |
| | | | LSB 20 | 0xAA | |
| | | Source Address | MSB 21 | 0x00 | Address of the source node of this network level transmission. |
| | | | 22 | 0x13 | |
| | | | 23 | 0xA2 | |
| | | | 24 | 0x00 | |
| | | | 25 | 0x40 | |
| | | | 26 | 0x52 | |
| | | | 27 | 0xDD | |
| | | | LSB 28 | 0xDD | |
| | | Responder Address | MSB 29 | 0x00 | Address of the node that generated this Route Information Packet after sending (or attempting to send) the packet to the next hop (the Reciever Node) |
| | | | 30 | 0x13 | |
| | | | 31 | 0xA2 | |
| | | | 32 | 0x00 | |
| | | | 33 | 0x40 | |
| | | | 34 | 0x52 | |
| | | | 35 | 0xBB | |
| | | | LSB 36 | 0xBB | |
| | | Receiver Address | MSB 37 | 0x00 | Address of the node to which the data packet was just sent (or attempted to be sent to) |
| | | | 38 | 0x13 | |
| | | | 39 | 0xA2 | |
| | | | 40 | 0x00 | |
| | | | 41 | 0x40 | |
| | | | 42 | 0x52 | |
| | | | 43 | 0xCC | |
| | | | LSB 44 | 0xCC | |
| | Checksum | | 45 | 0xCE | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: The above example represents a possible Route Information Frame that could be received when doing a trace route on a transmission from a radio with serial number 0x0013a2004052AAAA to a radio with serial number 0x0013a2004052DDDD. This particular

frame indicates that the transmission was successfully forwarded from the radio with serial number 0x0013a2004052BBBB to the radio with serial number 0x0013a2004052CCCC.

Aggregate Addressing Update

Frame type: 0x8E

An Aggregate Addressing Update frame is output on an API-enabled node when an address update frame (generated by the AG command being issued on a node in the network) causes the node to update its DH and DL registers.

| Frame Fields | | Offset | Example | Description |
|---------------|-----------------|--------|---------|--|
| API Packet | Start Delimiter | 0 | 0x7E | |
| | Length | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | LSB 2 | 0x12 | |
| | Frame Type | 3 | 0x8E | |
| | Format ID | 4 | 0x00 | Byte reserved to indicate format of additional packet information which may be added in future firmware revisions. In the current firmware revision, 0x00 is returned in this field. |
| | New Address | MSB 5 | 0x00 | Address to which DH and DL are being set |
| | | 6 | 0x13 | |
| | | 7 | 0xA2 | |
| | | 8 | 0x00 | |
| | | 9 | 0x40 | |
| | | 10 | 0x52 | |
| | | 11 | 0xBB | |
| | | LSB 12 | 0xBB | |
| | Old Address | 13 | 0x00 | Address to which DH and DL were previously set |
| | | 14 | 0x13 | |
| | | 15 | 0xA2 | |
| | | 16 | 0x00 | |
| | | 17 | 0x40 | |
| | | 18 | 0x52 | |
| | | 19 | 0xAA | |
| | | 20 | 0xAA | |
| | Checksum | 21 | 0x2E | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: In the above example a radio which had a destination address (DH/DL) of 0x0013A2004052AAAA updated its destination address to 0x0013A2004052BBBB.

RX Indicator

Frame Type: (0x90)

When the module receives an RF packet, it is sent out the UART using this message type.

| | Frame Fields | | Offset | Example | Description |
|---|---------------------|-----------------------|--------|---------|---|
| | | | | | |
| A P P l i c a t i o n | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x12 | |
| | Frame-specific Data | Frame Type | 3 | 0x90 | |
| | | Frame ID | 4 | 0x00 | Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent. |
| | | | | | |
| | | 64-bit Source Address | MSB 5 | 0x13 | 64-bit address of sender |
| | | | 6 | 0xA2 | |
| | | | 7 | 0x00 | |
| | | | 8 | 0x40 | |
| | | | 9 | 0x52 | |
| | | | 10 | 0x2B | |
| | | | LSB 11 | 0xAA | |
| | | Reserved | 12 | 0xFF | Reserved |
| | | | 13 | 0xFE | |
| | | Receive Options | 14 | 0x01 | bit 0: Packet was acknowledged. bit 1: Broadcasted packet. bits 6,7: b'01 - Point-Multipoint b'10 - Repeater mode (directed broadcast) b'11 - DigiMesh (not available on 10k product) other bits should be ignored. |
| | | | | | |
| | | Received Data | 15 | 0x52 | Received RF data |
| | | | 16 | 0x78 | |
| | | | 17 | 0x44 | |
| | | | 18 | 0x61 | |
| | | | 19 | 0x74 | |
| | | | 20 | 0x61 | |
| | Checksum | | 21 | 0x11 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: Example: In the above example, a device with a 64-bit address of 0x0013A200 40522BAA sends a unicast data transmission to a remote device with payload "RxData". If AO=0 on the receiving device, it would send the above frame out its serial interface.

Explicit Rx Indicator

Frame Type:0x91

When the modem receives an RF packet it is sent out the UART using this message type (when AO=1).

| | Frame Fields | Offset | Example | Description |
|---|---------------------|----------------------|---------|--|
| A P P L I C A T I O N | Start Delimiter | 0 | 0x7E | |
| | Length | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | LSB 2 | 0x18 | |
| | Frame-specific Data | Frame Type | 3 0x91 | |
| | | MSB 4 | 0x00 | 64-bit address of sender |
| | | 5 | 0x13 | |
| | | 6 | 0xA2 | |
| | | 7 | 0x00 | |
| | | 8 | 0x40 | |
| | | 9 | 0x52 | |
| | | 10 | 0x2B | |
| | | LSB 11 | 0xAA | |
| | | Reserved | 12 0xFF | Reserved. |
| | | 13 | 0xFE | |
| | | Source Endpoint | 14 0xE0 | Endpoint of the source that initiated the transmission |
| | | Destination Endpoint | 15 0xE0 | Endpoint of the destination the message is addressed to. |
| | | Cluster ID | 16 0x22 | Cluster ID the packet was addressed to. |
| | | 17 | 0x11 | |
| | | Profile ID | 18 0xC1 | Profile ID the packet was addressed to. |
| | | 19 | 0x05 | |
| | | Receive Options | 20 0x02 | bit 0: Packet was acknowledged. bit 1: Broadcasted packet. bits 6,7 b'01 - Point-Multipoint b'10 - Repeater mode (directed broadcast) b'11 - DigiMesh (not available on 10k product) other bits should be ignored. |
| | | Received Data | 21 0x52 | Received RF data |
| | | | 22 0x78 | |
| | | | 23 0x44 | |
| | | | 24 0x61 | |
| | | | 25 0x74 | |
| | | | 26 0x61 | |
| | Checksum | 27 | 0x56 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: In the above example, a device with a 64-bit address of 0x0013A200 40522BAA sends a broadcast data transmission to a remote device with payload "RxData". Suppose the transmission was sent with source and destination endpoints of 0xE0, cluster ID=0x2211, and profile ID=0xC105. If AO=1 on the receiving device, it would send the above frame out its serial interface.

Node Identification Indicator

Frame Type:0x95

This frame is received when a module transmits a node identification message to identify itself (when AO=0). The data portion of this frame is similar to a network discovery response frame (see ND command).

| | Frame Fields | | Offset | Example | Description |
|---|---------------------|--------------------------|--------|---------|--|
| A P P L I C A T I O N | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x25 | |
| | Frame-specific Data | Frame Type | 3 | 0x95 | |
| | | 64-bit Source Address | MSB 4 | 0x00 | 64-bit address of sender |
| | | | 5 | 0x13 | |
| | | | 6 | 0xA2 | |
| | | | 7 | 0x00 | |
| | | | 8 | 0x40 | |
| | | | 9 | 0x74 | |
| | | | 10 | 0x02 | |
| | | | LSB 11 | 0xAC | |
| | | Reserved | 12 | 0xFF | Reserved. |
| | | | 13 | 0xFE | |
| | | Receive Options | 14 | 0xC2 | 0x01 - Packet Acknowledged 0x02 - Packet was a broadcast packet 0x40 - Point-multipoint packet 0x80 - Directed broadcast packet 0xC0 - DigiMesh packet |
| | | Reserved | 15 | 0xFF | Reserved |
| | | | 16 | 0xFE | |
| | | 64-bit Address | MSB 17 | 0x00 | Indicates the 64-bit address of the remote module that transmitted the node identification frame. |
| | | | 18 | 0x13 | |
| | | | 19 | 0xA2 | |
| | | | 20 | 0x00 | |
| | | | 21 | 0x40 | |
| | | | 22 | 0x74 | |
| | | | 23 | 0x02 | |
| | | | LSB 24 | 0xAC | |
| | | NI String | 25 | 0x20 | Node identifier string on the remote device. The NI string is terminated with a NULL byte (0x00). |
| | | | 26 | 0x00 | |
| | | Reserved | 27 | 0xFF | Reserved |
| | | | 28 | 0xFE | |
| | | Device Type | 29 | 0x01 | 0=Coordinator 1=Normal Mode 2=End Device (See the NO command description for more options) |
| | | Source Event | 30 | 0x01 | 1=Frame sent by node identification pushbutton event (See D0 command description) |
| | | Digi Profile ID | 31 | 0xC1 | Set to Digi's application profile ID |
| | | | 32 | 0x05 | |
| | | Digi Manufacturer ID | 33 | 0x10 | Set to Digi's Manufacturer ID |
| | | | 34 | 0x1E | |
| | | Digi DD Value (optional) | 35 | 0x00 | Reports the DD value of the responding module (this field can be enabled with the NO command) |
| | | | 36 | 0x0C | |
| | | | 37 | 0x00 | |
| | | RSSI (optional) | 38 | 0x00 | |
| | | | 39 | 0x2E | |
| | Checksum | | 40 | 0x33 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: If the commissioning push button is pressed on a remote router device with 64-bit address 0x0013a200407402ac and default NI string, the following node identification indicator would be received: 0x7e 0025 9500 13a2 0040 7402 acff fec2 fffe 0013 a200 4074 02ac 2000 fffe 0101 c105 101e 000c 0000 2e33

Remote Command Response

Frame Type: 0x97

If a module receives a remote command response RF data frame in response to a Remote AT Command Request, the module will send a Remote AT Command Response message out the serial interface. Some commands may send back multiple frames--for example, Node Discover (ND) command.

| A P P l i c a t i o n | Frame Fields | | Offset | Example | Description |
|---|---------------------|--------------------------------|--------|---------|--|
| | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x13 | |
| | Frame-specific Data | Frame Type | 3 | 0x97 | |
| | | Frame ID | 4 | 0x55 | This is the same value passed in to the request. If Frame ID = 0 in the associated request frame then no response frame will be delivered. |
| | | | MSB 5 | 0x00 | |
| | | 64-bit Source (remote) Address | 6 | 0x13 | The address of the remote radio returning this response. |
| | | | 7 | 0xA2 | |
| | | | 8 | 0x00 | |
| | | | 9 | 0x40 | |
| | | | 10 | 0x52 | |
| | | | 11 | 0x2B | |
| | | | LSB 12 | 0xAA | |
| | | Reserved | 13 | 0xFF | Reserved |
| | | | 14 | 0xFE | |
| | | AT Commands | 15 | 0x53 | Name of the command |
| | | | 16 | 0x4C | |
| | | Command Status | 17 | 0x00 | The least significant nibble indicates the command status: 0 = OK 1 = ERROR 2 = Invalid Command 3 = Invalid Parameter The most significant nibble is a bitfield as follows: 0x40 = The RSSI field is invalid and should be ignored. Software prior to version 8x60 did not include RSSI information 0x80 = Response is a remote command. |
| | | | | | |
| | | | | | |
| | | | | | |
| | | Command Data | 18 | 0x40 | The value of the required register |
| | | | 19 | 0x52 | |
| | | | 20 | 0x2B | |
| | | | 21 | 0xAA | |
| | Checksum | | 22 | 0xF4 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

Example: If a remote command is sent to a remote device with 64-bit address 0x0013A20040522BAA to query the SL command, and if the frame ID=0x55, the response would look like the above example.

7. Advanced Application Features

Remote Configuration Commands

A module in API mode has provisions to send configuration commands to remote devices using the Remote Command Request API frame (See API Operations chapter.) This API frame can be used to send commands to a remote module to read or set command parameters.

Sending a Remote Command

To send a remote command, the Remote Command Request frame should be populated with the 64-bit address of the remote device, the correct command options value, and the command and parameter data (optional). If a command response is desired, the Frame ID should be set to a non-zero value. Only unicasts of remote commands are supported. Remote commands cannot be broadcast.

Applying Changes on Remote Devices

When remote commands are used to change command parameter settings on a remote device, parameter changes do not take effect until the changes are applied. For example, changing the BD parameter will not change the actual serial interface rate on the remote until the changes are applied. Changes can be applied using remote commands in one of three ways:

- Set the apply changes option bit in the API frame
- Issue an AC command to the remote device
- Issue a WR + FR command to the remote device to save changes and reset the device.

Remote Command Responses

If the remote device receives a remote command request transmission, and the API frame ID is non-zero, the remote will send a remote command response transmission back to the device that sent the remote command. When a remote command response transmission is received, a device sends a remote command response API frame out its serial interface. The remote command response indicates the status of the command (success, or reason for failure), and in the case of a command query, it will include the register value. The device that sends a remote command will not receive a remote command response frame if:

- The destination device could not be reached
- The frame ID in the remote command request is set to 0.

Network Commissioning and Diagnostics

Network commissioning is the process whereby devices in a network are discovered and configured for operation. The XBee modules include several features to support device discovery and configuration. In addition to configuring devices, a strategy must be developed to place devices to ensure reliable routes.

To accommodate these requirements, the XBee modules include various features to aid in device placement, configuration, and network diagnostics.

Device Configuration

XBee modules can be configured locally through serial commands (AT or API), or remotely through remote API commands. API devices can send configuration commands to set or read the configuration settings of any device in the network.

Network Link Establishment and Maintenance

Building Aggregate Routes

In many applications it is necessary for many or all of the nodes in the network to transmit data to a central aggregator node. In a new DigiMesh network the overhead of these nodes discovering routes to the

aggregator node can be extensive and taxing on the network. To eliminate this overhead the AG command can be used to automatically build routes to an aggregate node in a DigiMesh network.

To send a unicast, modules configured for transparent mode (AP=0) must set their DH/DL registers to the MAC address of the node to which they need to transmit to. In networks of transparent mode modules which transmit to an aggregator node it is necessary to set every module's DH/DL registers to the MAC address of the aggregator node. This can be a tedious process. The AR command can be used to set the DH/DL registers of all the nodes in a DigiMesh network to that of the aggregator node in a simple and effective method.

Upon deploying a DigiMesh network the AG command can be issued on the desired aggregator node to cause all nodes in the network to build routes to the aggregator node. The command can optionally be used to automatically update the DH/DL registers to match the MAC address of the aggregator node. The AG command requires a 64-bit parameter. The parameter indicates the current value of the DH/DL registers on a module which should be replaced by the 64-bit address of the node sending the AG broadcast. If it is not desirable to update the DH/DL of the module receiving the AG broadcast then the invalid address of 0xFFFFE can be used. API enabled modules will output an Aggregator Update API frame if they update their DH/DL address (see the API section of this manual for a description of the frame). All modules which receive an AG broadcast will update their routing table information to build a route to the sending module, regardless of whether or not their DH/DL address is updated. This routing information will be used for future transmissions of DigiMesh unicasts.

Example 1: To update the DH/DL registers of all modules in the network to be equal to the MAC address of an aggregator node with a MAC address of 0x0013a2004052c507 after network deployment the following technique could be employed:

1. Deploy all modules in the network with the default DH/DL of 0xFFFF.
2. Issue an ATAGFFFF command on the aggregator node.

Following the preceding sequence would result in all of the nodes in the network which received the AG broadcast to have a DH of 0x0013a200 and a DL of 0x4052c507. These nodes would have automatically built a route to the aggregator.

Example 2: To cause all nodes in the network to build routes to an aggregator node with a MAC address of 0x0013a2004052c507 without affecting the DH/DL of any nodes in the network the ATAGFFFE command should be issued on the aggregator node. This will cause an AG broadcast to be sent to all nodes in the network. All of the nodes will update their internal routing table information to contain a route to the aggregator node. None of the nodes will update their DH/DL registers (because none of the registers are set to an address of 0xFFFFE).

Node Replacement

The AG command can also be used to update the routing table and DH/DL registers in the network after a module is replaced. The DH/DL registers of nodes in the network can also be updated. To update only the routing table information without affecting the DH/DL registers then the process of Example 2 above can be used. To update the DH/DL registers of the network then the method of Example 3 below can be used.

Example 3: The module with serial number 0x0013a2004052c507 was being used as a network aggregator. It was replaced with a module with serial number 0x0013a200f5e4d3b2. The ATAG0013a2004052c507 command should be issued on the new module. This will cause all modules which had a DH/DL register setting of 0x0013a2004052c507 to update their DH/DL register setting to the MAC address of the sending module (0x0013a200f5e4d3b2).

Device Placement

For a network installation to be successful, the installer must be able to determine where to place individual XBee devices to establish reliable links throughout the network.

Link Testing

A good way to measure the performance of a network is to send unicast data through the network from one device to another to determine the success rate of many transmissions. To simplify link testing, the modules support a loopback cluster ID (0x12) on the data endpoint (0xE8). Any data sent to this cluster ID on the data endpoint will be transmitted back to the sender.

The configuration steps to send data to the loopback cluster ID depend on the AP setting:

AT Configuration (AP=0)

To send data to the loopback cluster ID on the data endpoint of a remote device, set the CI command value to 0x12. The SE and DE commands should be set to 0xE8 (default value). The DH and DL commands should be set to the address of the remote (0 for the coordinator, or the 64-bit address of the remote). After exiting command mode, any received serial characters will be transmitted to the remote device, and returned to the sender.

API Configuration (AP=1 or AP=2)

Send an Explicit Addressing Command API frame (0x11) using 0x12 as the cluster ID and 0xE8 as the source and destination endpoint. Data packets received by the remote will be echoed back to the sender.

RSSI Indicators

It is possible to measure the received signal strength on a device using the DB command. DB returns the RSSI value (measured in -dBm) of the last received packet. However, this number can be misleading in DigiMesh networks. The DB value only indicates the received signal strength of the last hop. If a transmission spans multiple hops, the DB value provides no indication of the overall transmission path, or the quality of the worst link - it only indicates the quality of the last link and should be used accordingly.

The DB value can be determined in hardware using the RSSI/PWM module pin (pin 6). If the RSSI PWM functionality is enabled (P0 command), when the module receives data, the RSSI PWM is set to a value based on the RSSI of the received packet. (Again, this value only indicates the quality of the last hop.) This pin could potentially be connected to an LED to indicate if the link is stable or not.

Device Discovery

Network Discovery

The network discovery command can be used to discover all Digi modules that have joined a network. Issuing the ND command sends a broadcast network discovery command throughout the network. All devices that receive the command will send a response that includes the device's addressing information, node identifier string (see NI command), and other relevant information. This command is useful for generating a list of all module addresses in a network.

When a device receives the network discovery command, it waits a random time before sending its own response. The maximum time delay is set on the ND sender with the NT command. The ND originator includes its NT setting in the transmission to provide a delay window for all devices in the network. Large networks may need to increase NT to improve network discovery reliability. The default NT value is 0x82 (13 seconds).

Neighbor Polling

The neighbor poll command can be used to discover the modules which are immediate neighbors (within RF range) of a particular node. This command is useful in determining network topology and determining possible routes. The command is issued using the FN command. The FN command can be initiated locally on a node using AT command mode or by using a local AT command request frame. The command can also be initiated remotely by sending the target node an FN command using a remote AT command request API frame.

A node which executes an FN command will send a broadcast to all of its immediate neighbors. All radios which receive this broadcast will send an RF packet to the node that initiated the FN command. In the case where the command is initiated remotely this means that the responses are sent directly to the node which sent the FN command to the target node. The response packet is output on the initiating radio in the same format as a network discovery frame.

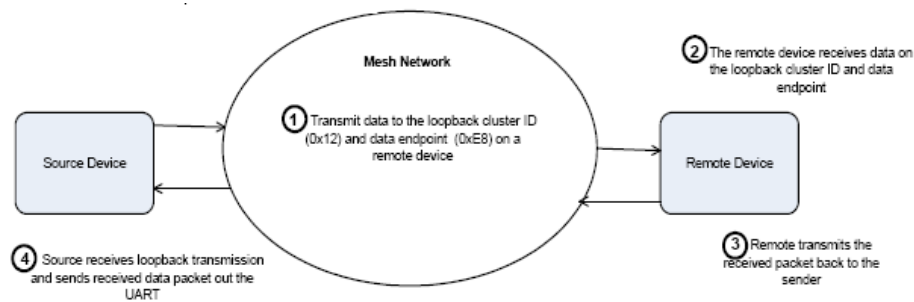
Link Reliability

For a mesh network installation to be successful, the installer must be able to determine where to place individual XBee devices to establish reliable links throughout the mesh network.

Network Link Testing

A good way to measure the performance of a mesh network is to send unicast data through the network from one device to another to determine the success rate of many transmissions. To simplify link testing, the

modules support a loopback cluster ID (0x12) on the data endpoint (0xE8). Any data sent to this cluster ID on the data endpoint will be transmitted back to the sender. This is shown in the figure below:



Demonstration of how the loopback cluster ID and data endpoint can be used to measure the link quality in a mesh network

The configuration steps to send data to the loopback cluster ID depend on the AP setting:

AT Configuration (AP=0)

To send data to the loopback cluster ID on the data endpoint of a remote device, set the CI command value to 0x12. The SE and DE commands should be set to 0xE8 (default value). The DH and DL commands should be set to the address of the remote. After exiting command mode, any received serial characters will be transmitted to the remote device, and returned to the sender.

API Configuration (AP=1 or AP=2)

Send an Explicit Addressing ZigBee Command API frame (0x11) using 0x12 as the cluster ID and 0xE8 as the source and destination endpoint. Data packets received by the remote will be echoed back to the sender.

Link Testing Between Adjacent Devices

It is often advantageous to test the quality of a link between two adjacent nodes in a network. The Test Link Request Cluster ID can be used to send a number of test packets between any two nodes in a network.

A link test can be initiated using an Explicit TX Request frame. The command frame should be addressed to the Test Link Request Cluster ID (0x0014) on destination endpoint 0xE6 on the radio which should execute the test link. The Explicit TX Request frame should contain a 12 byte payload with the following format:

| Number of Bytes | Field Name | Description |
|-----------------|---------------------|--|
| 8 | Destination address | The address with which the radio should test its link |
| 2 | Payload size | The size of the test packet. (The maximum payload size for this radio can be queried with the MP command.) |
| 2 | Iterations | The number of packets which should be sent. This should be a number between 1 and 4000. |

After completing the transmissions of the test link packets the executing radio will send the following data packet to the requesting radio's Test Link Result Cluster (0x0094) on endpoint (0xE6). If the requesting radio is configured to operate in API mode then the following information will be output as an API Explicit RX Indicator Frame:

| Number of Bytes | Field Name | Description |
|-----------------|---------------------|---|
| 8 | Destination address | The address with which the radio tested its link |
| 2 | Payload size | The size of the test packet that was sent to test the link. |
| 2 | Iterations | The number of packets which were sent. |
| 2 | Success | The number of packets successfully acknowledged. |
| 2 | Retries | The total number of MAC retries used to transfer all the packets. |
| 1 | Result | 0x00 - command was successful 0x03 - invalid parameter used |
| 1 | RR | The maximum number of MAC retries allowed. |
| 1 | maxRSSI | The strongest RSSI reading observed during the test. |
| 1 | minRSSI | The weakest RSSI reading observed during the test. |
| 1 | avgRSSI | The average RSSI reading observed during the test. |

Example:

Suppose that the link between radio A (SH/SL = 0x0013a20040521234) and radio B (SH/SL=0x0013a2004052abcd) is to be tested by transmitting 1000 40 byte packets. The following API packet should be sent to the serial interface of the radio on which the results should be output, radio C. Note that radio C can be the same radio as radio A or B (whitespace used to delineate fields, bold text is the payload portion of the packet):

```
7E 0020 11 01 0013A20040521234 FFFE E6 E6 0014 C105 00 00 0013A2004052ABCD 0028 03E8 EB
```

And the following is a possible packet that could be returned:

```
7E 0027 91 0013A20040521234 FFFE E6 E6 0094 C105 00 0013A2004052ABCD 0028 03E8 03E7 0064 00 0A 50 53 52 9F
```

(999 out of 1000 packets successful, 100 retries used, RR=10, maxRSSI=-80dBm, minRSSI=-83dBm, avgRSSI=-82dBm)

If the result field is not equal to zero then an error has occurred. The other fields in the packet should be ignored. If the Success field is equal to zero then the RSSI fields should be ignored.

Trace Routing

In many applications it is useful to determine the route which a DigiMesh unicast takes to its destination. This information is especially useful when setting up a network or diagnosing problems within a network. The Trace Route API option of Tx Request Packets (see the API section of this manual for a description of the API frames) causes routing information packets to be transmitted to the originator of a DigiMesh unicast by the intermediate nodes.

When a unicast is sent with the Trace Route API option enabled, the unicast is sent to its destination radios which forward the unicast to its eventual destination will transmit a Route Information (RI) packet back along the route to the unicast originator. A full description of Route Information API packets can be found in the API section of this manual. In general they contain addressing information for the unicast and the intermediate hop for which the trace route packet was generated, RSSI information, and other link quality information.

Example:

Suppose that a data packet with trace route enabled was successfully unicast from radio A to radio E, through radios B, C, and D. The following sequence would occur:

- After the successful MAC transmission of the data packet from A to B, A would output a RI Packet indicating that the transmission of the data packet from A to E was successfully forwarded one hop from A to B.
- After the successful MAC transmission of the data packet from B to C, B would transmit a RI Packet to A. A would output this RI packet out its serial interface upon reception.
- After the successful MAC transmission of the data packet from C to D, C would transmit a RI Packet to A (through B). A would output this RI packet out its serial interface upon reception.
- After the successful MAC transmission of the data packet from D to E, D would transmit a RI Packet to A (through C and B). A would output this RI packet out its serial interface upon reception.

It is important to note that Route Information packets are not guaranteed to arrive in the same order as the unicast packet took. It is also possible for the transmission of Route Information packets on a weak route to fail before arriving at the unicast originator.

Because of the large number of Route Information packets which can be generated by a unicast with Trace Route enabled it is suggested that the Trace Route option only be used for occasional diagnostic purposes and not for normal operations.

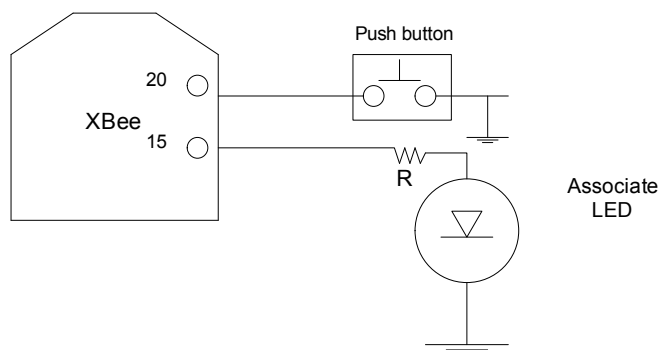
NACK Messages

The NACK API option of Tx Request Packets (see the API section of this manual for a description of the API frames) provides the option to have a Route Information packet generated and sent to the originator of a unicast when a MAC acknowledgment failure occurs on one of the hops to the destination. This information is useful because it allows marginal links to be identified and repaired.

Commissioning Pushbutton and Associate LED

The XBee modules support a set of commissioning and LED behaviors to aid in device deployment and commissioning. These include the commissioning push button definitions and associate LED behaviors. These features can be supported in hardware as shown below.

Commissioning Pushbutton and Associate LED Functionalities



A pushbutton and an LED can be connected to module pins 20 and 15 respectively to support the commissioning pushbutton and associated LED functionalities.

Commissioning Pushbutton

The commissioning pushbutton definitions provide a variety of simple functions to aid in deploying devices in a network. The commissioning button functionality on pin 20 is enabled by setting the D0 command to 1 (enabled by default).

| Button Presses | Sleep Configuration and Sync Status | Action |
|----------------|--------------------------------------|---|
| 1 | Not configured for sleep | Immediately sends a Node Identification broadcast transmission. All devices that receive this transmission will blink their Associate LED rapidly for 1 second. All API devices that receive this transmission will send a Node Identification frame out their serial interface (API ID 0x95). |
| 1 | Configured for asynchronous sleep | Wakes the module for 30 seconds. Immediately sends a Node Identification broadcast transmission. All devices that receive this transmission will blink their Associate LED rapidly for 1 second. All API devices that receive this transmission will send a Node Identification frame out their serial interface (API ID 0x95). |
| 1 | Configured for synchronous sleep | Wakes the module for 30 seconds (or until the entire module goes to sleep). Queues a Node Identification broadcast transmission to be sent at the beginning of the next network wake cycle. All devices that receive this transmission will blink their Associate LEDs rapidly for 1 second. All API devices that receive this transmission will send a Node Identification frame out their serial interface (API ID 0x95). |
| 2 | Not configured for synchronous sleep | No effect. |
| 2 | Configured for synchronous sleep | Causes a node which is configured with sleeping router nomination enabled (see description of the ATSO – sleep options command in the XBee module's Product Manual) to immediately nominate itself as the network sleep coordinator. |
| 4 | Any | Issues an ATRE to restore module parameters to default values. |

Button presses may be simulated in software using the ATCB command. ATCB should be issued with a parameter set to the number of button presses to execute. (i.e. sending ATCB1 will execute the action(s) associated with a single button press.)

The node identification frame is similar to the node discovery response frame – it contains the device's address, node identifier string (NI command), and other relevant data. All API devices that receive the node identification frame send it out their serial interface as an API Node Identification Indicator frame (0x95).

Associate LED

The Associate pin (pin 15) can provide indication of the device's sleep status and diagnostic information. To take advantage of these indications, an LED can be connected to the Associate pin as shown in the figure above. The Associate LED functionality is enabled by setting the D5 command to 1 (enabled by default). If enabled, the Associate pin is configured as an output and will behave as described in the following sections.

The Associate pin indicates the synchronization status of a sleep compatible node. On a non-sleep compatible node the pin functions as a power indicator. The following table describes this functionality.

The LT command can be used to override the blink rate of the Associate pin. When set to 0, the device uses the default blink time (500ms for sleep coordinator, 250ms otherwise).

| Sleep mode | LED Status | Meaning |
|------------|--------------|---|
| 0 | On, blinking | The device is powered and operating properly. |
| 1, 4, 5 | Off | The device is in a low power mode. |

| Sleep mode | LED Status | Meaning |
|------------|---------------------------------------|---|
| 1, 4, 5 | On, blinking | The device is powered, awake and is operating properly. |
| 7 | On, solid | The network is asleep or the device has not synchronized with the network or has lost synchronization with the network. |
| 7, 8 | On, slow blinking (500 ms blink time) | The device is acting as the network sleep coordinator and is operating properly. |
| 7, 8 | On, fast blinking (250 ms blink time) | The device is properly synchronized with the network. |
| 8 | Off | The device is in a low power mode. |
| 8 | On, solid | The device has not synchronized or has lost synchronization with the network. |

Diagnostics Support

The Associate pin works with the commissioning pushbutton to provide additional diagnostic behaviors to aid in deploying and testing a network. If the commissioning push button is pressed once the device transmits a broadcast node identification packet at the beginning of the next wake cycle if sleep compatible, or immediately if not sleep compatible. If the Associate LED functionality is enabled (D5 command), a device that receives this transmission will blink its Associate pin rapidly for 1 second.

I/O Line Monitoring

I/O Samples

The XBee modules support both analog input and digital IO line modes on several configurable pins.

Queried Sampling

Parameters for the pin configuration commands typically include the following:

| Pin Command Parameter | Description |
|-----------------------|--|
| 0 | Unmonitored digital input |
| 1 | Reserved for pin-specific alternate functionalities. |
| 2 | Analog input (A/D pins) or PWM output (PWM pins) |
| 3 | Digital input, monitored. |
| 4 | Digital output, low. |
| 5 | Digital output, high. |
| 7 | Alternate functionalities, where applicable. |

Setting the configuration command that corresponds to a particular pin will configure the pin:

| Module Pin Names | Module Pin Number | Configuration Command |
|-----------------------|-------------------|-----------------------|
| CD / DIO12 | 4 | P2 |
| PWM0 / RSSI / DIO10 | 6 | P0 |
| PWM1 / DIO11 | 7 | P1 |
| DTR / SLEEP_RQ / DIO8 | 9 | D8 |
| AD4 / DIO4 | 11 | D4 |
| CTS / DIO7 | 12 | D7 |
| ON_SLEEP / DIO9 | 13 | D9 |
| ASSOC / AD5 / DIO5 | 15 | D5 |

| Module Pin Names | Module Pin Number | Configuration Command |
|-------------------------------------|-------------------|-----------------------|
| RTS / DIO6 | 16 | D6 |
| AD3 / DIO3 | 17 | D3 |
| AD2 / DIO2 | 18 | D2 |
| AD1 / DIO1 | 19 | D1 |
| AD0 / DIO0 / CommissioningButton | 20 | D0 |

See the command table for more information. Pullup resistors for each digital input can be enabled using the PR command.

| | | |
|----------|----------------------|---|
| 1 | Sample Sets | Number of sample sets in the packet. (Always set to 1.) |
| 2 | Digital Channel Mask | <p>Indicates which digital IO lines have sampling enabled. Each bit corresponds to one digital IO line on the module.</p> <ul style="list-style-type: none"> • bit 0 = AD0/DIO0 • bit 1 = AD1/DIO1 • bit 2 = AD2/DIO2 • bit 3 = AD3/DIO3 • bit 4 = DIO4 • bit 5 = ASSOC/DIO5 • bit 6 = RTS/DIO6 • bit 7 = CTS/GPIO7 • bit 8 = DTR / SLEEP_RQ / DIO8 • bit 9 = ON_SLEEP / DIO9 • bit 10 = RSSI/DIO10 • bit 11 = PWM/DIO11 • bit 12 = CD/DIO12 <p>For example, a digital channel mask of 0x002F means DIO0,1,2,3, and 5 are enabled as digital IO.</p> |
| 1 | Analog Channel Mask | <p>Indicates which lines have analog inputs enabled for sampling. Each bit in the analog channel mask corresponds to one analog input channel.</p> <ul style="list-style-type: none"> • bit 0 = AD0/DIO0 • bit 1 = AD1/DIO1 • bit 2 = AD2/DIO2 • bit 3 = AD3/DIO3 • bit 4 = AD4/DIO4 |
| Variable | Sampled Data Set | <p>If any digital IO lines are enabled, the first two bytes of the data set indicate the state of all enabled digital IO. Only digital channels that are enabled in the Digital Channel Mask bytes have any meaning in the sample set. If no digital IO are enabled on the device, these 2 bytes will be omitted.</p> <p>Following the digital IO data (if any), each enabled analog channel will return 2 bytes. The data starts with AIN0 and continues sequentially for each enabled analog input channel up to AIN5.</p> |

If the IS command is issued from AT command mode then a carriage return delimited list will be returned containing the above-listed fields. If the command is issued via an API frame then the module will return an AT command response API frame with the IO data included in the command data portion of the packet.

| Example | Sample AT Response |
|----------|--|
| 0x01\r | [1 sample set] |
| 0x0C0C\r | [Digital Inputs: DIO 2, 3, 10, 11 enabled] |
| 0x03\r | [Analog Inputs: A/D 0, 1 enabled] |

| Example | Sample AT Response |
|----------|---|
| 0x0408\r | [Digital input states: DIO 3, 10 high, DIO 2, 11 low] |
| 0x03D0\r | [Analog input ADIO 0= 0x3D0] |
| 0x0124\r | [Analog input ADIO 1=0x120] |

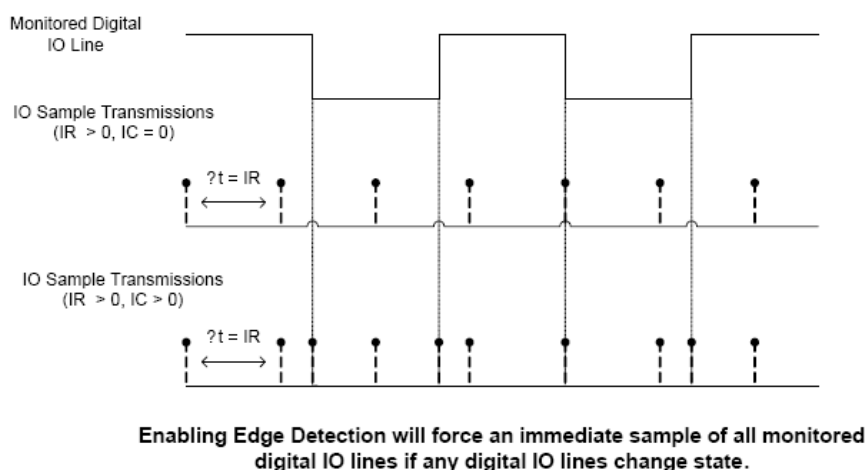
Periodic I/O Sampling

Periodic sampling allows an XBee-PRO module to take an I/O sample and transmit it to a remote device at a periodic rate. The periodic sample rate is set by the IR command. If IR is set to 0, periodic sampling is disabled. For all other values of IR, data will be sampled after IR milliseconds have elapsed and transmitted to a remote device. The DH and DL commands determine the destination address of the IO samples. Only devices with API mode enabled will send IO data samples out their serial interface. Devices not in API mode will discard received IO data samples.

A module with sleep enabled will transmit periodic I/O samples at the IR rate until the ST time expires and the device can resume sleeping. See the sleep section for more information on sleep.

Digital I/O Change Detection

Modules can be configured to transmit a data sample immediately whenever a monitored digital I/O pin changes state. The IC command is a bitmask that can be used to set which digital I/O lines should be monitored for a state change. If one or more bits in IC is set, an I/O sample will be transmitted as soon as a state change is observed in one of the monitored digital I/O lines. The figure below shows how edge detection can work with periodic sampling.



General Purpose Flash Memory

XBee-PRO 900HP modules provide 119 512-byte blocks of flash memory which can be read and written by the user application. This memory provides a non-volatile data storage area which can be used for a multitude of purposes. Some common uses of this data storage include: storing logged sensor data, buffering firmware upgrade data for a host microcontroller, or storing and retrieving data tables needed for calculations performed by a host microcontroller. The General Purpose Memory (GPM) is also used to store a firmware upgrade file for over-the-air firmware upgrades of the XBee module itself.

Accessing General Purpose Flash Memory

The GPM of a target node can be accessed locally or over-the-air by sending commands to the MEMORY_ACCESS cluster ID (0x23) on the DIGI_DEVICE endpoint (0xE6) of the target node using explicit API frames. (Explicit API frames are described in the API Operation section.)

To issue a GPM command the payload of an explicit API frame should be formatted in the following way:

| Byte Offset in Payload | Number of Bytes | Field Name | General Field Description |
|------------------------|-----------------|-----------------|--|
| 0 | 1 | GPM_CMD_ID | Specific GPM commands are described below |
| 1 | 1 | GPM_OPTIONS | Command-specific options |
| 2 | 2* | GPM_BLOCK_NUM | The block number addressed in the GPM |
| 4 | 2* | GPM_START_INDEX | The byte index within the addressed GPM block |
| 6 | 2* | GPM_NUM_BYTES | The number of bytes in the GPM_DATA field, or in the case of a READ, the number of bytes requested |
| 8 | varies | GPM_DATA | |

*Multi-byte parameters should be specified with big-endian byte ordering.

When a GPM command is sent to a radio via a unicast the receiving radio will unicast a response back to the requesting radio's source endpoint specified in the request packet. No response is sent for broadcast requests. If the source endpoint is set to the DIGI_DEVICE endpoint (0xE6) or explicit API mode is enabled on the requesting radio then a GPM response will be output as an explicit API RX indicator frame on the requesting node (assuming API mode is enabled.)

The format of the response is very similar to the request packet:

| Byte Offset in Payload | Number of Bytes | Field Name | General Field Description |
|------------------------|-----------------|-----------------|--|
| 0 | 1 | GPM_CMD_ID | This field will be the same as the request field |
| 1 | 1 | GPM_STATUS | Status indicating whether the command was successful |
| 2 | 2* | GPM_BLOCK_NUM | The block number addressed in the GPM |
| 4 | 2* | GPM_START_INDEX | The byte index within the addressed GPM block |
| 6 | 2* | GPM_NUM_BYTES | The number of bytes in the GPM_DATA field |
| 8 | varies | GPM_DATA | |

*Multi-byte parameters should be specified with big-endian byte ordering.

The following commands exist for interacting with GPM:

PLATFORM_INFO_REQUEST (0x00):

A PLATFORM_INFO_REQUEST frame can be sent to query details of the GPM structure.

| Field Name | Command-Specific Description |
|-----------------|---|
| GPM_CMD_ID | Should be set to PLATFORM_INFO_REQUEST (0x00) |
| GPM_OPTIONS | This field is unused for this command. Set to 0. |
| GPM_BLOCK_NUM | This field is unused for this command. Set to 0. |
| GPM_START_INDEX | This field is unused for this command. Set to 0. |
| GPM_NUM_BYTES | This field is unused for this command. Set to 0. |
| GPM_DATA | No data bytes should be specified for this command. |

PLATFORM_INFO (0x80):

When a PLATFORM_INFO_REQUEST command request has been unicast to a node, that node will send a response in the following format to the source endpoint specified in the requesting frame.

| Field Name | Command-Specific Description |
|-----------------|---|
| GPM_CMD_ID | Should be set to PLATFORM_INFO (0x80) |
| GPM_STATUS | A 1 in the least significant bit indicates an error occurred. All other bits are reserved at this time. |
| GPM_BLOCK_NUM | Indicates the number of GPM blocks available. |
| GPM_START_INDEX | Indicates the size, in bytes, of a GPM block. |
| GPM_NUM_BYTES | The number of bytes in the GPM_DATA field. For this command, this field will be set to 0. |
| GPM_DATA | No data bytes are specified for this command. |

Example:

A PLATFORM_INFO_REQUEST sent to a radio with a serial number of 0x0013a200407402AC should be formatted as follows (spaces added to delineate fields):

```
7E 001C 11 01 0013A200407402AC FFFE E6 E6 0023 C105 00 00 00 00 0000 0000 0000 24
```

Assuming all transmissions were successful, the following API packets would be output the source node's serial interface:

```
7E 0007 8B 01 FFFE 00 00 00 76
```

```
7E 001A 91 0013A200407402AC FFFE E6 E6 0023 C105 C1 80 00 0077 0200 0000 EB
```

ERASE (0x01):

The ERASE command erases (writes all bits to binary 1) one or all of the GPM flash blocks. The ERASE command can also be used to erase all blocks of the GPM by setting the GPM_NUM_BYTES field to 0.

| Field Name | Command-Specific Description |
|-----------------|---|
| GPM_CMD_ID | Should be set to ERASE (0x01) |
| GPM_OPTIONS | There are currently no options defined for the ERASE command. Set this field to 0. |
| GPM_BLOCK_NUM | Set to the index of the GPM block that should be erased. When erasing all GPM blocks, this field is ignored (set to 0). |
| GPM_START_INDEX | The ERASE command only works on complete GPM blocks. The command cannot be used to erase part of a GPM block. For this reason GPM_START_INDEX is unused (set to 0). |
| GPM_NUM_BYTES | Setting GPM_NUM_BYTES to 0 has a special meaning. It indicates that every flash block in the GPM should be erased (not just the one specified with GPM_BLOCK_NUM). In all other cases, the GPM_NUM_BYTES field should be set to the GPM flash block size. |
| GPM_DATA | No data bytes are specified for this command. |

ERASE_RESPONSE (0x81):

When an ERASE command request has been unicast to a node, that node will send a response in the following format to the source endpoint specified in the requesting frame.

| Field Name | Command-Specific Description |
|------------|--|
| GPM_CMD_ID | Should be set to ERASE_RESPONSE (0x81) |

| Field Name | Command-Specific Description |
|-----------------|---|
| GPM_STATUS | A 1 in the least significant bit indicates an error occurred. All other bits are reserved at this time. |
| GPM_BLOCK_NUM | Matches the parameter passed in the request frame. |
| GPM_START_INDEX | Matches the parameter passed in the request frame. |
| GPM_NUM_BYTES | The number of bytes in the GPM_DATA field. For this command, this field will be set to 0. |
| GPM_DATA | No data bytes are specified for this command. |

Example:

To erase flash block 42 of a target radio with serial number of 0x0013a200407402ac an ERASE packet should be formatted as follows (spaces added to delineate fields):

```
7E 001C 11 01 0013A200407402AC FFFE E6 E6 0023 C105 00 C0 01 00 002A 0000 0200 37
```

Assuming all transmissions were successful, the following API packets would be output the source node's serial interface:

```
7E 0007 8B 01 FFFE 00 00 00 76
```

```
7E 001A 91 0013A200407402AC FFFE E6 E6 0023 C105 C1 81 00 002A 0000 0000 39
```

WRITE (0x02) and ERASE_THEN_WRITE (0x03):

The WRITE command writes the specified bytes to the GPM location specified. Before writing bytes to a GPM block it is important that the bytes have been erased previously. The ERASE_THEN_WRITE command performs an ERASE of the entire GPM block specified with the GPM_BLOCK_NUM field prior to doing a WRITE.

| Field Name | Command-Specific Description |
|-----------------|--|
| GPM_CMD_ID | Should be set to WRITE (0x02) or ERASE_THEN_WRITE (0x03) |
| GPM_OPTIONS | There are currently no options defined for this command. Set this field to 0. |
| GPM_BLOCK_NUM | Set to the index of the GPM block that should be written. |
| GPM_START_INDEX | Set to the byte index within the GPM block where the given data should be written. |
| GPM_NUM_BYTES | Set to the number of bytes specified in the GPM_DATA field. Only one GPM block can be operated on per command. For this reason, GPM_START_INDEX + GPM_NUM_BYTES cannot be greater than the GPM block size. It is also important to remember that the number of bytes sent in an explicit API frame (including the GPM command fields) cannot exceed the maximum payload size of the radio. The maximum payload size can be queried with the NP AT command. |
| GPM_DATA | The data to be written. |

WRITE_RESPONSE (0x82) and ERASE_THEN_WRITE_RESPONSE(0x83):

When a WRITE or ERASE_THEN_WRITE command request has been unicast to a node, that node will send a response in the following format to the source endpoint specified in the requesting frame.

| Field Name | Command-Specific Description |
|---------------|---|
| GPM_CMD_ID | Should be set to WRITE_RESPONSE (0x82) or ERASE_THEN_WRITE_RESPONSE (0x83) |
| GPM_STATUS | A 1 in the least significant bit indicates an error occurred. All other bits are reserved at this time. |
| GPM_BLOCK_NUM | Matches the parameter passed in the request frame. |

| Field Name | Command-Specific Description |
|-----------------|---|
| GPM_START_INDEX | Matches the parameter passed in the request frame. |
| GPM_NUM_BYTES | The number of bytes in the GPM_DATA field. For this command, this field will be set to 0. |
| GPM_DATA | No data bytes are specified for these commands. |

Example:

To write 15 bytes of incrementing data to flash block 22 of a target radio with serial number of 0x0013a200407402ac a WRITE packet should be formatted as follows (spaces added to delineate fields):

```
7E 002B 11 01 0013A200407402AC FFFE E6 E6 0023 C105 00 C0 02 00 0016 0000 000F 0102030405060708090A0B0C0D0E0F C5
```

Assuming all transmissions were successful and that flash block 22 was previously erased, the following API packets would be output the source node's serial interface:

```
7E 0007 8B 01 FFFE 00 00 00 76
```

```
7E 001A 91 0013A200407402AC FFFE E6 E6 0023 C105 C1 82 00 0016 0000 0000 4C
```

READ (0x04):

The READ command can be used to read the specified number of bytes from the GPM location specified. Data can be queried from only one GPM block per command.

| Field Name | Command-Specific Description |
|-----------------|--|
| GPM_CMD_ID | Should be set to READ (0x04) |
| GPM_OPTIONS | There are currently no options defined for this command. Set this field to 0. |
| GPM_BLOCK_NUM | Set to the index of the GPM block that should be read. |
| GPM_START_INDEX | Set to the byte index within the GPM block where the given data should be read. |
| GPM_NUM_BYTES | Set to the number of data bytes to be read. Only one GPM block can be operated on per command. For this reason, GPM_START_INDEX + GPM_NUM_BYTES cannot be greater than the GPM block size. It is also important to remember that the number of bytes sent in an explicit API frame (including the GPM command fields) cannot exceed the maximum payload size of the radio. The maximum payload size can be queried with the NP AT command. |
| GPM_DATA | No data bytes should be specified for this command. |

READ_RESPONSE (0x84):

When a READ command request has been unicast to a node, that node will send a response in the following format to the source endpoint specified in the requesting frame.

| Field Name | Command-Specific Description |
|-----------------|---|
| GPM_CMD_ID | Should be set to READ_RESPONSE (0x84) |
| GPM_STATUS | A 1 in the least significant bit indicates an error occurred. All other bits are reserved at this time. |
| GPM_BLOCK_NUM | Matches the parameter passed in the request frame. |
| GPM_START_INDEX | Matches the parameter passed in the request frame. |
| GPM_NUM_BYTES | The number of bytes in the GPM_DATA field. |
| GPM_DATA | The bytes read from the GPM block specified. |

Example:

To read 15 bytes of previously written data from flash block 22 of a target radio with serial number of 0x0013a200407402ac a READ packet should be formatted as follows (spaces added to delineate fields):

```
7E 001C 11 01 0013A200407402AC FFFE E6 E6 0023 C105 00 C0 04 00 0016 0000 000F 3B
```

Assuming all transmissions were successful and that flash block 22 was previously written with incrementing data, the following API packets would be output the source node's serial interface:

```
7E 0007 8B 01 FFFE 00 00 00 76
```

```
7E 0029 91 0013A200407402AC FFFE E6 E6 0023 C105 C1 84 00 0016 0000 000F 0102030405060708090A0B0C0D0E0F C3
```

FIRMWARE_VERIFY (0x05) and FIRMWARE_VERIFY_AND_INSTALL(0x06):

The FIRMWARE_VERIFY and FIRMWARE_VERIFY_AND_INSTALL commands are used when remotely updating firmware on a module. Remote firmware upgrades are covered in detail in the next section. These commands check if the General Purpose Memory contains a valid over-the-air update file. For the FIRMWARE_VERIFY_AND_INSTALL command, if the GPM contains a valid firmware image then the module will reset and begin using the new firmware.

| Field Name | Command-Specific Description |
|-----------------|---|
| GPM_CMD_ID | Should be set to FIRMWARE_VERIFY (0x05) or FIRMWARE_VERIFY_AND_INSTALL (0x06) |
| GPM_OPTIONS | There are currently no options defined for this command. Set this field to 0. |
| GPM_BLOCK_NUM | This field is unused for this command. Set to 0. |
| GPM_START_INDEX | This field is unused for this command. Set to 0. |
| GPM_NUM_BYTES | This field is unused for this command. Set to 0. |
| GPM_DATA | This field is unused for this command. |

FIRMWARE_VERIFY_RESPONSE (0x85):

When a FIRMWARE_VERIFY command request has been unicast to a node, that node will send a response in the following format to the source endpoint specified in the requesting frame.

| Field Name | Command-Specific Description |
|-----------------|---|
| GPM_CMD_ID | Should be set to FIRMWARE_VERIFY_RESPONSE (0x85) |
| GPM_STATUS | A 1 in the least significant bit indicates the GPM does not contain a valid firmware image. A 0 in the least significant bit indicates the GPM does contain a valid firmware image. All other bits are reserved at this time. |
| GPM_BLOCK_NUM | This field is unused for this command. Set to 0. |
| GPM_START_INDEX | This field is unused for this command. Set to 0. |
| GPM_NUM_BYTES | This field is unused for this command. Set to 0. |
| GPM_DATA | This field is unused for this command. |

FIRMWARE_VERIFY_AND_INSTALL_RESPONSE (0x86):

When a FIRMWARE_VERIFY_AND_INSTALL command request has been unicast to a node, that node will send a response in the following format to the source endpoint specified in the requesting frame only if the GPM memory does not contain a valid image. If the image is valid, the module will reset and begin using the new firmware.

| Field Name | Command-Specific Description |
|-----------------|---|
| GPM_CMD_ID | Should be set to FIRMWARE_VERIFY_AND_INSTALL_RESPONSE (0x86) |
| GPM_STATUS | A 1 in the least significant bit indicates the GPM does not contain a valid firmware image. All other bits are reserved at this time. |
| GPM_BLOCK_NUM | This field is unused for this command. Set to 0. |
| GPM_START_INDEX | This field is unused for this command. Set to 0. |
| GPM_NUM_BYTES | This field is unused for this command. Set to 0. |
| GPM_DATA | This field is unused for this command. |

Example:

To verify a firmware image previously loaded into the GPM on a target radio with serial number of 0x0013a200407402ac a FIRMWARE_VERIFY packet should be formatted as follows (spaces added to delineate fields):

```
7E 001C 11 01 0013A200407402AC FFFE E6 E6 0023 C105 00 00 05 00 0000 0000 0000 1F
```

Assuming all transmissions were successful and that the firmware image previously loaded into the GPM is valid, the following API packets would be output the source node's serial interface:

```
7E 0007 8B 01 FFFE 00 00 00 76
```

```
7E 001A 91 0013A200407402AC FFFE E6 E6 0023 C105 C1 85 00 0000 0000 0000 5F
```

Working with Flash Memory

When working with the General Purpose Memory the user should be aware of a number of limitations associated with working with flash memory:

- Flash memory write operations are only capable of changing binary 1's to binary 0's. Only the erase operation can change binary 0's to binary 1's. For this reason it is usually necessary to erase a flash block before performing a write operation.
- A flash memory block must be erased in its entirety when performing an erase operation. A block cannot be partially erased.
- Flash memory has a limited lifetime. The flash memory on which the GPM is based is rated at 20,000 erase cycles before failure. Care must be taken to ensure that the frequency of erase/write operations allows for the desired product lifetime. Digi's warranty will not cover products whose number of erase cycles has been exceeded.
- Over-the-Air firmware upgrades (described in the next section) require the entire GPM be erased. Any user data stored in the GPM will be lost during an over-the-air upgrade.

Over-the-Air Firmware Upgrades

XBee-PRO 900HP modules provide two methods of updating the firmware on the module. Firmware can be updated locally via X-CTU (a free testing and configuration utility provided by Digi) using the radio's serial port interface. Firmware can also be updated using the radios' RF interface (Over-the-Air Updating.)

The over-the-air firmware upgrading method provided is a robust and versatile technique which can be tailored to many different networks and applications. It has been engineered to be reliable and minimize disruption of normal network operations.

There are three phases of the over-the-air upgrade process: distributing the new application, verifying the new application, and installing the new application. In the following section the node which will be upgraded will be referred to as the target node. The node providing the update information will be referred to as the source node. In most applications the source node will be locally attached to a PC running update software.

Distributing the New Application

The first phase of performing an over-the-air upgrade on a module is transferring the new firmware file to the target node. The new firmware image should be loaded in the target node's GPM prior to installation. XBee-PRO 900HP modules use an encrypted binary (.ebin) file for both serial and over-the-air firmware upgrades. These firmware files are available on the Digi Support website.

The contents of the .ebin file should be sent to the target radio using general purpose memory WRITE commands. The entire GPM should be erased prior to beginning an upload of an .ebin file. The contents of the .ebin file should be stored in order in the appropriate GPM memory blocks. The number of bytes that are sent in an individual GPM WRITE frame is flexible and can be catered to the user application.

Example:

XBee-PRO 900HP firmware version 8060 has an .ebin file of 55,141 bytes in length. Based on network traffic it was determined that sending a 128 byte packet every 30 seconds minimized network disruption. For this reason the .ebin should be divided and addressed as follows:

| GPM_BLOCK_NUM | GPM_START_INDEX | GPM_NUM_BYTES | .ebin bytes |
|---------------|-----------------|---------------|----------------|
| 0 | 0 | 128 | 0 to 127 |
| 0 | 128 | 128 | 128 to 255 |
| 0 | 256 | 128 | 256 to 383 |
| 0 | 384 | 128 | 384 to 511 |
| 1 | 0 | 128 | 512 to 639 |
| 1 | 128 | 128 | 640 to 767 |
| - | - | - | - |
| - | - | - | - |
| - | - | - | - |
| 107 | 0 | | 54784 to 54911 |
| 107 | 128 | | 54912 to 55039 |
| 107 | 256 | 101 | 55040 to 55140 |

Verifying the New Application

For an uploaded application to function correctly every single byte from the .ebin file must be properly transferred to the GPM. To guarantee that this is the case GPM VERIFY functions exist to ensure that all bytes are properly in place. The FIRMWARE_VERIFY function reports whether or not the uploaded data is valid. The FIRMWARE_VERIFY_AND_INSTALL command will report if the uploaded data is invalid. If the data is valid it will begin installing the application. No installation will take place on invalid data.

Installing the Application

When the entire .ebin file has been uploaded to the GPM of the target node a FIRMWARE_VERIFY_AND_INSTALL command can be issued. Once the target receives the command it will verify the .ebin file loaded in the GPM. If it is found to be valid then the module will install the new firmware. This installation process can take up to 8 seconds. During the installation the module will be unresponsive to both serial and RF communication. To complete the installation the target module will reset. AT parameter settings which have not been written to flash (using the WR command) will be lost.

Things to Remember

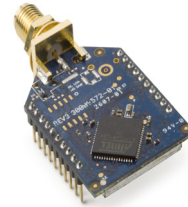
- The firmware upgrade process requires that the module resets itself. Because of this reset parameters which have not been written to flash will be lost after the reset. To avoid this, write all parameters with the WR command before doing a firmware upgrade. Packet routing information will also be lost after this reset. Route discoveries will be necessary for DigiMesh unicasts involving the upgraded node as a source, destination, or intermediate node.
- Because explicit API Tx frames can be addressed to a local node (accessible via the SPI or UART) or a remote node (accessible over the RF port) the same process can be used to update firmware on a module in either case.

Appendix A: XSC Firmware

XBee-PRO® 900HP/XBee-PRO® XSC RF Module

The XBee-PRO XSC (900 MHz) RF Modules were engineered to afford RF Modules and integrators an easy-to-use RF solution that provides reliable delivery of critical data between remote devices. These modules come configured to sustain reliable long-range wireless links. The XBee Module is a drop-in wireless solution that transfers a standard asynchronous serial data stream.

The S3 hardware variant is a legacy design and will become obsolete. New and old designs should use the S3B hardware variant, which features better performance, lower current draw, and is backward compatible with and a direct replacement for S3 radios. The S3B hardware with XSC firmware is also fully backward compatible (serial interface and over-the-air) with the 9XStream radios.



Key Features

Long Range Data Integrity

XBee-PRO XSC-S3:

- Indoor/Urban: 1200' (370m)
- Outdoor line-of-sight: Up to 6 miles (9.6 km)
- Outdoor line-of-sight: Up to 15 miles (24 km) w/ high gain antenna
- Receiver Sensitivity: -106 dBm,

XBee-PRO XSC-S3B:

- Indoor/Urban range: 2000' (610 m)
- Outdoor line-of-sight range: 9 miles (14 km)
- Receiver Sensitivity: -109 dBm

Advanced Networking & Security

- True peer-to-peer (no "master" required) communications
- Point-to-point & point-to-multipoint topologies supported
- Retries and Acknowledgements
- 7 hopping channels, each with over 65,000 available network addresses
- FHSS (Frequency Hopping Spread Spectrum)

Easy-to-Use

- No configuration required for out-of-the-box RF data communications
- Advanced configurations available through standard AT & binary commands
- Portable (small form factor easily designed into a wide range of data radio systems)
- Software-selectable serial interface baud rates
- I/O Support: CTS, RTS (& more)
- Support for multiple data formats (parity, start and stop bits, etc.)
- Power-saving Sleep Modes

Worldwide Acceptance

FCC Certified (USA) - Refer to Appendices B and C for FCC Requirements.

Systems that include XBee-PRO Modules inherit Digi's FCC Certification

ISM (Industrial, Scientific & Medical) frequency band

Manufactured under **ISO 9001:2000** registered standards

XBee-PRO™ XSC (900 MHz) RF Modules are approved for use in **US** and **Canada**.

RoHS compliant



Specifications

Table 1-01. XBee-PRO XSC RF Module Specifications

| Specification | XBee-PRO XSC (S3 Hardware) | XBee-PRO XSC (S3B Hardware) |
|-----------------------------|---|--|
| Performance | | |
| Indoor/Urban Range | Up to 1200ft (370m) | up to 2000ft (610m) |
| Outdoor line-of-sight Range | Up to 6 miles (9.6km) w/ dipole antenna Up to 15 miles (24km) w/ high-gain antenna | Up to 9 miles (14km) w/ dipole antenna Up to 28 miles (45km) w/ high-gain antenna |
| Interface Data Rate | 125 - 65,000 bps (Software selectable, includes non-standard baud rates) | |
| Throughput Data Rate | 9,600 bps | 9.6kbps or 19.2kbps |
| RF Data Rate | 10kbps | 10kbps or 20kbps |
| Transmit Power Output | +20dBm (100mW) | Up to 24dBm (250mW) software selectable |

Table 1-01. XBee-PRO XSC RF Module Specifications

| Specification | XBee-PRO XSC (S3 Hardware) | XBee-PRO XSC (S3B Hardware) |
|---|--|---|
| Receiver Sensitivity | -106dBm | -109dBm at 9600 baud -107dBm at 19200 baud |
| Power Requirements | | |
| Supply Voltage | 3.0-3.6 VDC regulated | 2.1 to 3.6VDC |
| Receive Current | 65mA | 26mA typical |
| Transmit Current | 265mA | 215mA at 24dBm |
| Power Down Current | 50uA | 2.5uA typical @3.3v |
| General | | |
| Frequency Range | 902-928MHz (located in the 900MHz ISM Band) | |
| Spread Spectrum | Frequency Hopping | |
| Network Topology | Point-to-Point, Peer-to-Peer, Point-to-Multipoint | |
| Channel Capacity | 7 hop sequences share 25 frequencies | |
| Board-level Serial Data Interface (S3B) | 3V CMOS UART (5V-tolerant) | 3V CMOS UART |
| Physical Properties | | |
| Module Board Size | 1.297" x 0.962" x 0.215 (3.29cm x 2.44cm x 0.546cm) Note: Dimensions do not include connector/antenna or pin lengths | |
| Weight | 5 to 8 grams, depending on the antenna option | |
| Connector | 2 rows of 10 pins, 22mm apart with 2mm spaced male Berg-type headers | |
| Operating Temperature | -40 to 85° C (industrial) | |
| Antenna Options | | |
| Integrated Wire | ¼ wave monopole, 3.25" (8.26cm) length, 1.9dBi Gain | |
| RF Connector | Reverse-polarity SMA or U.FL | |
| Impedance | 50 ohms unbalanced | |
| Certifications | | |
| FCC Part 15.247 | MCQ-XBEEEXSC | MCQ-XBPS3B, or MCQ-XB900HP (see preface) |
| Industry Canada (IC) | 1846A-XBEEEXSC | 1846A-XBPS3B, or 1846A-XB900HP (see preface) |
| Europe | N/A | |
| RoHS | Compliant | |
| Australia | N/A | C-Tick |

Pin Signals

XBee-PRO XSC RF Module Pin Numbers (top view, shield underneath)

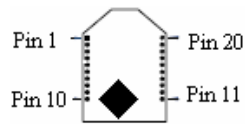
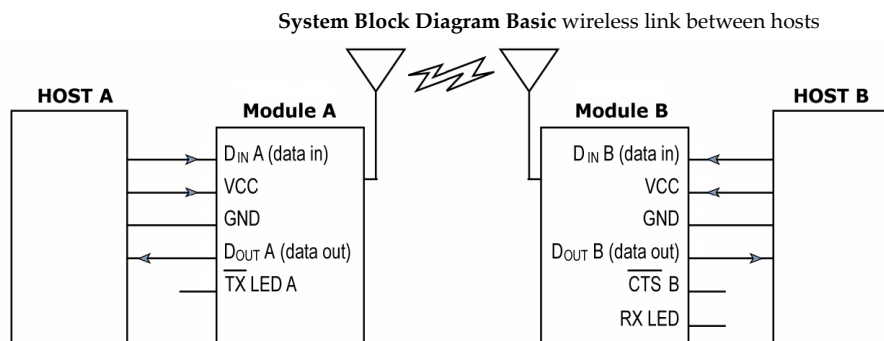


Table 1-02. J1 Pin Signal Descriptions
(Low-asserted signals distinguished with a horizontal line over signal name.)

| Module Pin | Public Signal | Notes | I/O | When Active | Function |
|------------|---|---------------------------|-----|-------------|---|
| 1 | VCC | | I | | Supply Voltage |
| 2 | DO (Data Out) | | O | n/a | Serial data exiting the module (to the UART host). Refer to the Serial Communications section for more information |
| 3 | DI (Data In) | | I | n/a | Serial data entering the module (from UART host). Refer to the Serial Communications section for more information. |
| 4 | DO3 / RX LED | | O | high | Pin is driven high during RF data reception; otherwise, the pin is driven low. Refer to the CD Command section to enable. |
| 5 | $\overline{\text{Reset}}$ | **Has a pull up resistor | I/O | low | Re-boot module.(minimum pulse is 90us) Open Drain configuration. Module will drive reset line low momentarily on reboot and power up. |
| 6 | $\overline{\text{Config}}$ | *Has a pull up resistor | I | low / high | Pin can be used as a backup method for entering Command Mode during power-up. Primary method is with "+++". Refer to the AT Commands section for more information. |
| 7 | | | O | Driven high | Do not Connect |
| 8 | | | NC | | Do not Connect |
| 9 | DI3 / SLEEP | *Has a pull up resistor | I | high | By default, DI3 pin is not used. To configure this pin to support Sleep Modes, refer to the Sleep Mode, SM Command and PW Command sections. |
| 10 | GND | | | | Ground |
| 11 | | | O | Driven high | Do not Connect |
| 12 | DO2 / $\overline{\text{CTS}}$ / RS-485 Enable | | O | low | <p>$\overline{\text{CTS}}$ (clear-to-send) flow control - When pin is driven low, UART host is permitted to send serial data to the module. Refer to the Serial Communications and CS Command sections for more information.</p> <p>-----</p> <p>RS-485 Enable - To configure this pin to enable RS-485 (2-wire or 4-wire) communications, refer to the Serial Communications and CS Command sections.</p> |
| 13 | ON / $\overline{\text{Sleep}}$ | | O | high | high = Indicates power is on and module is not in Sleep Mode. Low = Sleep mode or module is unpowered |
| 14 | VREF | | I | n/a | Not used on this module. For compatibility with other XBee modules, we recommend connecting this pin to a voltage reference if Analog sampling is desired. Otherwise, connect to GND. |
| 15 | $\overline{\text{TX}}$ / PWR | | O | n/a | <p>low = $\overline{\text{TX}}$ - Pin pulses low during transmission</p> <p>-----</p> <p>high = PWR - Indicates power is on and module is not in Sleep Mode</p> |
| 16 | DI2 / $\overline{\text{RTS}}$ / CMD | *Has a pull down resistor | I | low | <p>RTS (request-to-send) flow control - By default, this pin is not used. To configure this pin to regulate the flow of serial data exiting the module, refer to the Serial Communications and RT Command sections.</p> <p>-----</p> <p>CMD -Refer to Binary Commands and RT Command sections to enable binary command programming.</p> |
| 17 | | | O | Driven low | Do not Connect |
| 18 | | | O | Driven low | Do not Connect |
| 19 | | | O | Driven low | Do not Connect |
| 20 | | | O | Driven low | Do not Connect |

Note:*S3 has a 100k pull-up. S3B has internal pull-up. **S3 has 10k pull-up. S3B has internal pull-up.

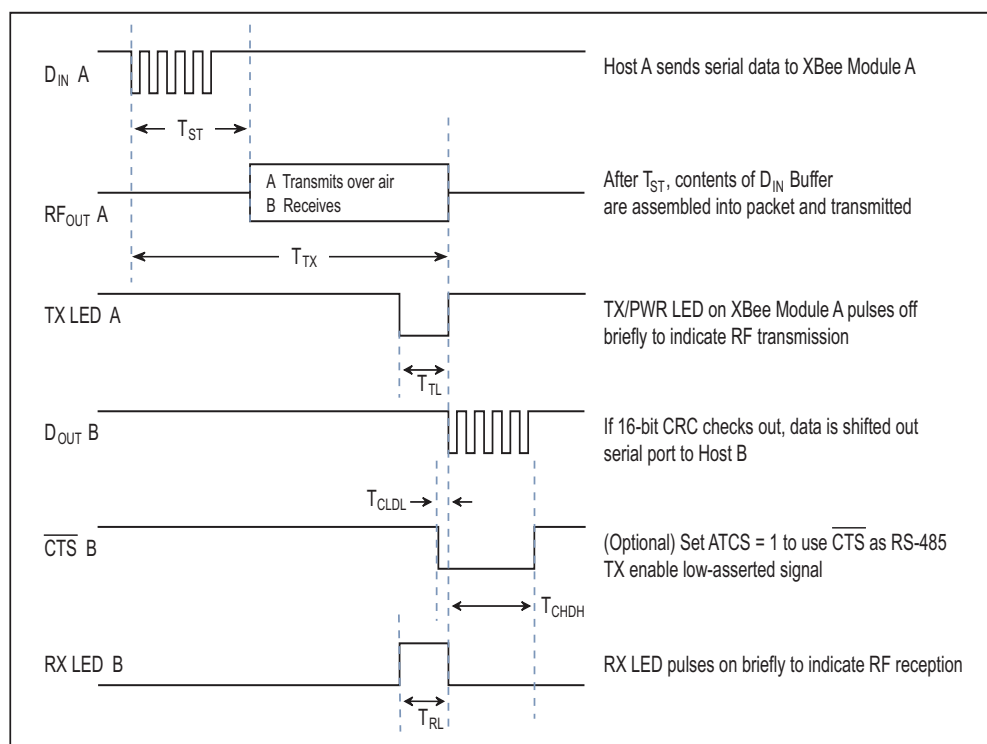
Electrical Characteristics



The data flow sequence is initiated when the first byte of data is received in the DI Buffer of the transmitting module (XBee Module A). As long as XBee Module A is not already receiving RF data, data in the DI Buffer is packetized, then transmitted over-the-air to XBee Module B.

Timing Specifications

Timing Specifications ("A" and "B" refer to Figure .)



Typical AC Characteristics (SY parameter = 0, symbols correspond to Figure and Figure .)

| Symbol | Description | 9600 baud rate (32 byte packet) |
|----------|--|------------------------------------|
| T_{TX} | Latency from the time data is transmitted until received | 72.0 ms |
| T_{TL} | Time that TX/PWR pin is driven low | 16.8 ms |
| T_{RL} | Time that RX LED pin is driven high | 25.6 ms |
| T_{ST} | Channel Initialization Time | 35.0 ms |

DC Characteristics ($V_{CC} = 3.0\text{--}3.6\text{ VDC}$)

| Symbol | Parameter | Condition | Min | Typical | Max | Units |
|----------|---------------------------|---------------------------------------|----------------|---------|---------------------|-------|
| V_{CC} | Module Supply Voltage | | *3.0 | | 3.6 | V |
| V_{IL} | Input Low Voltage | All input signals | -0.3 | | 0.3 V_{CC} | V |
| V_{IH} | Input High voltage | All input signals | 0.7 V_{CC} | | $V_{CC} + 0.3^{**}$ | V |
| V_{OL} | Output Low-Level Voltage | $I_{out} = I_{out_Max}$ | | | 0.4 | V |
| V_{OH} | Output High-Level Voltage | $I_{out} = I_{out_Max}$ | $V_{CC} - 0.4$ | | | V |
| I_L | Input Leakage Current | ***With Pull-up resistors disabled | | 40 | 400 | nA |
| I_{O1} | Output Current | pins 2, 15 (Dout, ~TX/Pwr) | | | 2 | mA |
| I_{O2} | Output Current | pins 4, 12, 13 (DCD, ~CTS, ON/~Sleep) | | | 8 | mA |

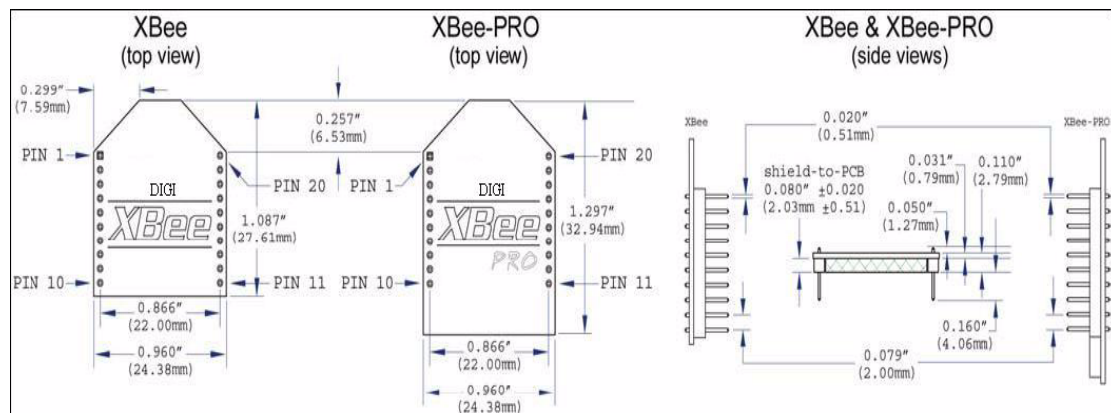
Note: *Min Voltage for S3B is 2.1v, however Max Power will be reduced and Sensitivity may degrade.

**S3 is tolerant up to 5.5v on input pins.

***S3B can have pull-ups enabled and still maintain low leakage current.

Mechanical Drawings

Mechanical Drawings



RF Module Operation

Serial Communications

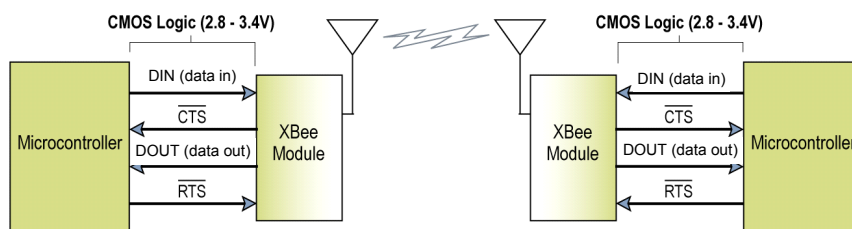
The XBee module interfaces to a host device through a CMOS-level asynchronous serial port. Through its serial port, the module can communicate with any UART voltage compatible device or through a level translator to any RS-232/485/422 device.

UART-Interfaced Data Flow

Devices that have a UART interface can connect directly through the pins of the XBee module as shown in the figure below.

System Data Flow Diagram in a UART-interfaced environment

(Low-asserted signals distinguished with horizontal line over signal name.)

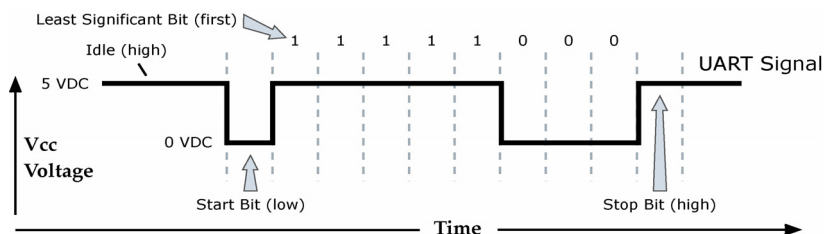


Serial Data

Data enters the XBee module through the DI pin as an asynchronous serial signal. The signal should idle high when no data is being transmitted.

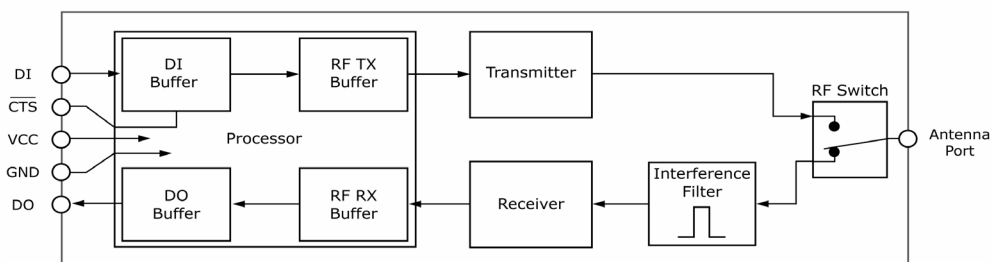
The UART performs tasks, such as timing and parity checking, that are needed for data communications. Serial communication consists of two UARTs, one being the XBee's and the other being the Microcontroller's, configured with compatible parameters (baud rate, parity, start bits, stop bits, data bits) to have successful communication. Each data packet consists of a start bit (low), 8 data bits (least significant bit first) and a stop bit (high). The following figure illustrates the serial bit pattern of data passing through the module.

UART data packet 0x1F (decimal number "31") as transmitted through the XBee Module
Example Data Format is 8-N-1 (bits - parity - # of stop bits)



Flow Control

Internal Data Flow Diagram (The five most commonly-used pin signals shown.)



DI (Data In) Buffer and Flow Control

When serial data enters the XBee module through the DI Pin, then the data is stored in the DI Buffer until it can be transmitted.

When the RO parameter threshold is satisfied (refer to Transmit Mode and Command Descriptions sections for more information), the module attempts to initialize an RF connection. If the module is already receiving RF data, the serial data is stored in the module's DI Buffer. If the DI buffer becomes full, hardware or software flow control must be implemented in order to prevent overflow (loss of data between the host and XBee RF Module).

How to eliminate the need for flow control:

- Send messages that are smaller than the DI buffer size, which is generally around 1,000 bytes.
- Interface at a lower baud rate (BD parameter) than the fixed RF data rate with the Retries functionality (RR parameter) disabled.

Two cases in which the DI Buffer may become full and possibly overflow:

- If the serial interface data rate is set higher than the RF data rate of the module, the module will receive data from the host faster than it can transmit the data over-the-air.
- If the module is receiving a continuous stream of data, monitoring data on a network, or awaiting acknowledgments for Retries functionality, any serial data that arrives on the DI pin is placed in the DI Buffer. The data in the DI buffer will be transmitted over-the-air when the module no longer detects RF data in the network.

Hardware Flow Control (CTS). When the DI buffer is 65 bytes away from being full; by default, the module de-asserts (high) $\overline{\text{CTS}}$ to signal to the host device to stop sending data [refer to FT (Flow Control Threshold) and CS (DO2 Configuration) Commands]. $\overline{\text{CTS}}$ is re-asserted after the DI Buffer has 34 bytes of memory available.

Software Flow Control (XON). XON/XOFF software flow control can be enabled using the FL (Software Flow Control) command.

DO (Data Out) Buffer and Flow Control

When RF data is received, the data enters the DO buffer and is then sent out the serial port to a host device. Once the DO Buffer reaches capacity, any additional incoming RF data is lost.

Two cases in which the DO Buffer may become full and possibly overflow:

- If the RF data rate is higher than the set interface data rate of the module, the module will receive data from the transmitting module faster than it can send the data to the host.
- If the host does not allow the RF module to send data out of the DO buffer because of hardware or software flow control.

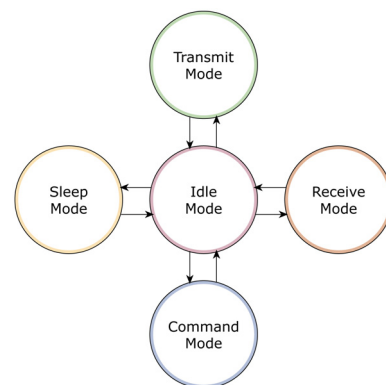
Hardware Flow Control (RTS). If $\overline{\text{RTS}}$ is enabled for flow control (RT Parameter = 2), data will not be sent out the DO Buffer as long as $\overline{\text{RTS}}$ (pin 16) is de-asserted.

Software Flow Control (XOFF). XON/XOFF software flow control can be enabled using the FL (Software Flow Control) Command. This option only works with ASCII data.

Modes of Operation

XBee-PRO® 900HP/XBee-PRO® XSC RF Modules operate in five modes.

Modes of Operation



Idle Mode

When not receiving or transmitting data, the RF module is in Idle Mode. The module shifts into the other modes of operation under the following conditions:

- Transmit Mode (Serial data is received in the DI Buffer)
- Receive Mode (Valid RF data is received through the antenna)
- Sleep Mode (Sleep Mode condition is met)
- Command Mode (Command Mode Sequence is issued)

Transmit Mode

When the first byte of serial data is received from the UART in the DI buffer, the modem attempts to shift to Transmit Mode and initiate an RF connection with other modems. After transmission is complete, the modem returns to Idle Mode.

RF transmission begins after either of the following criteria is met:

1. RB bytes have been received in the DI buffer and are pending for RF transmission [refer to RB (Packetization Threshold) command, p34].

- The RB parameter may be set to any value between 1 and the RF packet size (PK), inclusive. When RB = 0, the packetization threshold is ignored.

2. At least one character has been received in the DI buffer (pending for RF transmission) and RO time has been observed on the UART [refer to RO (Packetization Timeout) command].

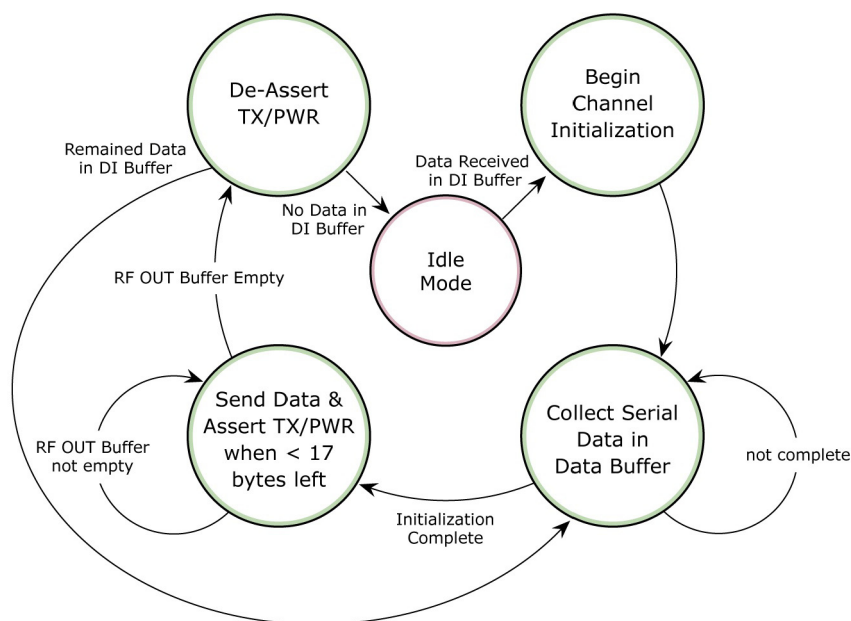
- The time out can be disabled by setting RO to zero. In this case, transmission will begin after RB bytes have been received in the DI buffer.

Note: RF reception must complete before the modem is able to enter into Transmit Mode.

After either RB or RO conditions are met, the modem then initializes a communications channel. [Channel initialization is the process of sending an RF initializer that synchronizes receiving modems with the transmitting modem. During channel initialization, incoming serial data accumulates in the DI buffer.]

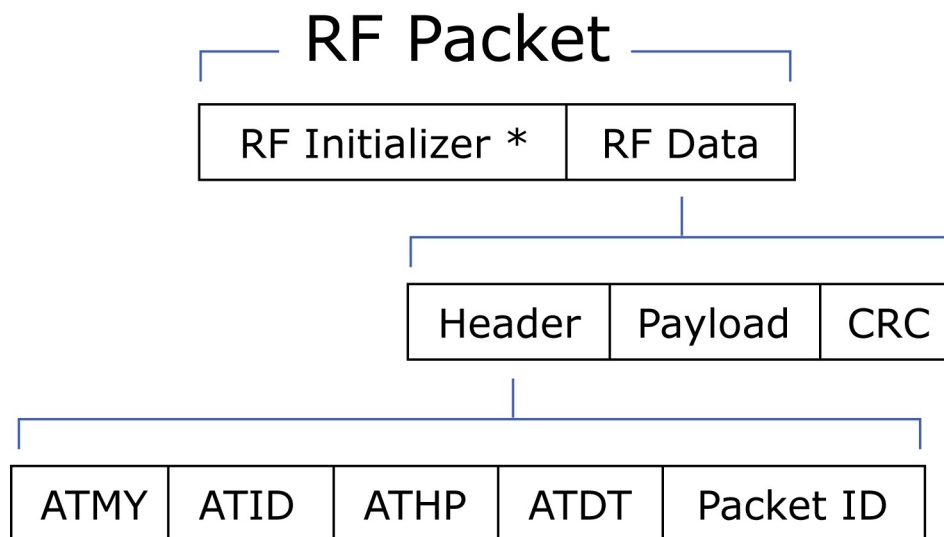
Serial data in the DI buffer is grouped into RF packets [refer to PK (RF Packet Size)]; converted to RF data; then transmitted over-the-air until the DI buffer is empty.

RF data, which includes the payload data, follows the RF initializer. The payload includes up to the maximum packet size (PK Command) bytes. As the transmitting modem nears the end of the transmission, it inspects the DI buffer to see if more data exists to be transmitted. This could be the case if more than PK bytes were originally pending in the DI buffer or if more bytes arrived from the UART after the transmission began. If more data is pending, the transmitting modem assembles a subsequent packet for transmission.



RF Packet

The RF packet is the sequence of data used for communicating information between Digi Radios. An RF Packet consists of an RF Initializer and RF Data.



When streaming multiple RF packets, the RF Initializer is only sent in front of the first packet.

RF Initializer

An RF initializer is sent each time a new connection sequence begins. The RF initializer contains channel information that notifies receiving modems of information such as the hopping pattern used by the transmitting modem. The first transmission always sends an RF initializer.

An RF initializer can be of various lengths depending on the amount of time determined to be required to prepare a receiving modem. For example, a wake-up initializer is a type of RF initializer used to wake remote modems from Sleep Mode (Refer to the FH, LH, HT and SM Commands for more information). The length of the wake-up initializer should be longer than the length of time remote modems are in cyclic sleep.

Header

The header contains network addressing information that filters incoming RF data. The receiving modem checks for a matching Hopping Channel (HP parameter), Vendor Identification Number (ID parameter) and Destination Address (DT parameter). Data that does not pass through all three network filter layers is discarded.

CRC (Cyclic Redundancy Check)

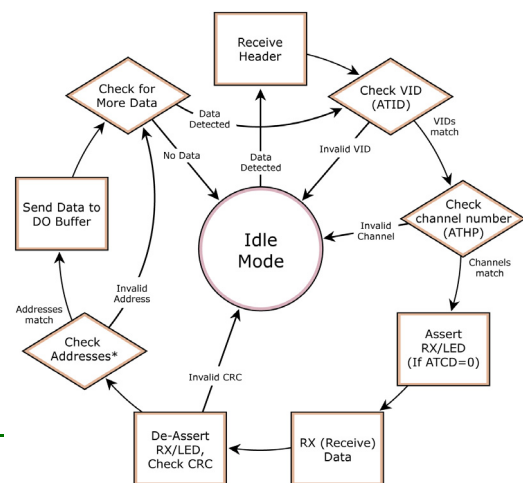
To verify data integrity and provide built-in error checking, a 16-bit CRC (Cyclic Redundancy Check) is computed for the transmitted data and attached to the end of each RF packet. On the receiving end, the receiving modem computes the CRC on all incoming RF data. Received data that has an invalid CRC is discarded.

Receive Mode

If a module detects RF data while operating in Idle Mode, the module transitions into Receive Mode to start receiving RF packets.

Reception of RF Data

After a packet is received, the module checks the CRC (cyclic redundancy check) to ensure that the data was transmitted without error. If the CRC data bits on the incoming packet are invalid, the packet is discarded. If the CRC is valid, the packet proceeds to the DO Buffer. The module returns to Idle Mode after valid RF data is no longer detected or after an error is detected in the received RF data. If serial data is stored in the DI buffer while the module is in Receive Mode, the serial data will be transmitted after the module is finished receiving data and returns to Idle Mode.



Sleep Mode

Sleep Modes enable the XBee module to operate at minimal power consumption when not in use. The following Sleep Mode options are available:

- Pin Sleep
- Cyclic Sleep

For the module to transition into Sleep Mode, the module must have a non-zero SM (Sleep Mode) Parameter and one of the following must occur:

- The module is idle (no data transmission or reception) for a user-defined period of time [Refer to the ST (Time before Sleep) Command].
- SLEEP is asserted (only for Pin Sleep option).

In Sleep Mode, the module will not transmit or receive data until the module first transitions to Idle Mode. All Sleep Modes are enabled and disabled using SM Command. Transitions into and out of Sleep Modes are triggered by various events as shown in the table below.

Summary of Sleep Mode Configurations

| Sleep Mode Setting | Transition into Sleep Mode | Transition out of Sleep Mode | Related Commands | Typical Power Consumption (S3) | Typical Power Consumption (S3B) |
|-------------------------|--|--|--------------------|--------------------------------|---------------------------------|
| Pin Sleep (SM = 1) | Microcontroller can shut down and wake modules by asserting (high) SLEEP (pin 9). Note: The module will complete a transmission or reception before activating Pin Sleep. | De-assert (low) SLEEP (pin 9). | SM | 50 μ A | 2.5uA |
| Cyclic Sleep (SM = 3-8) | Automatic transition to Sleep Mode occurs in cycles as defined by the SM (Sleep Mode) Command. Note: The cyclic sleep time interval must be shorter than the "Wake-up Initializer Timer" (set by LH Command). | After the cyclic sleep time interval elapses. Note: Module can be forced into Idle Mode if PW (Pin Wake-up) Command is enabled. | SM, ST, HT, LH, PW | 76 μ A when sleeping | 2.5uA when sleeping |

Pin Sleep (SM = 1)

In order to achieve this state, SLEEP pin must be asserted (high). The module remains in Pin Sleep until the SLEEP pin is de-asserted.

After enabling Pin Sleep, the SLEEP pin controls whether the XBee module is active or in Sleep Mode. When SLEEP is de-asserted (low), the module is fully operational. When SLEEP is asserted (high), the module transitions to Sleep Mode and remains in its lowest power-consuming state until the SLEEP pin is de-asserted. SLEEP is only active if the module is setup to operate in this mode; otherwise the pin is ignored.

Once in Pin Sleep Mode, $\overline{\text{CTS}}$ is de-asserted (high), indicating that data should not be sent to the module. The PWR pin is also de-asserted (low) when the module is in Pin Sleep Mode.

Note: The module will complete a transmission or reception before activating Pin Sleep.

Cyclic Sleep (SM = 3-8)

Cyclic Sleep is the Sleep Mode in which the XBee module enters into a low-power state and awakens periodically to determine if any transmissions are being sent.

When Cyclic Sleep settings are enabled, the XBee module goes into Sleep Mode after a user-defined period of inactivity (no transmission or reception on the RF channel). The user-defined period is determined by ST (Time before Sleep) Command.

While the module is in Cyclic Sleep Mode, $\overline{\text{CTS}}$ is de-asserted (high) to indicate that data should not be sent to the module during this time. When the module awakens to listen for data, $\overline{\text{CTS}}$ is asserted and any data received on the DI Pin is transmitted. The PWR pin is also de-asserted (low) when the module is in Cyclic Sleep Mode.

The module remains in Sleep Mode for a user-defined period of time ranging from 0.5 seconds to 16 seconds (SM Parameters 3 through 8). After this interval of time, the module returns to Idle Mode and listens for a valid data packet for 100 ms. If the module does not detect valid data (on any frequency), the module returns to Sleep Mode. If valid data is detected, the module transitions into Receive Mode and receives incoming RF packets. The module then returns to Sleep Mode after a Period of inactivity that is determined by ST "Time before Sleep" Command.

The module can also be configured to wake from cyclic sleep when SLEEP (pin 9) is de-asserted (low). To configure a module to operate in this manner, PW (Pin Wake-up) Command must be issued. Once SLEEP is de-asserted, the module is forced into Idle Mode and can begin transmitting or receiving data. It remains active until no data is detected for the period of time specified by the ST Command, at which point it resumes its low-power cyclic state.

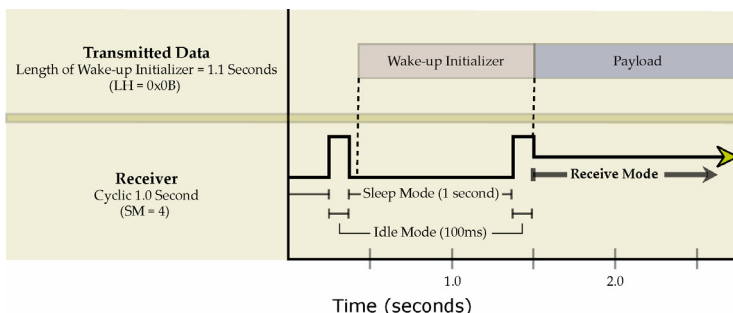
Note: The cyclic interval time defined by SM (Sleep Mode) Command must be shorter than the interval time defined by LH (Wake-up Initializer Timer).

For example: If SM=4 (Cyclic 1.0 second sleep), the LH Parameter should equal 0x0B ("1.1" seconds). With these parameters set, there is no risk of the receiving module being asleep for the duration of wake-up initializer transmission. "Cyclic Scanning" explains in further detail the relationship between "Cyclic Sleep" and "Wake-up Initializer Timer"

Cyclic Scanning. Each RF transmission consists of an RF Initializer and payload. The wake-up initializer contains initialization information and all receiving modules must wake during the wake-up initializer portion of data transmission in order to be synchronized with the transmitting module and receive the data.

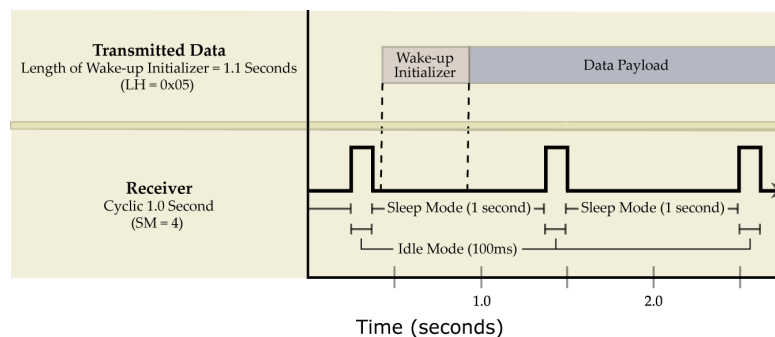
Correct Configuration (LH > SM)

Length of the wake-up initializer exceeds the time interval of Cyclic Sleep. The receiver is guaranteed to detect the wake-up initializer and receive the accompanying payload data.



Incorrect Configuration (LH < SM)

Length of wake-up initializer is shorter than the time interval of Cyclic Sleep. This configuration is vulnerable to the receiver waking and missing the wake-up initializer (and therefore also the accompanying payload data).



Command Mode

To modify or read module parameters, the module must first enter into Command Mode, the state in which received characters on the UART are interpreted as commands. Two command types are available for programming the module:

- AT Commands
- Binary Commands

For modified parameter values to persist in the module registry, changes must be saved to non-volatile memory using WR (Write) Command. Otherwise, parameters are restored to previously saved values after the module is powered off and then on again.

AT Commands

To Enter AT Command Mode:

- Send the 3-character command sequence "+++" and observe guard times before and after the command characters. [refer to 'Default AT Command Mode Sequence' below.] The 'Terminal' tab (or other serial communications software) of the X-CTU Software can be used to enter the sequence.

[OR]

- Assert (low) the CONFIG pin and either turn the power going to the module off and back on. (If using a Digi XBIB-R Interface Board, the same result can be achieved by holding the Data-In line low (also known as a break) while rebooting the module by pressing the reset button on the module assembly [module assembly = module mounted to an interface board]).

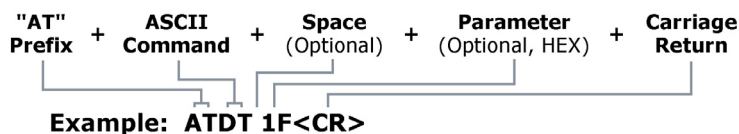
Default AT Command Mode Sequence (for transition to Command Mode):

- No characters sent for one second [refer to the BT (Guard Time Before) Command]
- Input three plus characters ("+++") within one second [refer to the CC (Command Sequence Character) Command.]
- No characters sent for one second [refer to the AT (Guard Time After) Command.]

To Send AT Commands:

Send AT commands and parameters using the syntax shown below.

Figure 1-1. Syntax for sending AT Commands



To read a parameter value stored in the module register, leave the parameter field blank.

The preceding example would change the module's Destination Address to "0x1F". To store the new value to non-volatile (long term) memory, the Write (ATWR) command must subsequently be sent before powering off the module.

System Response. When a command is sent to the module, the module will parse and execute the command. Upon successful execution of a command, the module returns an "OK" message. If execution of a command results in an error, the module returns an "ERROR" message.

To Exit AT Command Mode:

- If no valid AT Commands are received within the time specified by CT (Command Mode Timeout) Command, the module automatically returns to Idle Mode.
[OR]
- Send ATCN (Exit Command Mode) Command.

For an example of programming the RF module using AT Commands and descriptions of each configurable parameter, refer to the "RF Module Configuration" chapter.

Binary Commands

Sending and receiving parameter values using binary commands is the fastest way to change operating parameters of the module. Binary commands are used most often to sample signal strength (RS parameter) and/or error counts; or to change module addresses and channels for polling systems when a quick response is necessary. Since the sending and receiving of parameter values takes place through the same data path as 'live' data (received RF payload), follow the CTS pin as outlined in Figure 2-012 to distinguish between the two types of data (commands vs 'live' data).

Common questions regarding the use of binary commands:

- What are the implications of asserting CMD while live data is being sent or received?
- After sending serial data, is there a minimum time delay before CMD can be asserted?
- Is a time delay required after CMD is de-asserted before payload data can be sent?
- How to discern between live data and data received in response to a command?

CMD (pin 16) must be asserted in order to send binary commands to the module. The CMD pin can be asserted to recognize binary commands anytime during the transmission or reception of data. The status of the CMD signal is only checked at the end of the stop bit as the byte is shifted into the serial port. The application does not allow control over when data is received, except by waiting for dead time between bursts of communication.

If the command is sent in the middle of a stream of payload data to be transmitted, the command will essentially be executed in the order it is received. If the radio is continuously receiving data, the radio will wait for a break in the received data before executing the command. The $\overline{\text{CTS}}$ signal will frame the response coming from the binary command request [Figure].

A minimum time delay of 100 μs (after the stop bit of the command byte has been sent) must be observed before pin 5 can be de-asserted. The command executes after all parameters associated with the command have been sent. If all parameters are not received within 0.5 seconds, the module aborts the command and returns to Idle Mode.

Note: Binary commands that return only one parameter byte must also be written with two parameter bytes, 0-padded, LSB first. Refer to “Programming Examples” section [p18] for a binary programming example.

Commands can be queried for their current value by sending the command logically ORed (bit-wise) with the value 0x80 (hexadecimal) with CMD asserted. When the binary value is sent (with no parameters), the current value of the command parameter is sent back through the DO pin.

Binary Command Write then Read

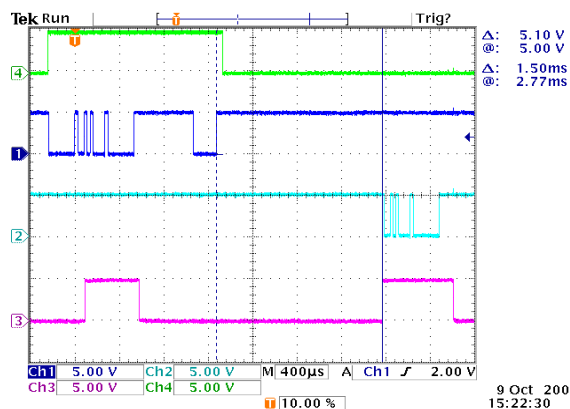
Signal #4 is CMD (pin 16)

Signal #1 is the DIN (pin 3) signal to the radio

Signal #2 is the DOUT (pin 2) signal from the radio

Signal #3 is $\overline{\text{CTS}}$ (pin 12)

In this graph, a value was written to a register and then read out to verify it. While not in the middle of other received data, note that the $\overline{\text{CTS}}$ signal outlines the data response out of the module.



IMPORTANT: For the XBee module to recognize a binary command, the RT (DI2 Configuration) parameter must be set to one. If binary programming is not enabled $\text{RT} = 0$ or 2 , the module will not recognize that the CMD pin is asserted and therefore will not recognize the data as binary commands.

RF Module Configuration

XBee Programming Examples

For information about entering and exiting AT and Binary Command Modes, refer to the Command Mode section.

AT Commands

To Send AT Commands (Using the 'Terminal' tab of the X-CTU Software)

Example: Utilize the 'Terminal' tab of the X-CTU Software to change the module's DT (Destination Address) parameter and save the new address to non-volatile memory. This example requires the installation of Digi's X-CTU Software and a serial connection to a PC.

Select the 'Terminal' tab of the X-CTU Software and enter the following command lines:

Method 1 (One line per command)

| Send AT Command | System Response |
|------------------|---|
| +++ | OK <CR> (Enter into Command Mode) |
| ATDT <Enter> | {current value} <CR> (Read Destination Address) |
| ATDT1A0D <Enter> | OK <CR> (Modify Destination Address) |
| ATWR <Enter> | OK <CR> (Write to non-volatile memory) |
| ATCN <Enter> | OK <CR> (Exit Command Mode) |

Method 2 (Multiple commands on one line)

| Send AT Command | System Response |
|------------------------|---|
| +++ | OK <CR> (Enter into Command Mode) |
| ATDT <Enter> | {current value} <CR> (Read Destination Address) |
| ATDT1A0D,WR,CN <Enter> | OK <CR> (Execute commands) |

Note: Do not send commands to the module during flash programming (when parameters are being written to the module registry).

Wait for the "OK" system response that follows the ATWR command before entering the next command or use flow control.

Note: When using X-CTU Software to program a module, PC com port settings must match the baud (interface data rate), parity & stop bits parameter settings of the module. Use the 'Com Port Setup' section of the "PC Settings" tab to configure PC com port settings to match those of the module.

Binary Commands

To Send Binary Commands

Example: Use binary commands to change the XBee module's destination address to 0x1A0D and save the new address to non-volatile memory.

1. RT Command must be set to "1" in AT Command Mode to enable binary programming.
2. Assert CMD (Pin 16 is driven high). (Enter Binary Command Mode)
3. Send Bytes (parameter bytes must be 2 bytes long):

| | |
|---|---|
| 00 | (Send DT (Destination Address) Command) |
| 0D | (Least significant byte of parameter bytes) |
| 1A | (Most significant byte of parameter bytes) |
| 08 | (Send WR (Write) Command) |
| 4. De-assert CMD (Pin 16 is driven low) | (Exit Binary Command Mode) |

Note: $\overline{\text{CTS}}$ is de-asserted high when commands are being executed. Hardware flow control must be disabled as $\overline{\text{CTS}}$ will hold off parameter bytes.

Command Reference Table

Table 1-03. AT Commands (The RF Module expects numerical values in hexadecimal. “d” denotes decimal equivalent.)

| AT Command | Binary Command | AT Command Name | Range | Command Category | # Bytes Returned | Factory Default |
|------------|----------------|---------------------------------|--|-----------------------|------------------|-----------------|
| *AM | 0x3A (58d) | Auto-set MY | - | Networking & Security | - | - |
| AT | 0x05 (5d) | Guard Time After | 0x02 – 0xFFFF [x 100 msec] | Command Mode Options | 2 | 0x0A (10d) |
| BD | 0x15 (21d) | Interface Data Rate | Standard baud rates: 0 – 6 Non-standard baud rates: 0x7D – 0xFFFF | Serial Interfacing | 2 | 0x03 9600bps |
| BT | 0x04 (4d) | Guard Time Before | 2 – 0xFFFF [x 100 msec] | Command Mode Options | 2 | 0x0A (10d) |
| CC | 0x13 (19d) | Command Sequence Character | 0x20 – 0x7F | Command Mode Options | 1 | 0x2B (“+”) |
| CD | 0x28 (40d) | DO3 Configuration | 0 - 4 | Serial Interfacing | 1 | 0 |
| CN | 0x09 (9d) | Exit AT Command Mode | - | Command Mode Options | - | - |
| CS | 0x1F (31d) | DO2 Configuration | 0 – 4 | Serial Interfacing | 1 | 0 |
| CT | 0x06 (6d) | Command Mode Timeout | 0x02 – 0xFFFF [x 100 msec] | Command Mode Options | 2 | 0xC8 (200d) |
| DT | 0x00 (0d) | Destination Address | 0 – 0xFFFF | Networking | 2 | 0 |
| E0 | 0x0A (10d) | Echo Off | - | Command Mode Options | - | - |
| E1 | 0x0B (11d) | Echo On | - | Command Mode Options | - | - |
| ER | 0x0F (15d) | Receive Error Count | 0 – 0xFFFF | Diagnostics | 2 | 0 |
| FH | 0x0D (13d) | Force Wake-up Initializer | - | Sleep (Low Power) | - | - |
| FL | 0x07 (7d) | Software Flow Control | 0 – 1 | Serial Interfacing | 1 | 0 |
| FR | N/A | Forces the module to Reset | | (Special) | | |
| FT | 0x24 (36d) | Flow Control Threshold | 0 – (DI buffer – 0x11) [bytes] | Serial Interfacing | 2 | varies |
| GD | 0x10 (16d) | Receive Good Count | 0 – 0xFFFF | Diagnostics | 2 | 0 |
| HP | 0x11 (17d) | Hopping Channel | 0 – 6 | Networking | 1 | 0 |
| HT | 0x03 (3d) | Time before Wake-up Initializer | 0 – 0xFFFF [x 100 msec] | Sleep (Low Power) | 2 | 0xFFFF |
| ID | 0x27 (39d) | Module VID | User set table: 0x10 - 0xFFFF Read-only: 0x8000 – 0xFFFF | Networking | 2 | - |
| LH | 0x0C (12d) | Wake-up Initializer Timer | 0 – 0xFF [x 100 msec] | Sleep (Low Power) | 1 | 1 |
| MD | 0x32 (50d) | RF Mode | 0 – 4 | Networking & Security | 1 | 0 |
| MK | 0x12 (18d) | Address Mask | 0 – 0xFFFF | Networking | 2 | 0xFFFF |
| *MY | 0x2A (42d) | Source Address | 0 – 0xFFFF | Networking & Security | 2 | 0xFFFF |
| NB | 0x23 (35d) | Parity | 0 – 5 | Serial Interfacing | 1 | 0 |
| PC | 0x1E (30d) | Power-up Mode | 0 – 1 | Command Mode Options | 1 | 0 |
| *PK | 0x29 (41d) | RF Packet Size | 0 - 0x100 [bytes] | Serial Interfacing | 2 | 0x40 (64d) |
| *PL | 0x3c (60d) | RF Power Level | 0-4 | (Special) | 1 | 4 |
| PW | 0x1D (29d) | Pin Wake-up | 0 – 1 | Sleep (Low Power) | 1 | 0 |
| *RB | 0x20 (32d) | Packetization Threshold | 0 - 0x100 [bytes] | Serial Interfacing | 2 | 0x01 |
| RE | 0x0E (14d) | Restore Defaults | - | (Special) | - | - |
| RN | 0x19 (25d) | Delay Slots | 0 – 0xFF [slots] | Networking | 1 | 0 |
| RO | 0x21 (33d) | Packetization Timeout | 0 – 0xFFFF [x 200 µsec] | Serial Interfacing | 2 | 0 |
| RP | 0x22 (34d) | RSSI PWM Timer | 0 - 0x7F [x 100 msec] | Diagnostics | 1 | 0 |
| RR | 0x18 (24d) | Retries | 0 – 0xFF | Networking | 1 | 0 |
| RS | 0x1C (28d) | RSSI | 0x06 – 0x36 [read-only] | Diagnostics | 1 | - |
| RT | 0x16 (22d) | DI2 Configuration | 0 - 2 | Serial Interfacing | 1 | 0 |
| *RZ | 0x2C (44d) | DI Buffer Size | [read-only] | Diagnostics | - | - |
| SB | 0x36 (54d) | Stop Bits | 0 - 1 | Serial Interfacing | 1 | 0 |
| SH | 0x25 (37d) | Serial Number High | 0 – 0xFFFF [read-only] | Diagnostics | 2 | - |
| SL | 0x26 (38d) | Serial Number Low | 0 – 0xFFFF [read-only] | Diagnostics | 2 | - |
| SM | 0x01 (1d) | Sleep Mode | 0, 1, 3 - 8 | Sleep (Low Power) | 1 | 0 |
| ST | 0x02 (2d) | Time before Sleep | 0x10 – 0xFFFF [x 100 msec] | Sleep (Low Power) | 2 | 0x64 (100d) |
| SY | 0x17 (23d) | Time before Initialization | 0 – 0xFF [x 100 msec] | Networking | 1 | 0 (disabled) |
| TR | 0x1B (27d) | Transmit Error Count | 0 – 0xFFFF | Diagnostics | 2 | 0 |

| | | | | | | |
|----|------------|------------------|---------------------------|-------------|---|--------|
| TT | 0x1A (26d) | Streaming Limit | 0 – 0xFFFF [0 = disabled] | Networking | 2 | 0xFFFF |
| VR | 0x14 (20d) | Firmware Version | 0 - 0xFFFF [read-only] | Diagnostics | 2 | - |
| WR | 0x08 (8d) | Write | - | (Special) | - | - |

NOTE: AT Commands issued without a parameter value are interpreted as queries and will return the currently stored parameter.

*Commands only supported on S3B hardware.

Command Descriptions

Commands in this section are listed alphabetically. Command categories are designated between the "< >" symbols that follow each command title. Modules expect numerical values in hexadecimal and those values are designated by a "0x" prefix.

Modules operating within the same network should contain the same firmware platform to ensure the same AT Command parameters are supported.

AM (Auto-set MY) Command

| Command Summary | Description |
|--|--|
| AT Command: ATAM | <Networking & Security> AM Command is used to automatically set the MY (Source Address) parameter from the factory-set module serial number. The address is formed with bits 29, 28 and 13-0 of the serial number (in that order). |
| Binary Command: 0x3A (58 decimal) | |
| This command is only supported on S3B modules. | |
| | |

AT (Guard Time After) Command

| Command Summary | Description |
|---|--|
| AT Command: ATAT | <Command Mode Options> AT Command is used to set the time-of-silence that follows the command sequence character (CC Command). By default, AT Command Mode will activate after one second of silence. Refer to the AT Commands section to view the default AT Command Mode Sequence. |
| Binary Command: 0x05 (5 decimal) | |
| Parameter Range: 0x02 – 0xFFFF [x 100 milliseconds] | |
| Number of bytes returned: 2 | |
| Default Parameter Value: 0x0A (10 decimal) | |
| Related Commands: BT (Guard Time Before), CC (Command Sequence Character) | |

BD (Interface Data Rate) Command**Command Summary**

AT Command: ATBD

Binary Command: 0x15 (21 decimal)

Parameter Range (Standard baud rates): 0 – 6

(Non-standard baud rates): 0x7D – 0xFFFF (125d – 65535d)

| Parameter Value | BAUD (bps) Configuration |
|-----------------|--------------------------|
| 0 | 1200 |
| 1 | 2400 |
| 2 | 4800 |
| 3 | 9600 |
| 4 | 19200 |
| 5 | 38400 |
| 6 | 57600 |

Number of bytes returned: 2

Default Parameter Value: Set to equal module's factory-set RF data rate.

Description

<Serial Interfacing> BD Command allows the user to adjust the UART interface data rate and thus modify the rate at which serial data is sent to the module. The new baud rate does not take effect until the CN (Exit AT Command Mode) Command is issued. The RF data rate is not affected by the BD Command. Although most applications will only require one of the seven standard baud rates, non-standard baud rates are also supported.

Note: If the serial data rate is set to exceed the fixed RF data rate of the module, flow control may need to be implemented as described in the Pin Signals and Flow Control sections of this manual.

Non-standard Interface Data Rates: When parameter values outside the range of standard baud rates are sent, the closest interface data rate represented by the number is stored in the BD register. For example, a rate of 19200 bps can be set by sending the following command line "ATBD4B00". NOTE: When using X-CTU Software, non-standard interface data rates can only be set and read using the X-CTU 'Terminal' tab. Non-standard rates are not accessible through the 'Modem Configuration' tab.

When the BD command is sent with a non-standard interface data rate, the UART will adjust to accommodate the requested interface rate. In most cases, the clock resolution will cause the stored BD parameter to vary from the parameter that was sent (refer to the table below). Reading the BD command (send "ATBD" command without an associated parameter value) will return the value that was actually stored to the BD register.

Parameter Sent vs. Parameter Stored

| BD Parameter Sent (HEX) | Interface Data Rate (bps) | S3 BD Parameter Stored (HEX) | S3B BD Parameter Stored (HEX) |
|-------------------------|---------------------------|------------------------------|-------------------------------|
| 0 | 1200 | 0 | 0 |
| 4 | 19,200 | 4 | 4 |
| 6 | 57600 | 6 | 5 |
| 12C | 300 | 12B | 12B |
| E100 | 57600 | E883 | E10D |

BT (Guard Time Before) Command**Command Summary**

AT Command: ATBT

Binary Command: 0x04 (4 decimal)

Parameter Range: 2 – 0xFFFF
[x 100 milliseconds]

Default Parameter Value: 0x0A (10 decimal)

Number of bytes returned: 2

Related Commands: AT (Guard Time After), CC (Command Sequence Character)

Description

<Command Mode Options> BT Command is used to set the DI pin silence time that must precede the command sequence character (CC Command) of the AT Command Mode Sequence. Refer to the AT Commands section to view the default AT Command Mode Sequence.

CC (Command Sequence Character) Command

| Command Summary | Description |
|---|---|
| AT Command: ATCC | <Command Mode Options> CC Command is used to set the ASCII character to be used between Guard Times of the AT Command Mode Sequence (BT+ CC + AT). The AT Command Mode Sequence activates AT Command Mode (from Idle Mode). Refer to the AT Commands section [p. 18] to view the default AT Command Mode Sequence. |
| Binary Command: 0x13 (19 decimal) | |
| Parameter Range: 0x20 – 0x7F | |
| Default Parameter Value: 0x2B (ASCII “+” sign) | |
| Number of bytes returned: 1 | |
| Related Commands: AT (Guard Time After), BT (Guard Time Before) | |

CD (DO3 Configuration) Command

| | | Description |
|-----------------------------------|---|---|
| AT Command: ATCD | | <Command Mode Options> CD Command is used to define the behavior of the DO3/RX LED line. |
| Binary Command: 0x28 (40 decimal) | | |
| Parameter Range: 0 – 3 | | |
| Parameter Value | Configuration | |
| 0 | RX LED | |
| 1 | Default high | |
| 2 | Default low | |
| 3 | (reserved) | |
| 4 | Assert only when packet addressed to module is sent | |
| Default Parameter Value: 0 | | |
| Number of bytes returned: 1 | | |

CN (Exit AT Command Mode) Command

| Command Summary | Description |
|----------------------------------|--|
| AT Command: ATCN | <Command Mode Options> CN Command is used to explicitly exit AT Command Mode. |
| Binary Command: 0x09 (9 decimal) | |

CS (DO2 Configuration) Command

| Command Summary | Description |
|--|---|
| AT Command: ATCS | <Serial Interfacing> CS Command is used to select the behavior of the DO2 pin signal. This output can provide RS-232 flow control, control the TX enable signal (for RS-485 or RS-422 operations), or set the default level for the I/O line passing function. By default, DO2 provides RS-232 $\overline{\text{CTS}}$ (Clear-to-Send) flow control. |
| Binary Command: 0x1F (31 decimal) | |
| Parameter Range: 0 – 4 | |
| Parameter Value | Configuration |
| 0 | RS-232 $\overline{\text{CTS}}$ flow control |
| 1 | RS-485 TX enable low |
| 2 | high |
| 3 | RS-485 TX enable high |
| 4 | low |
| Default Parameter Value: 0 | |
| Number of bytes returned: 1 | |
| Minimum Firmware Version Required: 4.27D | |

CT (Command Mode Time out) Command

| Command Summary | Description |
|---|---|
| AT Command: ATCT | <Command Mode Options> CT Command sets the amount of time before AT Command Mode terminates automatically. After a CT time of inactivity, the module exits AT Command Mode and returns to Idle Mode. AT Command Mode can also be exited manually using CN (Exit AT Command Mode) Command. |
| Binary Command: 0x06 (6 decimal) | |
| Parameter Range: 0x02 – 0xFFFF [x 100 milliseconds] | |
| Default Parameter Value: 0xC8 (200 decimal, 20 seconds) | |
| Number of bytes returned: 2 | |

DT (Destination Address) Command

| Command Summary | Description |
|--|---|
| AT Command: ATDT | <p><Networking> DT Command is used to set the networking address of a Module. Modules use three network layers – Vendor Identification Number (ATID), Channels (ATHP), and Destination Addresses (ATDT). DT Command assigns an address to a module that enables it to communicate only with other modules having the same addresses. All modules that share the same Destination Address can communicate freely with each other. Modules in the same network with a different Destination Address (than that of the transmitter) will listen to all transmissions to stay synchronized, but will not send any of the data out their serial ports.</p> |
| Binary Command: 0x00 | |
| Parameter Range: 0 – 0xFFFF | |
| Default Parameter Value: 0 | |
| Number of bytes returned: 2 | |
| Related Commands: HP (Hopping Channel), ID (Module VID), MK (Address Mask) | |

E0 (Echo Off) Command

| Command Summary | Description |
|-----------------------------------|--|
| AT Command: ATE0 | <p><Command Mode Options> E0 Command turns off character echo in AT Command Mode. By default, echo is off.</p> |
| Binary Command: 0x0A (10 decimal) | |

E1 (Echo On) Command

| Command Summary | Description |
|-----------------------------------|---|
| AT Command: ATE1 | <p><Command Mode Options> E1 Command turns on the echo in AT Command Mode. Each typed character will be echoed back to the terminal when ATE1 is active. E0 is the default.</p> |
| Binary Command: 0x0B (11 decimal) | |

ER (Receive Error Count) Command

| Command Summary | Description |
|---|---|
| AT Command: ATER | <Diagnostics> Set/Read the receive-error. The error-count records the number of packets partially received then aborted on a reception error. This value returns to 0 after a reset and is not non-volatile (Value does not persist in the module's memory after a power-up sequence). Once the "Receive Error Count" reaches its maximum value (up to 0xFFFF), it remains at its maximum count value until the maximum count value is explicitly changed or the module is reset. |
| Binary Command: 0x0F (15 decimal) | |
| Parameter Range: 0 – 0xFFFF | |
| Default Parameter Value: 0 | |
| Number of bytes returned: 2 | |
| Related Commands: GD (Receive Good Count) | |

FH (Force Wake-up Initializer) Command

| Command Summary | Description |
|-----------------------------------|--|
| AT Command: ATFH | <Sleep (Low Power)> FH Command is used to force a Wake-up Initializer to be sent on the next transmit. WR (Write) Command does not need to be issued with FH Command. Use only with cyclic sleep modes active on remote modules. |
| Binary Command: 0x0D (13 decimal) | |

FL (Software Flow Control) Command

| Command Summary | Description | |
|----------------------------------|---|-------------------------------|
| AT Command: ATFL | <Serial Interfacing> FL Command is used to configure software flow control. Hardware flow control is implemented with the Module as the DO2 pin (), which regulates when serial data can be transferred to the module. FL Command can be used to allow software flow control to also be enabled. XON character used is 0x11 (17 decimal). XOFF character used is 0x13 (19 decimal). | |
| Binary Command: 0x07 (7 decimal) | | |
| Parameter Range: 0 – 1 | | |
| Parameter Value | | Configuration |
| 0 | | Disable software flow control |
| 1 | Enable software flow control | |
| Default Parameter Value: 0 | | |
| Number of bytes returned: 1 | | |

FR (Force Reset) Command

| Command Summary | Description |
|-------------------------------|--|
| AT Command: ATFR | <Special> FR command is used in order to reset the module through the UART. The characters "OK"<CR> will be returned and the module will reset 100ms |
| Binary Command: Not available | |

FT (Flow Control Threshold) Command

| Command Summary | Description |
|---|---|
| AT Command: ATFT | <Serial Interfacing> Flow Control Threshold – Set or read flow control threshold. De-assert CTS and/or send XOFF when FT bytes are in the UART receive buffer. Re-assert CTS when less than FT – 16 bytes are in the UART receive buffer. |
| Binary Command: 0x24 (36 decimal) | |
| Parameter Range: 0 – (DI buffer size minus 0x11 bytes) | |
| Default Parameter Value: DI Buffer size minus 0x11 (17 decimal) | |
| Number of bytes returned: 2 | |
| Minimum Firmware Version Required: 4.27B | |

GD (Receive Good Count) Command

| Command Summary | Description |
|--|--|
| AT Command: ATGD | <Diagnostics> Set/Read the count of good received RF packets. Parameter value is reset to 0 after every reset and is not non-volatile (Value does not persist in the module's memory after a power-up sequence). Once the "Receive Good Count" reaches its maximum value (up to 0xFFFF), it remains at its maximum count value until the maximum count value is manually changed or the module is reset. |
| Binary Command: 0x10 (16 decimal) | |
| Parameter Range: 0 – 0xFFFF | |
| Default Parameter Value: 0 | |
| Number of bytes returned: 2 | |
| Related Commands: ER (Receive Error Count) | |

HP (Hopping Channel) Command

| Command Summary | Description |
|--|---|
| AT Command: ATHP | <Networking> HP Command is used to set the module's hopping channel number. A channel is one of three layers of addressing available to the module. In order for modules to communicate with each other, the modules must have the same channel number since each network uses a different hopping sequence. Different channels can be used to prevent modules in one network from listening to transmissions of another. |
| Binary Command: 0x11 (17 decimal) | |
| Parameter Range: 0 – 6 | |
| Default Parameter Value: 0 | |
| Number of bytes returned: 1 | |
| Related Commands: DT (Destination Address), ID (Module VID), MK (Address Mask) | |

HT (Time before Wake-up Initializer) Command

| Command Summary | Description |
|---|--|
| AT Command: ATHT | <p><Sleep (Low Power)> If any modules within range are running in a “Cyclic Sleep” setting, a wake-up initializer must be used by the transmitting module for sleeping modules to remain awake [refer to the LH (“Wake-up InitializerTimer”) Command]. When a receiving module in Cyclic Sleep wakes, it must detect the wake-up initializer in order to remain awake and receive data. The value of HT Parameter tells the transmitter, “After a period of inactivity (no transmitting or receiving) lasting HT amount of time, send a long wake-up initializer”. HT Parameter should be set to match the inactivity time out [specified by ST (Time before Sleep) Command] used by the receiver(s). From the receiving module perspective, after HT time elapses and the inactivity time out [ST Command] is met, the receiver goes into cyclic sleep. In cyclic sleep, the receiver wakes once per sleep interval to check for a wakeup initializer. When a wake-up initializer is detected, the module will stay awake to receive data. The wake-up initializer must be longer than the cyclic sleep interval to ensure that sleeping modules detect incoming data. When HT time elapses, the transmitter then knows that it needs to send a long Wake-up Initializer for all receivers to be able to remain awake and receive the next transmission. Matching HT to the time specified by ST on the receiving module guarantees that all receivers will detect the next transmission.</p> |
| Binary Command: 0x03 (3 decimal) | |
| Parameter Range: 0 – 0xFFFF [x 100 milliseconds] | |
| Default Parameter Value: 0xFFFF (means that long wake-up initializer will not be sent) | |
| Number of bytes returned: 2 | |
| Related Commands: LH (Wake-up Initializer Timer), SM (Sleep Mode), ST (Time before Sleep) | |

ID (Modem VID) Command

| Command Summary | Description |
|--|---|
| AT Command: ATID | <p><Networking> Set/Read the “Vendor Identification Number”. Only modems with matching IDs can communicate with each other. Modules with non-matching VIDs will not receive unintended data transmission.</p> |
| Binary Command: 0x27 (39 decimal) | |
| Parameter Range (user-set table) 0x10 – 0x7FFFF (Factory-set and read-only) 0x8000 – 0xFFFF | |
| Number of bytes returned: 2 | |

LH (Wake-up Initializer Timer) Command

| Command Summary | Description |
|---|--|
| AT Command: ATLH | <p><Sleep (Low Power)> LH Command adjusts the duration of time for which the RF initializer is sent. When receiving modules are put into Cyclic Sleep Mode, they power-down after a period of inactivity [specified by ST (Time before Sleep) Command] and will periodically awaken and listen for transmitted data. In order for the receiving modules to remain awake, they must detect ~35ms of the wake-up initializer. LH Command must be used whenever a receiver is operating in Cyclic Sleep Mode. This lengthens the Wake-up Initializer to a specific amount of time (in tenths of a second). The Wake-up Initializer Time must be longer than the cyclic sleep time that is determined by SM (Sleep Mode) Command. If the wake-up initializer time were less than the Cyclic Sleep interval, the connection would be at risk of missing the wake-up initializer transmission. Refer to Figures 3.1 & 3.2 of the SM Command description to view diagrams of correct and incorrect configurations. The images help visualize the importance that the value of LH be greater than the value of SM.</p> |
| Binary Command: 0x0C (12 decimal) | |
| Parameter Range: 0 – 0xFF [x 100 milliseconds] | |
| Default Parameter Value: 1 | |
| Number of bytes returned: 1 | |
| Related Commands: HT (Time before Wake-up Initializer), SM (Sleep Mode), ST (Time before Sleep) | |

MD (RF Mode) Command

| Command Summary | | Description |
|-----------------------------------|--------------------------------------|---|
| AT Command: ATMD | | <Networking & Security> The MD command is used to select/read the RF Mode (Peer-to-peer or Repeater Modes) of the module. |
| Binary Command: 0x32 (50 decimal) | | |
| Parameter Range: 0, 3, 4 | | |
| Parameter | Configuration | Repeater Mode enables longer range via an intermediary module. When MD=3, the module will act as a “store and forward” repeater. Any packets not addressed to this node will be repeated. |
| 0 | Peer-to-Peer (transparent operation) | |
| 3 | Repeater & End Node | |
| 4 | End Node | |
| Default Parameter Value: 0 | | A Repeater End Node (MD=4) handles repeated messages, but will not forward the data over-the-air. Refer to the Repeater Mode section [p. 40] for more information. |
| Number of bytes returned: 1 | | |

MK (Address Mask) Command

| Command Summary | Description |
|--|--|
| AT Command: ATMK | <p><Networking> MK Command is used to set/read the Address Mask.</p> <p>All data packets contain the Destination Address of the transmitting module.</p> <p>When an RF data packet is received, the transmitter's Destination Address is logically "ANDed" (bitwise) with the Address Mask of the receiver. The resulting value must match the Destination Address or the Address Mask of the receiver for the packet to be received and sent out the module's DO serial port. If the "ANDed" value does not match either the Destination Address or the Address Mask of the receiver, the packet is discarded. (All "0" values are treated as "irrelevant" values and are ignored.)</p> |
| Binary Command: 0x12 (18 decimal) | |
| Parameter Range: 0 – 0xFFFF | |
| Default Parameter Value: 0xFFFF (Destination address (DT parameter) of the transmitting module must exactly match the destination address of the receiving module.) | |
| Number of bytes returned: 2 | |
| Related Commands: DT (Destination Address), HP (Hopping Channel), ID (Module VID) | |

MY (Source Address) Command

| Command Summary | Description |
|---|---|
| AT Command: ATMY | <Networking & Security> Set/Read the source address of the module. Refer to the Addressing section [p. 38] of the RF Communication Modes chapter for more information. |
| Binary Command: 0x2A (42 decimal) | |
| Parameter Range: 0 – 0xFFFF | |
| Default Parameter Value: 0xFFFF (Disabled – the DT (Destination Address) parameter serves as both source and destination address.) | |
| Number of bytes returned: 2 | |
| Related Commands: DT (Destination Address), HP (Hopping Channel), ID (Modem VID), MK (Address Mask), AM (Auto-set MY) | |
| This command is only supported on S3B modules. | |

NB (Parity) Command

| Command Summary | Description | | | | | | | | | | | | | | |
|--|--|-----------------|---------------|---|---|---|------------|---|-----------|---|------------|---|-------------|---|---------------------------|
| AT Command: ATNB | <Serial Interfacing> Select/Read parity settings for UART communications. | | | | | | | | | | | | | | |
| Binary Command: 0x23 (35 decimal) | | | | | | | | | | | | | | | |
| Parameter Range:0 – 4 (S3 Hardware) 0–5 (S3B Hardware) | | | | | | | | | | | | | | | |
| <table><tr><th>Parameter Value</th><th>Configuration</th></tr><tr><td>0</td><td>8-bit (no parity or 7-bit (any parity))</td></tr><tr><td>1</td><td>8-bit even</td></tr><tr><td>2</td><td>8-bit odd</td></tr><tr><td>3</td><td>8-bit mark</td></tr><tr><td>4</td><td>8-bit space</td></tr><tr><td>5</td><td>9-bit data (S3B Hardware)</td></tr></table> | | Parameter Value | Configuration | 0 | 8-bit (no parity or 7-bit (any parity)) | 1 | 8-bit even | 2 | 8-bit odd | 3 | 8-bit mark | 4 | 8-bit space | 5 | 9-bit data (S3B Hardware) |
| Parameter Value | | Configuration | | | | | | | | | | | | | |
| 0 | 8-bit (no parity or 7-bit (any parity)) | | | | | | | | | | | | | | |
| 1 | 8-bit even | | | | | | | | | | | | | | |
| 2 | 8-bit odd | | | | | | | | | | | | | | |
| 3 | 8-bit mark | | | | | | | | | | | | | | |
| 4 | 8-bit space | | | | | | | | | | | | | | |
| 5 | 9-bit data (S3B Hardware) | | | | | | | | | | | | | | |
| Default Parameter Value: 0 | | | | | | | | | | | | | | | |
| Number of bytes returned: 1 | | | | | | | | | | | | | | | |

PC (Power-up to AT Mode) Command**Command Summary**

AT Command: ATPC

Binary Command: 0x1E (30 decimal)

Parameter Range: 0 - 1

| Parameter Value | Configuration |
|-----------------|-----------------------------|
| 0 | Power-up to Idle Mode |
| 1 | Power-up to AT Command Mode |

Default Parameter Value: 0

Number of bytes returned: 1

Description

<Command Mode

Options> PC

Command allows the module to power-up directly into AT

Command Mode from reset or power-on. If PC Command is enabled with SM

Parameter set to 1, DI3 (pin 9) can be used to enter the module into AT Command Mode. When the DI3 pin is de-asserted (low), the module will wake-up in AT Command Mode. This behavior allows module DTR emulation.

PK (RF Packet Size) Command**Command Summary**

AT Command: ATPK

Binary Command: 0x29 (41 decimal)

Parameter Range: 0 - 0x100 [Bytes]

Default Parameter Value: 0x40 (64 decimal)

Number of bytes returned: 2

Related Commands: RB (Packetization Threshold), RO (Packetization Time out)

This command is only supported on S3B modules.

Description

<Serial Interfacing> Set/Read the maximum size of the RF packets sent out a transmitting module. The maximum packet size can be used along with the RB and RO parameters to implicitly set the channel dwell time. Changes to this parameter may have a secondary effect on the RB (Packet Control Characters) parameter. RB must always be less than or equal to PK. If PK is changed to a value less than the current value of RB, RB is automatically lowered to be equal to PK.

PL (Module Power Level) Command**Command Summary**

AT Command: ATPL

Binary Command: 0x3C (60 decimal)

Parameter Range: 0 - 4

| Parameter Value | Configuration |
|-----------------|---------------------|
| 0 | +7.0 dBm, (5 mW) |
| 1 | +15.0dBm, (32 mW) |
| 2 | +18.0dBm, (63 mW) |
| 3 | +21.0dBm, (125 mW) |
| 4 | +24.0 dBm, (250 mW) |

Default Parameter Value: 4

Number of bytes returned: 1

This command is only supported on S3B hardware

Description

<Special Commands> Set/Read the power level at which the RF module transmits conducted power. This command is only supported on S3B hardware. Power level 4 is calibrated and the other power levels are approximate.

PW (Pin Wake-up) Command**Command Summary**

AT Command: ATPW

Binary Command: 0x1D (29 decimal)

Parameter Range: 0 – 1

| Parameter Value | Configuration |
|-----------------|---------------|
| 0 | Disabled |
| 1 | Enabled |

Default Parameter Value: 0

Number of bytes returned: 1

Related Commands: SM (Sleep Mode), ST (Time before Sleep)

Description

<Sleep (Low Power)> Under normal operation, a module in Cyclic Sleep Mode cycles from an active state to a low-power state at regular intervals until data is ready to be received. If the PW Parameter is set to 1, SLEEP (pin 2) can be used to wake the module from Cyclic Sleep. If the SLEEP pin is de-asserted (low), the module will be fully operational and will not go into Cyclic Sleep. Once SLEEP is asserted, the module will remain active for the period of time specified by ST (Time before Sleep) Command, and will return to Cyclic Sleep Mode (if no data is ready to be transmitted). PW Command is only valid if Cyclic Sleep has been enabled.

RB (Packetization Threshold) Command**Command Summary**

AT Command: ATRB

Binary Command: 0x20 (32 decimal)

Parameter Range: 0 – 0x100 [Bytes]
(Maximum value equals the current value of PK Parameter (up to 0x100 HEX (800 decimal)))

Default Parameter Value: 1

Number of bytes returned: 2

Related Commands: PK (RF Packet Size), RO (Packetization Time out)

This command is only supported on S3B modules.

Description

<Serial Interfacing> RF transmission will commence when data is in the DI Buffer and either of the following criteria are met:

- RO times out on the UART receive lines (ignored if RO = 0)
- RB characters have been received by the UART (ignored if RB = 0)

If PK is lowered below the value of RB; RB is automatically lowered to match PK. Note: RB and RO criteria only apply to the first packet of a multi-packet transmission. If data remains in the DI Buffer after the first packet, transmissions will continue in streaming manner until there is no data left in the DI Buffer (UART receive buffer).

RE (Restore Defaults) Command**Command Summary**

AT Command: ATRE

Binary Command: 0x0E (14 decimal)

Description

<Diagnostics> RE Command restores all configurable parameters to factory default settings. However, RE Command will not write the default values to non-volatile (persistent) memory. Unless the WR (Write) Command is issued after the RE command, the default settings will not be saved in the event of module reset or power-down.

RN (Delay Slots) Command

| Command Summary | Description |
|--|---|
| AT Command: ATRN | <p><Networking> RN Command is only applicable if retries have been enabled [RR (Retries) Command], or if forced delays will be inserted into a transmission [refer to TT (Streaming Limit) Command]. RN Command is used to adjust the time delay that the transmitter inserts before attempting to resend a packet. If the transmitter fails to receive an acknowledgement after sending a packet, it will insert a random number of delay slots (ranging from 0 to (RN minus 1)) before attempting to resend the packet. Each delay slot lasts for a period of 38ms.</p> <p>If two modules attempted to transmit at the same time, the random time delay after packet failure would allow one of the two modules to transmit the packet successfully, while the other would wait until the channel opens up to begin transmission.</p> |
| Binary Command: 0x19 (25 decimal) | |
| Parameter Range: 0 – 0xFF [slots] | |
| Default Parameter Value: 0 (no delay slots inserted) | |
| Number of bytes returned: 1 | |

RO (Packetization Time out) Command

| Command Summary | Description |
|--|---|
| AT Command: ATRO | <p><Serial Interfacing> RO Command is used to specify/read the time of silence (no bytes received) after which transmission begins. After a serial byte is received and if no other byte is received before the RO time out, the transmission will start.</p> |
| Binary Command: 0x21 (33 decimal) | |
| Parameter Range: 0 – 0xFFFF [x 200 µs] | |
| Default Parameter Value: 0 | |
| Number of bytes returned: 2 | |

RP (RSSI PWM Timer) Command

| Command Summary | Description |
|--|--|
| AT Command: ATRP | <p><Diagnostics> RP Command is used to enable a PWM ("Pulse Width Modulation") output on the Config pin which is calibrated to show the level the received RF signal is above the sensitivity level of the module. The PWM pulses vary from zero to 95 percent. Zero percent means the received RF signal is at or below the published sensitivity level of the module. The following table shows levels above sensitivity and PWM values.</p> <p>The total period of the PWM output is 8.32 ms. There are 40 steps in the PWM output and therefore the minimum step size is 0.208 ms.</p> |
| Binary Command: 0x22 (34 decimal) | |
| Parameter Range: 0 – 0x7F [x 100 milliseconds] | |
| Default Parameter Value: 0 (disabled) | |
| Number of bytes returned: 1 | |

PWM Chart

| dBm above Sensitivity | PWM percentage (high period / total period) |
|------------------------------|--|
| 10 | 47.5 % |
| 20 | 62.5 % |
| 30 | 77.5 % |

A non-zero value defines the time that the PWM output will be active with the RSSI value of the last received RF packet. After the set time when no RF packets are received, the PWM output will be set low (0 percent PWM) until another RF packet is received. The PWM output will also be set low at power-up. A parameter value of 0xFF permanently enables the PWM output and it will always reflect the value of the last received RF packet.

PWM output shares the Config input pin. When the module is powered, the Config pin will be an input. During the power-up sequence, the Config pin will be read to determine whether the module is going into AT Command Mode. After this, if RP parameter is a non-zero value, the Config pin will be configured as an output and set low until the first RF packet is received. With a non-zero RP parameter, the Config pin will be an input for RP ms after power up.

RZ (DI Buffer Size) Command

| Command Summary | Description |
|--|--|
| AT Command: ATRZ | <Diagnostics> The RZ command is used to read the size of the DI buffer (UART RX (Receive)). Note: The DO buffer size can be determined by multiplying the DI buffer size by 1.5. |
| Binary Command: 0x2C (44 decimal) | |
| Parameter Range: Read-only | |
| Number of bytes returned: 1 | |
| This command is only supported on S3B modules. | |

RR (Retries) Command

| Command Summary | Description |
|---------------------------------------|--|
| AT Command: ATRR | Networking> RRR Command specifies the number of retries that can be sent for a given RF packet. Once RR Command is enabled (set to a non-zero value), RF packet acknowledgements and retries are enabled. After transmitting a packet, the transmitter will wait to receive an acknowledgement from a receiver. If the acknowledgement is not received in the period of time specified by the RN (Delay Slots) Command, the transmitter will transmit the original packet again. The packet will be transmitted repeatedly until an acknowledgement is received or until the packet has been sent RR times. Note: For retries to work correctly, all modules in the system must have retries enabled. |
| Binary Command: 0x18 (24 decimal) | |
| Parameter Range: 0 – 0xFF | |
| Default Parameter Value: 0 (disabled) | |
| Number of bytes returned: 1 | |

RS (RSSI) Command**Command Summary**

AT Command: ATRS
 Binary Command: 0x1C (28 decimal)
 Parameter Range: 0x06 – 0x36 [read-only]
 Number of bytes returned: 1

Description

<Diagnostics> RS Command returns the signal level of the last packet received. This reading is useful for determining range characteristics of the modules under various conditions of noise and distance. Once the command is issued, the module will return a value between 0x6 and 0x36 where 0x36 represents a very strong signal level and 0x4 indicates a low signal level.

RT (DI2 Configuration) Command**Command Summary**

AT Command: ATRT
 Binary Command: 0x16 (22 decimal)
 Parameter Range: 0 – 2

| Parameter Value | Configuration |
|-----------------|---------------------------|
| 0 | disabled |
| 1 | Enable Binary Programming |
| 2 | Enable RTS Flow Control |

Default Parameter Value: 0

Number of bytes returned: 1

Description

<Serial Interfacing> RT command is used to dictate the behavior of the DI2/RTS/CMD line. RT Command must be issued to enable RTS flow control or binary programming.

SB (Stop Bits) Command**Command Summary**

AT Command: ATSB
 Binary Command: 0x36 (54 decimal)
 Parameter Range: 0 – 1

| Parameter Value | Configuration |
|-----------------|---------------|
| 0 | 1 stop bits |
| 1 | 2 stop bits |

Default Parameter Value: 0

Number of bytes returned: 1

Description

SB Command is used to set/read the number of stop bits in the data packets.

SH (Serial Number High) Command**Command Summary**

AT Command: ATSH
 Binary Command: 0x25 (37 decimal)
 Parameter Range: 0 – 0xFFFF [read-only]
 Number of bytes returned: 2
 Related Commands: SL (Serial Number Low)

Description

<Diagnostics> Read the serial number high word of the module.

SL (Serial Number Low) Command**Command Summary**

| |
|---|
| AT Command: ATSL |
| Binary Command: 0x26 (38 decimal) |
| Parameter Range: 0 – 0xFFFF [read-only] |
| Number of bytes returned: 2 |
| Related Commands: SH (Serial Number High) |

Description

<Diagnostics> Read the serial number low word of the module.

SM (Sleep Mode) Command**Command Summary**

| |
|---------------------------|
| AT Command: ATSM |
| Binary Command: 0x01 |
| Parameter Range: 0, 1 3–8 |

| Parameter Value | Configuration |
|-----------------|--|
| 0 | Disabled |
| 1 | Pin Sleep |
| 3 | Cyclic 0.5 second sleep (Module wakes every 0.5 seconds) |
| 4 | Cyclic 1.0 second sleep |
| 5 | Cyclic 2.0 second sleep |
| 6 | Cyclic 4.0 second sleep |
| 7 | Cyclic 8.0 second sleep |
| 8 | Cyclic 16.0 second sleep |

Default Parameter Value: 0

Number of bytes returned: 1

Related Commands:

For Pin Sleep – PC (Power-up Mode), PW (Pin Wake-up)

For Serial Port Sleep – ST (Time before Sleep)

For Cyclic Sleep – ST (Time before Sleep), LH (Wake-up Initializer Timer), HT (Time Before Wake-up Initializer), PW (Pin Wake-up)

Description

<Sleep Mode (Low Power)> SM Command is used to adjust Sleep Mode settings. By default, Sleep Mode is disabled and the module remains continually active. SM Command allows the module to run in a lower-powerstate and be configured in one of eight settings. Cyclic Sleep settings wake the module after the amount of time designated by SM Command. If the module detects a wake-up initializer during the time it is awake, it will synchronize with the transmitter and start receiving data after the wake-up initializer runs its duration. Otherwise, it returns to Sleep Mode and continue to cycle in and out of inactivity until the Wake-up Initializer is detected. If a Cyclic Sleep setting is chosen, the ST, LH and HT parameters must also be set as described in the “Sleep Mode” section of this manual.

ST (Time before Sleep) Command**Command Summary**

| |
|---|
| AT Command: ATST |
| Binary Command: 0x02 |
| Parameter Range: 0x10 – 0xFFFF [x 100 milliseconds] |
| Default Parameter Value: 0x64 (100 decimal) |
| Number of bytes returned: 2 |
| Related Commands: SM (Sleep Mode), LH (Wake-up Initializer Timer), HT (Time before Wake-up Initializer) |

Description

<Sleep Mode (Low Power)> ST Command sets the period of time (in tenths of seconds) in which the module remains inactive before entering into Sleep Mode. For example, if the ST Parameter is set to 0x64 (100 decimal), the module will enter into Sleep mode after 10 seconds of inactivity (no transmitting or receiving). This command can only be used if Cyclic Sleep or Serial Port Sleep Mode settings have been selected using SM (Sleep Mode) Command.

SY (Time before Initialization) Command

| Command Summary | Description |
|---|--|
| AT Command: ATSY | <p><Networking> SY Command keeps a communication channel open as long as module transmits or receives before the active connection expires. It can be used to reduce latency in a query/response sequence and should be set 100 ms longer than the delay between transmissions. This command allows multiple Modules to share a hopping channel for a given amount of time after receiving data. By default, all packets include an RF initializer that contains channel information used to synchronize any listening receivers to the transmitter's hopping pattern. Once a new module comes within range, it is able to instantly synchronize to the transmitter and start receiving data. If no new modules are introduced into the system, the synchronization information becomes redundant once modules have become synchronized.</p> <p>SY Command allows the modules to remove this information from the RF Initializer after the initial synchronization. For example, changing the SY Parameter to 0x14 (20 decimal) allows all modules to remain in sync for 2 seconds after the last data packet was received. Synchronization information is not re-sent unless transmission stops for more than 2 seconds. This command allows significant savings in packet transmission time.</p> <p>Warning: Not recommended for use in an interference-prone environment. Interference can break up the session and the communications channel will not be available again until SY time expires. With SY set to zero, the channel session is opened and closed with each transmission – resulting in a more robust link with more latency.</p> |
| Binary Command: 0x17 (23 decimal) | |
| Parameter Range: 0 – 0xFF [x 100 milliseconds] | |
| Default Parameter Value: 0 (Disabled – channel initialization information is sent with each RF packet.) | |
| Number of bytes returned: 1 | |

TR (Transmit Error Count) Command

| Command Summary | Description |
|-----------------------------------|---|
| AT Command: ATTR | <p><Diagnostics> TR Command records the number of retransmit failures. This number is incremented each time a packet is not acknowledged within the number of retransmits specified by the RR (Retries) Command. It therefore counts the number of packets that were not successfully received and have been dropped. The TR Parameter is not non-volatile and will therefore be reset to zero each time the module is reset.</p> |
| Binary Command: 0x1B (27 decimal) | |
| Parameter Range: 0 – 0xFFFF | |
| Default Parameter Value: 0 | |
| Number of bytes returned: 2 | |
| Related Commands: RR (Retries) | |

TT (Streaming Limit) Command

| Command Summary | Description |
|---|---|
| AT Command: ATTT | <p><Networking> TT Command defines a limit on the number of bytes that can be sent out before a random delay is issued. TT Command is used to simulate full-duplex behavior.</p> <p>If a module is sending a continuous stream of RF data, a delay is inserted which stops its transmission and allows other modules time to transmit (once it sends number of bytes specified by TT Command). Inserted random delay lasts between 1 & 'RN + 1' delay slots, where each delay slot lasts 38 ms.</p> |
| Binary Command: 0x1A (26 decimal) | |
| Parameter Range: 0 – 0xFFFF (0 = disabled) | |
| Default Parameter Value: 0xFFFF (65535 decimal) | |
| Number of bytes returned: 2 | |
| Related Commands: RN (Delay Slots) | |


RF Communication Modes

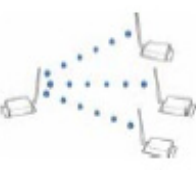

Network configurations covered in this chapter are described in terms of the following:

- Network Topology (Point-to-Point, Point-to-Multipoint or Peer-to-Peer)
- RF Communication Type (Basic or Acknowledged)
- RF Mode (Streaming, Repeater, Acknowledged or Multi-Streaming)

The following table provides a summary of the network configurations supported.

Summary of network configurations supported by the XStream RF Module

| Point-to Point | | |
|---|--|--|
|  | Definition | An RF data link between two modules |
| | Sample Network Profile * (Broadcast Communications) | Use default values for all modules. |
| | Sample Network Profile * (Acknowledged Communications) | All Modules: ATAM [auto-set MY (Source Address) parameter] ** ATDTFFFF [set Destination Address to 0xFFFF] |
| | Basic Communication RF Modes | Streaming Mode [p. 39], Repeater Mode [p. 40] |
| | Acknowledged Communication RF Mode | Acknowledge Mode [p. 43] |
| Point-to -Multipoint | | |

| Point-to-Point | | |
|---|---|---|
|  | Definition | RF Data links between one base and multiple remotes. |
| | Sample Network Profile * (Basic Communications) | Base: ATMY 0 [set Source Address to 0x00] ATDT FFFF [set Destination Address to 0xFFFF] Remotes: ATAM [auto-set MY (Source Address) parameter] ** ATDT 0 [set Destination Address to 0x00] |
| | Sample Network Profile * (Acknowledged Communications) | Base: ATMY 0 [set Source Address to 0x00] ATDT FFFF [set Destination Address to 0xFFFF] ATRR 3 [set number of Retries to 3] Remotes: ATAM [auto-set MY (Source Address) parameter] ** ATDT 0 [set Destination Address to 0x00] ATRR 3 [set number of Retries to 3] |
| | Basic Communication RF Modes | Streaming Mode [p.39], Repeater Mode [p.40] |
| | Acknowledged Communication RF Modes | Acknowledged Mode [p.43] |
| Peer-to-Peer | | |
|  | Definition | Modules remain synchronized without use of a master/server. Each module shares the roles of master and slave. MaxStream's peer-to-peer architecture features fast synch times (35ms to synchronize modules) and fast cold start times (50ms before transmission). |
| | Sample Network Profile * (Basic Communications) | Use default values for all modules. |
| | Sample Network Profile * (Acknowledged Communications) | All Modules: ATAM [auto-set MY (Source Address) parameter] ** ATDT FFFF [set Destination Address to 0xFFFF] ATRR 3 [set number of Retries to 3] |
| | Basic Communication RF Mode | Streaming Mode [p.39] |
| | Acknowledged Communication RF Mode | Acknowledged Mode [p.43] |

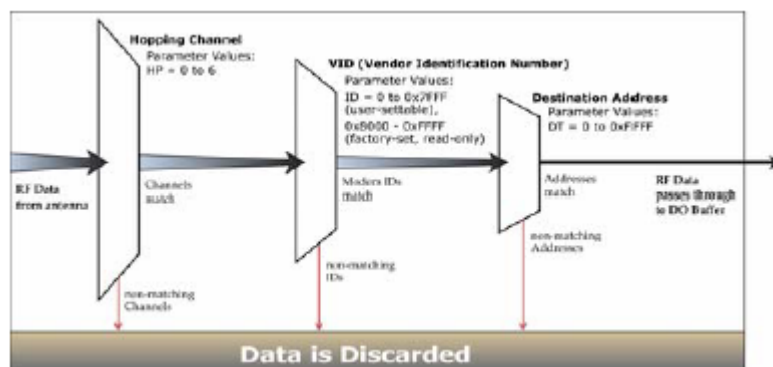
*Assume default values for parameters not listed. Profiles do not reflect addressing implementations.

**AM (Auto-set MY) Command must be issued through a terminal program such as the one incorporated in the X-CTU 'Terminal' tab.

Addressing

Each RF packet contains addressing information that is used to filter incoming RF data. Receiving modules inspect the Hopping Channel (HP parameter), Vendor Identification Number (ID parameter) and Destination Address (DT parameter) contained in each RF packet. Data that does not pass through all three network security layers is discarded.

Filtration layers contained in the RF packet header



Address Recognition

Transmissions can be addressed to a specific module or group of modules using the DT (Destination Address) and MK (Address Mask) parameters. The transmitting module dictates whether the packet is intended for a specific module (local address) or multiple modules (global address) by comparing the packet's DT parameter to its own MK parameter.

Local Packets vs. Global Packets (Transmitting Module)

TX_DT = Transmitter Destination Address

TX_MK = Transmitter Address Mask

Note: When TX_DT = 0xFFFF (default), RF packets are global and are received by all modules within range. (Receivers do not send ACKs.)

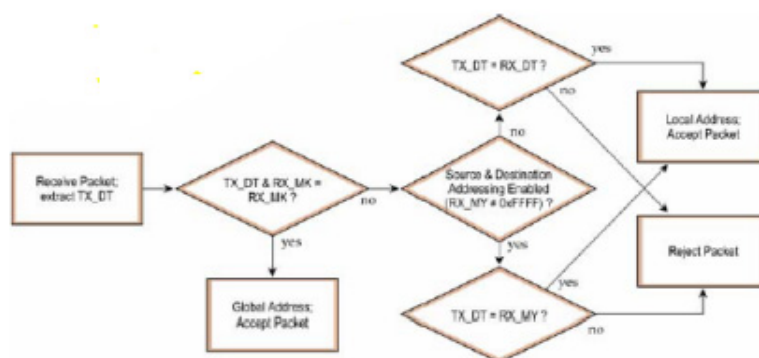
A receiving module will only accept a packet if a packet is addressed to it (either as a global or local packet). The RX module makes this determination by inspecting the destination address of the RF packet and comparing it to its own address and mask. The Destination Address of the TX module is logically "ANDed" with the Address Mask of the RX module.

Address Recognition (Receiving Module)

TX_DT = Transmitter Destination Address

RX_DT = Receiver Destination Address

RX_MY = Receiver Source Address



Basic Communications

Basic Communications are accomplished through two sub-types:

- **Broadcast** - By default, XStream Modules communicate through Broadcast communications and within a peer-to-peer network topology. When any module transmits, all other modules within range will receive the data and pass it directly to their host device.
- **Addressed** - If addressing parameters match, received RF data is forwarded to the DO (Data Out) buffer; otherwise, the RF data is discarded.

When using Basic Communications, any functions such as acknowledgements are handled at the application layer by the integrator. The Broadcast Modes provide transparent communications, meaning that the RF link simply replaces a wired link.

Streaming Mode (Default)

Characteristics: Highest data throughput

Lowest latency and jitter

Reduced immunity to interference

Transmissions never acknowledged (ACK) by receiving module(s)

Required Parameter Values (TX Module): RR (Retries) = 0

Related Commands: Networking (DT, MK, MY), Serial Interfacing (PK, RB, RO, TT)

Recommended Use: Mode is most appropriate for data systems more sensitive to latency and/or jitter than to occasional packet loss.

Streaming Mode Data Flow

Streaming Mode State Diagram (TX Module)

Events & processes in this mode are common to all of the other RF Modes.

NOTE: When streaming data, RB and RO parameters are only observed on the first packet.

After transmission begins, the TX event will continue uninterrupted until the DI buffer is empty or the streaming limit (TT Command) is reached. As with the first packet, the payload of each subsequent packet includes up to the maximum packet size (PK Command).

The streaming limit (TT Command) is specified by the transmitting module as the maximum number of bytes the transmitting module can send in one transmission event. After the TT parameter threshold is reached, the transmitting module will force a random delay of 1 to RN delay slots (exactly 1 delay slot if RN = 0).

Subsequent packets are sent without an RF initializer since receiving modules stay synchronized with the transmitting module for the duration of the transmission event (from preceding packet information). However, due to interference, some receiving modules may lose data (and synchronization to the transmitting module), particularly during long transmission events.

Once the transmitting module has sent all pending data or has reached the TT limit, the transmission event ends. The transmitting module will not transmit again for exactly RN delay slots if the local (i.e. transmitting module's) RN parameter is set to a non-zero value. The receiving module(s) will not transmit for a random number of delays between 0 and (RN-1) if the local (i.e. receiving module's) RN parameter is set to a non-zero value. These delays are intended to lessen congestion following long bursts of packets from a single transmitting module, during which several receiving modules may have become ready to transmit.

Repeater Mode

Characteristics: Self-organizing - No route configuration is necessary

Self-healing / Fault-tolerant

Low power consumption and Minimized interference

Network throughput is determined by number of hops, not by number of repeaters. Multiple repeaters within range of source node count as one hop.

Supports "transparent" multi-drop mode or addressed data filtering mode.

Duplicate RF packets are automatically filtered out.

All packets propagate to every node in the network (filtering rules apply).

Broadcast communications - each packet comes out every node exactly once.

Addressed communications - all radios see every packet. Only the module with a matching address will forward it to the DO buffer (UART IN).

Data entering the network on any module is transmitted and forwarded through every repeater module until it reaches the ends of the network.

Each repeater will repeat a packet only once.

Constraints: Requires that each module have a unique MY (Source Address) parameter.

System must introduce just one packet at a time to the network for transmission (256 bytes max).

Each hop (H) decreases network throughput by a factor of $1/(H+1)$. Additional repeaters add network redundancy without decreasing throughput.

Required Parameter Values (TX Module): MD = 3 or 4, MY = unique value (can be accomplished by issuing the AM (Auto-set MY) and WR (Write) commands to all modules in the network).

Related Commands: Networking (MD, DT, MY, AM), Serial Interfacing (RN, PK, RO, RB).

Recommended Use: Use in networks where intermediary nodes are needed to relay data to modules that are beyond the transmission range of the base module.

Theory of Operation

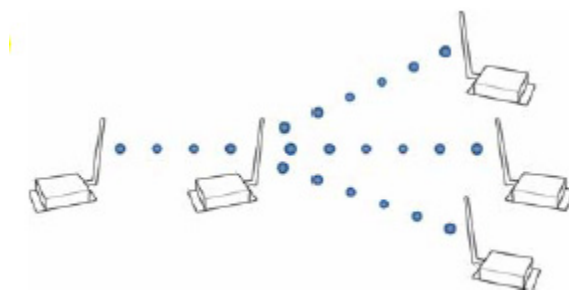
Integrators can extend the effective range and reliability of a data radio system by forwarding traffic through one or more repeaters.

Instead of using routing tables and path discovery to establish dynamic paths through a network, the repeater system uses a sophisticated algorithm to propagate each RF packet through the entire network.

The network supports RF packets of up to 256 bytes. The repeater network can operate using broadcast or addressed communications for multi-drop networks and works well in many systems with no special configuration.

When in Repeater Mode, the network repeats each message among all available nodes exactly one time. This mechanism eliminates the need for configuring specific routes. The network is self-organizing and self-healing so that the system is able to receive transmissions in the event of a module going down.

Sample Repeater Network Topology



Repeater Network Configuration

A network may consist of End Nodes (EN), End/Repeater Nodes (ERN) and a Base Node (BN). The base node initiates all communications.

The repeater network can be configured to operate using Basic Broadcast or Basic Addressed communications. The addressing capabilities of the modules allow integrators to send a packet as a global packet (DT = 0xFFFF) and shift out of every radio in the network (Basic Broadcast). Alternatively, the packet can be sent with a specific DT (Destination Address) parameter so that it is only accepted by a specific remote node (Basic Addressed).

Configuration Instruction (Basic Broadcast Communications)

Assign each module a unique MY (source) address. (The AM (Auto-set MY) command will configure a unique source address that is based on module serial number.)

Enable Basic Broadcast Communications (DT = 0xFFFF) or Addressed Broadcast Communications (ATDT specifies a specific destination)

Configure PK, RO and RB to ensure that RF packet aligns with protocol packet. (ex. PK=0x100, RB=0x100, RO depends on baud rate).

Configure one or more repeaters in the system (ATMD = 3).

Configure remote nodes as destinations (MD = 4). This will ensure that the remote node waits for the repeater traffic to subside before it transmits a response.

The configuration instructions above reflect configuration for a Basic Broadcast Repeater system. To configure a Basic Addressed Repeater system, use the DT (Destination Address) parameter to assign unique addresses to each module in the network.

Algorithm details

- Packet ID (PID) is composed of transmitting module MY address and packet serial number.
- Incoming packets with a PID already found in the PID buffer will be ignored.
- Each module maintains a PID buffer 8 deep of previously received packets (managed as FIFO).

Packets may be shifted out the serial port and/or repeated depending on the DT parameter contained in the RF packet.

DT (Destination Address) parameter truth table

| Address Match | Send out serial port? | Repeat? |
|---------------|-----------------------|---------|
| Global | Yes | Yes |
| Local | Yes | Yes |
| None | No | Yes |

Repeat delay based on RSSI

A transmitted packet may be received by more than one repeater at the same time. In order to reduce the probability that the repeaters will transmit at the same instant, resulting in a collision and possible data loss; an algorithm has been developed that will allow a variable back-off prior to retransmission of the packet by a repeater. The algorithm allows radios that receive the packet with a stronger RF signal (RSSI) to have the first opportunity to retransmit the packet.

The RN (Delay Slots) parameter is used to configure this delay. Set RN=0 (no delays) for small networks with few repeaters or repeaters that are not within range of each other. Set RN=1 for systems with 2 to 5 repeaters that may be within range of each other.

The actual length of the delay is computed by the formula:

$$\text{Delay (ms)} = L * DS$$

$$DS = (-41 - \text{RSSI}) / 10 * RN + \text{RandomInt}(0, RN)$$

Where L is the length of the transmitted packet in milliseconds, DS is the number of delay slots to wait, RSSI is the received signal strength in dBm, RN is the value of the RN register and RandomInt(A,B) is a function that returns a random integer from A to B-0.

Response packet delay

As a packet propagates through the repeater network, if any node receives the data and generates a quick response, the response needs to be delayed so as not to collide with subsequent retransmissions of the original packet. To reduce collisions, both repeater and end node radios in a repeater network will delay transmission of data shifted in the serial port to allow any repeaters within range to complete their retransmissions.

The time for this delay is computed by the formula:

$$\text{Maximum Delay (ms)} = L * DS$$

$$DS = ((-41 - (-100)) / 10 * RN) + RN + 1$$

Where L is the length of the transmitted packet in milliseconds, DS is the number of delay slots to wait, RSSI is the received signal strength in dBm, and RN is the value of the RN register.

Use Case - Broadcast Repeater Network

Consider modules R1 through R10 each communicating to a PLC using the ModBus protocol and spaced evenly in a line. All ten nodes are configured as 'destinations & repeaters' within the scope of Basic Broadcast Communications (MD=3, AM, DT=0xFFFF, PK=0x100, RO=0x03, RB=0x100, RN=1). The Base Host (BH) shifts payload that is destined for R10 to R1. R1 initializes RF communication and transmits payload to nodes R2 through R5 which are all within range of R1. Modules R2 through R5 receive the RF packet and retransmit the packet simultaneously. They also send the data out the serial ports, to the PLC's.

Commands used to configure repeater functions

| AT Command | Binary Command | AT Command Name | Range | # Bytes Returned | Factory Default |
|------------|----------------|---------------------|----------------|------------------|-----------------|
| AM | 0x3A (58d) | Auto-set MY | - | - | - |
| DT | 0x00 (0d) | Destination Address | 0-0xFFFF | 2 | 0 |
| MD | 0x3C (60d) | RF Mode | 3-4 | 1 | 0 |
| MY | 0x2A (42d) | Source Address | 0-0xFFFF | 2 | 0xFFFF |
| RN | 0x19 (25d) | Delay Slots | 0-0xFF [slots] | 1 | 0 |
| WR | 0x08 (8d) | Write | - | - | - |

Bandwidth Considerations

Using broadcast repeaters in a network reduces the overall network data throughput as each repeater must buffer an entire packet before retransmitting it. For example: if the destination is within range of the transmitter and the packet is 32 bytes long, the transmission will take approximately 72ms on a 9600 baud XSC Module. If that same packet has to propagate through two repeaters, it will take 72ms to arrive at the first repeater, another 72 ms to get to the second and a final 72ms to get to the destination for a total of 216ms. Taking into account UART transfer times (~1ms/byte at 9600 baud), a server to send a 32 byte query and receive a 32 byte response is ~200ms, allowing for 5 polls per second. With the two repeaters in the path, the same query/response sequence would take about 500ms for 2 polls per second.

To summarize, this system is sending and receiving 64 bytes 5 times per second for a throughput of 320 bytes per second with no repeaters and 128 bytes per second with 2 repeaters. Generally, the network throughput will decrease by a factor of $1/(R+1)$, with R representing the number of repeaters between the source and destination.

Acknowledged Communications

Acknowledged Mode

Characteristics: Reliable delivery through positive acknowledgements for each packet
Throughput, latency and jitter vary depending on the quality of the channel and the strength of the signal.

Recommended Use: Acknowledge Mode configuration is appropriate when reliable delivery is required between modules. If messages are smaller than 256 bytes, use RB and RO commands to align RF packets with application packets.

Required Parameter Values (TX Module): RR (Retries) ≥ 1

Related Commands: Networking (DT, MK, RR), Serial Interfacing (PK, RN, TT, RO, RB)

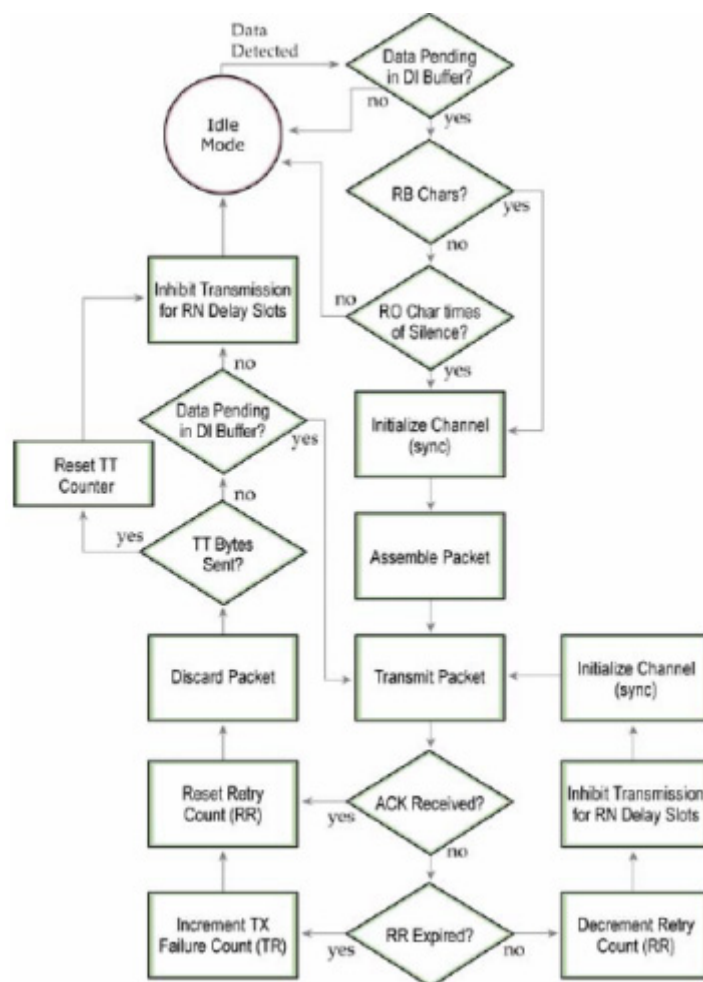
Sample Network Profile

| Module | Parameter Settings (assume default values for parameter not listed) |
|--------|---|
| All | ATRR A [set number of Retries to 0x0A] ATRN 5 [set number of Delay Slots to 5] |

Acknowledged Mode Connection Sequence

Acknowledged Mode State Diagram

After sending a packet while in Acknowledged Mode, the transmitting module listens for the ACK (acknowledgement). If it receives the ACK, it will either send a subsequent packet (if more transmit data is pending), or will wait for exactly RN random delay slots before allowing another transmission (if no more data is pending for transmission). If the transmitting module does not receive the ACK within the allotted time, it will retransmit the packet with a new RF initializer following the ACK slot. There is no delay between the first ACK slot and the first retransmission. Subsequent retransmissions incur a delay of a random number of delay slots, between 0 and RN. If RN is set to 0 on the transmitting module, there are never any back-off delays between retransmissions. Note that during back-off delays, the transmitting module will go into Idle Mode and may receive RF data. This can have the effect of increasing the back-off delay, as the radio cannot return to RF transmit (or retransmit) mode as long as it is receiving RF data.



After receiving and acknowledging a packet, the receiving module will move to the next frequency and listen for either a retransmission or new data for a specific period of time. Even if the transmitting module has indicated that it has no more pending transmit data, it may have not

received the previous ACK, and so it may retransmit the packet (potentially with no delay after the ACK slot). In this case, the receiving module will always detect the immediate retransmission, which will hold off the communications channel and thereby reduce collisions. Receiving modules acknowledge each retransmission they receive, but they only pass the first copy of a packet they receive out the UART. RB and RO parameters are not applied to subsequent packets. This means that once transmission has begun, it will continue uninterrupted until the DI buffer is empty or the streaming limit (TT) has been reached. As with the first packet, the payload of each subsequent packet includes up to the maximum packet size (PK parameter). The transmitting module checks for more pending data near the end of each packet. The streaming limit (TT parameter) specifies the maximum number of bytes that the transmitting module will send in one transmission event, which may consist of many packets and retries. If the TT parameter is reached, the transmitting module will force a random delay of 1 to RN delay slots (exactly 1 delay slot if RN is zero). Each packet is counted only once toward TT, no matter how many times the packet is retransmitted. Subsequent packets in acknowledged mode are similar to those in streaming mode, with the addition of an acknowledgement between each packet, and the possibility of retransmissions. Subsequent packets are sent without an RF initializer, as the receiving modules are already synchronized to the transmitting module from the preceding packet(s) and they remain synchronized for the duration of the transmission event. Each retransmission of a packet includes an RF initializer. Once the transmitting module has sent all pending data or has reached the TT limit, the acknowledged transmission event is completed. The transmitting module will not transmit again for exactly RN delay slots, if the local RN parameter is set to a nonzero value. The receiving module will not transmit for a random number of delay slots between 0 and (RN-1), if the local RN parameter is set to a nonzero value. These delays are intended to lessen congestion following long bursts of packets from a single transmitting module, during which several receiving modules may have themselves become ready to transmit.

Appendix B: Agency Certifications for S3B Hardware

Please note that both Appendix B and Appendix C contain Agency Certification information. Please refer to the Preface for instructions on which appendix applies to your product.

FCC (United States) Certification

The XBee-PRO® 900HP/XBee-PRO® XSC RF Module complies with Part 15 of the FCC rules and regulations. Compliance with the labeling requirements, FCC notices and antenna usage guidelines is required.

In order to operate under Digi's FCC Certification, RF Modules/integrators must comply with the following regulations:

1. The system integrator must ensure that the text provided with this device [Figure A-01] is placed on the outside of the final product and within the final product operation manual.
2. The XBee-PRO® 900HP/XBee-PRO® XSC RF Module may only be used with antennas that have been tested and approved for use with this module refer to Table A-1.

Labeling Requirements



WARNING: The Original Equipment Manufacturer (OEM) must ensure that FCC labeling requirements are met. This includes a clearly visible label on the outside of the final product enclosure that displays the text shown in the figure below.

Figure A-01. Required FCC Label for OEM products containing the XBee-PRO® 900HP/XBee-PRO® XSC RF Module.

XBEE PRO 900HP

Contains FCC ID: MCQ-XB900HP

The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (i.) this device may not cause harmful interference and (ii.) this device must accept any interference received, including interference that may cause undesired operation.

FCC Notices

IMPORTANT: The XBee-PRO® 900HP/XBee-PRO® XSC OEM RF Module has been certified by the FCC for use with other products without any further certification (as per FCC section 2.1091). Modifications not expressly approved by Digi could void the user's authority to operate the equipment.

IMPORTANT: OEMs must test final product to comply with unintentional radiators (FCC section 15.107 & 15.109) before declaring compliance of their final product to Part 15 of the FCC Rules.

IMPORTANT: The RF module has been certified for remote and base radio applications. If the module will be used for portable applications, the device must undergo SAR testing.

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures: Re-orient or relocate the receiving antenna, Increase the separation between the

equipment and receiver, Connect equipment and receiver to outlets on different circuits, or Consult the dealer or an experienced radio/TV technician for help.

Limited Modular Approval

This is an RF module approved for Limited Modular use operating as a mobile transmitting device with respect to section 2.1091 and is limited to OEM installation for Mobile and Fixed applications only. During final installation, end-users are prohibited from access to any programming parameters. Professional installation adjustment is required for setting module power and antenna gain to meet EIRP compliance for high gain antenna(s).

Final antenna installation and operating configurations of this transmitter including antenna gain and cable loss must not exceed the EIRP of the configuration used for calculating MPE. Grantee (Digi) must coordinate with OEM integrators to ensure the end-users and installers of products operating with the module are provided with operating instructions to satisfy RF exposure requirements.

The FCC grant is valid only when the device is sold to OEM integrators. Integrators are instructed to ensure the end-user has no manual instructions to remove, adjust or install the device.

FCC-approved Antennas



WARNING: This device has been tested with Reverse Polarity SMA connectors with the antennas listed in the tables of this section. When integrated into OEM products, fixed antennas require installation preventing end-users from replacing them with non-approved antennas. Antennas not listed in the tables must be tested to comply with FCC Section 15.203 (unique antenna connectors) and Section 15.247 (emissions).

WARNING: The FCC requires that all spread spectrum devices operating within the Unlicensed radio frequency bands must limit themselves to a maximum radiated power of 4 Watts EIRP. Failure to observe this limit is a violation of our warranty terms, and shall void the user's authority to operate the equipment.

This can be stated: RF power - cable loss + antenna gain \leq 36 dBm eirp.

Fixed Base Station and Mobile Applications

Digi RF Modules are pre-FCC approved for use in fixed base station and mobile applications. When the antenna is mounted at least 20cm (8") from nearby persons, the application is considered a mobile application.

Portable Applications and SAR Testing

If the module will be used at distances closer than 20cm to all persons, the device may be required to undergo SAR testing. Co-location with other transmitting antennas closer than 20cm should be avoided.

RF Exposure

This statement must be included as a CAUTION statement in OEM product manuals.



WARNING: This equipment is approved only for mobile and base station transmitting devices. Antenna(s) used for this transmitter must be installed to provide a separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter.

IC (Industry Canada) Certification

This device complies with Industry Canada licence-exempt RSS standard(s). Operation is subject to the following two conditions: (1) this device may not cause interference, and (2) this device must accept any interference, including interference that may cause undesired operation of the device.

Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes: (1) l'appareil ne doit pas produire de brouillage, et (2) l'utilisateur de l'appareil doit accepter

tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement

Labeling Requirements

Labeling requirements for Industry Canada are similar to those of the FCC. A clearly visible label on the outside of the final product enclosure must display one of the following text:

Contains IC: 1846A-XB900HP

The integrator is responsible for its product to comply with IC ICES-003 & FCC Part 15, Sub. B-Unintentional Radiators. ICES-003 is the same as FCC Part 15 Sub. B and Industry Canada accepts FCC test report or CISPR 22 test report for compliance with ICES-003.

Antenna Options: 900 MHz Antenna Listings

The antennas in the tables below have been approved for use with this module. Digi does not carry all of these antenna variants. Contact Digi Sales for available antennas.

Table A-01. Antennas approved for use with the XBee-PRO 900HP RF Module

| Part Number | Type | Connector | Gain | Application | Cable Loss or Power Reduction for S3B Radio |
|----------------------------------|-------------------|-----------|---------|-------------|---|
| Omni-directional antennas | | | | | |
| A09-F0 | Fiberglass Base | RPN | 0 dBi | Fixed | 0dB |
| A09-F1 | Fiberglass Base | RPN | 1.0 dBi | Fixed | 0dB |
| A09-F2 | Fiberglass Base | RPN | 2.1 dBi | Fixed | 0dB |
| A09-F3 | Fiberglass Base | RPN | 3.1 dBi | Fixed | 0dB |
| A09-F4 | Fiberglass Base | RPN | 4.1 dBi | Fixed | 0dB |
| A09-F5 | Fiberglass Base | RPN | 5.1 dBi | Fixed | 0dB |
| A09-F6 | Fiberglass Base | RPN | 6.1 dBi | Fixed | 0dB |
| A09-F7 | Fiberglass Base | RPN | 7.1 dBi | Fixed | 0dB |
| A09-F8 | Fiberglass Base | RPN | 8.1 dBi | Fixed | 0dB |
| A09-F9 | Base Station | RPSMAF | 9.2dBi | Fixed | 0dB |
| A09-W7 | Wire Base Station | RPN | 7.1 dBi | Fixed | 0dB |
| A09-F0 | Fiberglass Base | RPSMA | 0 dBi | Fixed | 0dB |
| A09-F1 | Fiberglass Base | RPSMA | 1.0 dBi | Fixed | 0dB |
| A09-F2 | Fiberglass Base | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-F3 | Fiberglass Base | RPSMA | 3.1 dBi | Fixed | 0dB |
| A09-F4 | Fiberglass Base | RPSMA | 4.1 dBi | Fixed | 0dB |
| A09-F5 | Fiberglass Base | RPSMA | 5.1 dBi | Fixed | 0dB |
| A09-F6 | Fiberglass Base | RPSMA | 6.1 dBi | Fixed | 0dB |
| A09-F7 | Fiberglass Base | RPSMA | 7.1 dBi | Fixed | 0dB |
| A09-F8 | Fiberglass Base | RPSMA | 8.1 dBi | Fixed | 0dB |
| A09-M7 | Base Station | RPSMAF | 7.2dBi | Fixed | 0dB |
| A09-W7SM | Wire Base Station | RPSMA | 7.1 dBi | Fixed | 0dB |
| A09-F0TM | Fiberglass Base | RPTNC | 0 dBi | Fixed | 0dB |
| A09-F1TM | Fiberglass Base | RPTNC | 1.0 dBi | Fixed | 0dB |
| A09-F2TM | Fiberglass Base | RPTNC | 2.1 dBi | Fixed | 0dB |
| A09-F3TM | Fiberglass Base | RPTNC | 3.1 dBi | Fixed | 0dB |

| | | | | | |
|-----------------|----------------------------------|------------|----------|----------------|-----|
| A09-F4TM | Fiberglass Base | RPTNC | 4.1 dBi | Fixed | 0dB |
| A09-F5TM | Fiberglass Base | RPTNC | 5.1 dBi | Fixed | 0dB |
| A09-F6TM | Fiberglass Base | RPTNC | 6.1 dBi | Fixed | 0dB |
| A09-F7TM | Fiberglass Base | RPTNC | 7.1 dBi | Fixed | 0dB |
| A09-F8TM | Fiberglass Base | RPTNC | 8.1 dBi | Fixed | 0dB |
| A09-W7TM | Wire Base Station | RPTNC | 7.1 dBi | Fixed | 0dB |
| A09-HSM-7 | Straight half-wave | RPSMA | 3.0 dBi | Fixed / Mobile | 0dB |
| A09-HASM-675 | Articulated half- | RPSMA | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-HABMM-P6I | Articulated half- | MMCX | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-HABMM-6-P6I | Articulated half- | MMCX | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-HBMM-P6I | Straight half-wave | MMCX | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-HRSM | Right angle half- | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-HASM-7 | Articulated half- | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-HG | Glass mounted | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-HATM | Articulated half- | RPTNC | 2.1 dBi | Fixed | 0dB |
| A09-H | Half-wave dipole | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-HBMMP6I | 1/2 wave antenna | MMCX | 2.1dBi | Mobile | 0dB |
| A09-QBMP6I | 1/4 wave antenna | MMCX | 1.9 dBi | Mobile | 0dB |
| A09-QI | 1/4 wave integrated wire antenna | Integrated | 1.9 dBi | Mobile | 0dB |
| 29000187 | Helical | Integrated | -2.0 dBi | Fixed/Mobile | 0dB |
| A09-QW | Quarter-wave wire | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QRAMM | 3 "Quarter-wave | MMCX | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-QSM-3 | Quarter-wave | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QSM-3H | Heavy duty quarter- | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QBMM-P6I | Quarter-wave w/ 6" | MMCX | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QHRN | Miniature Helical | Permanent | -1 dBi | Fixed / Mobile | 0dB |
| A09-QHSN | Miniature Helical | Permanent | -1 dBi | Fixed / Mobile | 0dB |
| A09-QHSM-2 | 2" Straight | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QHRSM-2 | 2" Right angle | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QHRSM-170 | 1.7" Right angle | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QRSM-380 | 3.8" Right angle | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QAPM-520 | 5.2" Articulated | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QSPM-3 | 3" Straight screw | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QAPM-3 | 3" Articulated screw | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QAPM-3H | 3" Articulated screw | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-DPSM-P12F | omni directional | RPSMA | 3.0 dBi | Fixed | 0dB |
| A09-D3NF-P12F | omni directional | RPN | 3.0 dBi | Fixed | 0dB |
| A09-D3SM-P12F | omni directional w/ | RPSMA | 3.0 dBi | Fixed | 0dB |
| A09-D3PNF | omni directional | RPN | 3.0 dBi | Fixed | 0dB |
| A09-D3TM-P12F | omni directional w/ | RPTNC | 3.0 dBi | Fixed | 0dB |
| A09-D3PTM | omni directional | RPTNC | 3.0 dBi | Fixed | 0dB |
| A09-M0SM | Mag Mount | RPSMA | 0 dBi | Fixed | 0dB |
| A09-M2SM | Mag Mount | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-M3SM | Mag Mount | RPSMA | 3.1 dBi | Fixed | 0dB |
| A09-M5SM | Mag Mount | RPSMA | 5.1 dBi | Fixed | 0dB |
| A09-M7SM | Mag Mount | RPSMA | 7.1 dBi | Fixed | 0dB |
| A09-M8SM | Mag Mount | RPSMA | 8.1 dBi | Fixed | 0dB |

| | | | | | |
|----------------------|-----------------|-------|----------|----------------|-------|
| A09-M0TM | Mag Mount | RPTNC | 0 dBi | Fixed | 0dB |
| A09-M2TM | Mag Mount | RPTNC | 2.1 dBi | Fixed | 0dB |
| A09-M3TM | Mag Mount | RPTNC | 3.1 dBi | Fixed | 0dB |
| A09-M5TM | Mag Mount | RPTNC | 5.1 dBi | Fixed | 0dB |
| A09-M7TM | Mag Mount | RPTNC | 7.1 dBi | Fixed | 0dB |
| A09-M8TM | Mag Mount | RPTNC | 8.1 dBi | Fixed | 0dB |
| Yagi antennas | | | | | |
| A09-Y6 | 2 Element Yagi | RPN | 6.1 dBi | Fixed / Mobile | 0dB |
| A09-Y7 | 3 Element Yagi | RPN | 7.1 dBi | Fixed / Mobile | 0dB |
| A09-Y8 | 4 Element Yagi | RPN | 8.1 dBi | Fixed / Mobile | 0dB |
| A09-Y9 | 4 Element Yagi | RPN | 9.1 dBi | Fixed / Mobile | 0dB |
| A09-Y10 | 5 Element Yagi | RPN | 10.1 dBi | Fixed / Mobile | 0dB |
| A09-Y11 | 6 Element Yagi | RPN | 11.1 dBi | Fixed / Mobile | 0dB |
| A09-Y12 | 7 Element Yagi | RPN | 12.1 dBi | Fixed / Mobile | 0dB |
| A09-Y13 | 9 Element Yagi | RPN | 13.1 dBi | Fixed / Mobile | 0.8dB |
| A09-Y14 | 10 Element Yagi | RPN | 14.1 dBi | Fixed / Mobile | 1.8dB |
| A09-Y14 | 12 Element Yagi | RPN | 14.1 dBi | Fixed / Mobile | 1.8dB |
| A09-Y15 | 13 Element Yagi | RPN | 15.1 dBi | Fixed / Mobile | 2.8dB |
| A09-Y15 | 15 Element Yagi | RPN | 15.1 dBi | Fixed / Mobile | 2.8dB |
| A09-Y6TM | 2 Element Yagi | RPTNC | 6.1 dBi | Fixed / Mobile | 0dB |
| A09-Y7TM | 3 Element Yagi | RPTNC | 7.1 dBi | Fixed / Mobile | 0dB |
| A09-Y8TM | 4 Element Yagi | RPTNC | 8.1 dBi | Fixed / Mobile | 0dB |
| A09-Y9TM | 4 Element Yagi | RPTNC | 9.1 dBi | Fixed / Mobile | 0dB |
| A09-Y10TM | 5 Element Yagi | RPTNC | 10.1 dBi | Fixed / Mobile | 0dB |
| A09-Y11TM | 6 Element Yagi | RPTNC | 11.1 dBi | Fixed / Mobile | 0dB |
| A09-Y12TM | 7 Element Yagi | RPTNC | 12.1 dBi | Fixed / Mobile | 0dB |
| A09-Y13TM | 9 Element Yagi | RPTNC | 13.1 dBi | Fixed / Mobile | 0.8dB |
| A09-Y14TM | 10 Element Yagi | RPTNC | 14.1 dBi | Fixed / Mobile | 1.8dB |
| A09-Y14TM | 12 Element Yagi | RPTNC | 14.1 dBi | Fixed / Mobile | 1.8dB |
| A09-Y15TM | 13 Element Yagi | RPTNC | 15.1 dBi | Fixed / Mobile | 2.8dB |
| A09-Y15TM | 15 Element Yagi | RPTNC | 15.1 dBi | Fixed / Mobile | 2.8dB |

Transmitters with Detachable Antennas

This radio transmitter (IC: 1846A-XBEE900HP) has been approved by Industry Canada to operate with the antenna types listed in the table above with the maximum permissible gain and required antenna impedance for each antenna type indicated. Antenna types not included in this list, having a gain greater than the maximum gain indicated for that type, are strictly prohibited for use with this device.

Le présent émetteur radio (IC: 1846A-XBEE900HP) a été approuvé par Industrie Canada pour fonctionner avec les types d'antenne énumérés ci-dessous et ayant un gain admissible maximal et l'impédance requise pour chaque type d'antenne. Les types d'antenne non inclus dans cette liste, ou dont le gain est supérieur au gain maximal indiqué, sont strictement interdits pour l'exploitation de l'émetteur.

Detachable Antenna

Under Industry Canada regulations, this radio transmitter may only operate using an antenna of a type and maximum (or lesser) gain approved for the transmitter by Industry Canada. To reduce potential radio interference to other users, the antenna type and its gain should be so chosen that the equivalent isotropically radiated power (e.i.r.p.) is not more than that necessary for successful communication.

Conformément à la réglementation d'Industrie Canada, le présent émetteur radio peut fonctionner avec une antenne d'un type et d'un gain maximal (ou inférieur) approuvé pour l'émetteur par Industrie Canada. Dans le but de réduire les risques de brouillage radioélectrique à l'intention des autres utilisateurs, il faut choisir le type

d'antenne et son gain de sorte que la puissance isotrope rayonnée équivalente (p.i.r.e.) ne dépasse pas l'intensité nécessaire à l'établissement d'une communication satisfaisante.

Appendix C: Agency Certifications for Legacy S3/S3B Hardware

Please note that both Appendix B and Appendix C contain Agency Certification information. Please refer to the Preface for instructions on which appendix applies to your product.

FCC (United States) Certification

The XBee-PRO® XSC RF Module complies with Part 15 of the FCC rules and regulations. Compliance with the labeling requirements, FCC notices and antenna usage guidelines is required.

In order to operate under Digi's FCC Certification, RF Modules/integrators must comply with the following regulations:

1. The system integrator must ensure that the text provided with this device [Figure A-01] is placed on the outside of the final product and within the final product operation manual.
2. The XBee-PRO® XSC RF Module may only be used with antennas that have been tested and approved for use with this module refer to Table A-1.

Labeling Requirements



WARNING: The Original Equipment Manufacturer (OEM) must ensure that FCC labeling requirements are met. This includes a clearly visible label on the outside of the final product enclosure that displays the text shown in the figure below.

Figure A-01. Required FCC Label for OEM products containing the XBee-PRO® XSC RF Module.

XBEE PRO S3

Contains FCC ID: MCQ-XBEEEXSC

The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (i.) this device may not cause harmful interference and (ii.) this device must accept any interference received, including interference that may cause undesired operation.

OR

XBEE PRO S3B

Contains FCC ID: MCQ-XBPS3B

The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (i.) this device may not cause harmful interference and (ii.) this device must accept any interference received, including interference that may cause undesired operation.

FCC Notices

IMPORTANT: The XBee-PRO® XSC OEM RF Module has been certified by the FCC for use with other products without any further certification (as per FCC section 2.1091). Modifications not expressly approved by Digi could void the user's authority to operate the equipment.

IMPORTANT: OEMs must test final product to comply with unintentional radiators (FCC section 15.107 & 15.109) before declaring compliance of their final product to Part 15 of the FCC Rules.

IMPORTANT: The RF module has been certified for remote and base radio applications. If the module will be used for portable applications, the device must undergo SAR testing.

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures: Re-orient or relocate the receiving antenna, Increase the separation between the equipment and receiver, Connect equipment and receiver to outlets on different circuits, or Consult the dealer or an experienced radio/TV technician for help.

Limited Modular Approval

This is an RF module approved for Limited Modular use operating as a mobile transmitting device with respect to section 2.1091 and is limited to OEM installation for Mobile and Fixed applications only. During final installation, end-users are prohibited from access to any programming parameters. Professional installation adjustment is required for setting module power and antenna gain to meet EIRP compliance for high gain antenna(s).

Final antenna installation and operating configurations of this transmitter including antenna gain and cable loss must not exceed the EIRP of the configuration used for calculating MPE. Grantee (Digi) must coordinate with OEM integrators to ensure the end-users and installers of products operating with the module are provided with operating instructions to satisfy RF exposure requirements.

The FCC grant is valid only when the device is sold to OEM integrators. Integrators are instructed to ensure the end-user has no manual instructions to remove, adjust or install the device.

FCC-approved Antennas



WARNING: This device has been tested with Reverse Polarity SMA connectors with the antennas listed in the tables of this section. When integrated into OEM products, fixed antennas require installation preventing end-users from replacing them with non-approved antennas. Antennas not listed in the tables must be tested to comply with FCC Section 15.203 (unique antenna connectors) and Section 15.247 (emissions).

Fixed Base Station and Mobile Applications

Digi RF Modules are pre-FCC approved for use in fixed base station and mobile applications. When the antenna is mounted at least 20cm (8") from nearby persons, the application is considered a mobile application.

Portable Applications and SAR Testing

If the module will be used at distances closer than 20cm to all persons, the device may be required to undergo SAR testing. Co-location with other transmitting antennas closer than 20cm should be avoided.

RF Exposure

This statement must be included as a CAUTION statement in OEM product manuals.



WARNING: This equipment is approved only for mobile and base station transmitting devices. Antenna(s) used for this transmitter must be installed to provide a separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter.

IC (Industry Canada) Certification

This device complies with Industry Canada licence-exempt RSS standard(s). Operation is subject to the following two conditions: (1) this device may not cause interference, and (2) this device must accept any interference, including interference that may cause

undesired operation of the device.

Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes: (1) l'appareil ne doit pas produire de brouillage, et (2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement

Labeling Requirements

Labeling requirements for Industry Canada are similar to those of the FCC. A clearly visible label on the outside of the final product enclosure must display one of the following text:

Contains IC: 1846A-XBEEEXSC

OR

Contains IC: 1846A-XBPS3B

The integrator is responsible for its product to comply with IC ICES-003 & FCC Part 15, Sub. B-Unintentional Radiators. ICES-003 is the same as FCC Part 15 Sub. B and Industry Canada accepts FCC test report or CISPR 22 test report for compliance with ICES-003.

Antenna Options: 900 MHz Antenna Listings

Table A-01. Antennas approved for use with the XBee-PRO XSC RF Module

| Part Number | Type | Connector | Gain | Application | Cable Loss or Power Reduction for S3B Radio |
|----------------------------------|-------------------|-----------|---------|-------------|---|
| Omni-directional antennas | | | | | |
| A09-F0 | Fiberglass Base | RPN | 0 dBi | Fixed | 0dB |
| A09-F1 | Fiberglass Base | RPN | 1.0 dBi | Fixed | 0dB |
| A09-F2 | Fiberglass Base | RPN | 2.1 dBi | Fixed | 0dB |
| A09-F3 | Fiberglass Base | RPN | 3.1 dBi | Fixed | 0dB |
| A09-F4 | Fiberglass Base | RPN | 4.1 dBi | Fixed | 0dB |
| A09-F5 | Fiberglass Base | RPN | 5.1 dBi | Fixed | 0dB |
| A09-F6 | Fiberglass Base | RPN | 6.1 dBi | Fixed | 0dB |
| A09-F7 | Fiberglass Base | RPN | 7.1 dBi | Fixed | 0dB |
| A09-F8 | Fiberglass Base | RPN | 8.1 dBi | Fixed | 0dB |
| A09-F9 | Base Station | RPSMAF | 9.2dBi | Fixed | 0dB |
| A09-W7 | Wire Base Station | RPN | 7.1 dBi | Fixed | 0dB |
| A09-F0 | Fiberglass Base | RPSMA | 0 dBi | Fixed | 0dB |
| A09-F1 | Fiberglass Base | RPSMA | 1.0 dBi | Fixed | 0dB |
| A09-F2 | Fiberglass Base | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-F3 | Fiberglass Base | RPSMA | 3.1 dBi | Fixed | 0dB |
| A09-F4 | Fiberglass Base | RPSMA | 4.1 dBi | Fixed | 0dB |
| A09-F5 | Fiberglass Base | RPSMA | 5.1 dBi | Fixed | 0dB |
| A09-F6 | Fiberglass Base | RPSMA | 6.1 dBi | Fixed | 0dB |
| A09-F7 | Fiberglass Base | RPSMA | 7.1 dBi | Fixed | 0dB |
| A09-F8 | Fiberglass Base | RPSMA | 8.1 dBi | Fixed | 0dB |
| A09-M7 | Base Station | RPSMAF | 7.2dBi | Fixed | 0dB |

| | | | | | |
|-----------------|----------------------------------|------------|----------|----------------|-----|
| A09-W7SM | Wire Base Station | RPSMA | 7.1 dBi | Fixed | 0dB |
| A09-F0TM | Fiberglass Base | RPTNC | 0 dBi | Fixed | 0dB |
| A09-F1TM | Fiberglass Base | RPTNC | 1.0 dBi | Fixed | 0dB |
| A09-F2TM | Fiberglass Base | RPTNC | 2.1 dBi | Fixed | 0dB |
| A09-F3TM | Fiberglass Base | RPTNC | 3.1 dBi | Fixed | 0dB |
| A09-F4TM | Fiberglass Base | RPTNC | 4.1 dBi | Fixed | 0dB |
| A09-F5TM | Fiberglass Base | RPTNC | 5.1 dBi | Fixed | 0dB |
| A09-F6TM | Fiberglass Base | RPTNC | 6.1 dBi | Fixed | 0dB |
| A09-F7TM | Fiberglass Base | RPTNC | 7.1 dBi | Fixed | 0dB |
| A09-F8TM | Fiberglass Base | RPTNC | 8.1 dBi | Fixed | 0dB |
| A09-W7TM | Wire Base Station | RPTNC | 7.1 dBi | Fixed | 0dB |
| A09-HSM-7 | Straight half-wave | RPSMA | 3.0 dBi | Fixed / Mobile | 0dB |
| A09-HASM-675 | Articulated half- | RPSMA | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-HABMM-P6I | Articulated half- | MMCX | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-HABMM-6-P6I | Articulated half- | MMCX | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-HBMM-P6I | Straight half-wave | MMCX | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-HRSM | Right angle half- | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-HASM-7 | Articulated half- | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-HG | Glass mounted | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-HATM | Articulated half- | RPTNC | 2.1 dBi | Fixed | 0dB |
| A09-H | Half-wave dipole | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-HBMMP6I | 1/2 wave antenna | MMCX | 2.1dBi | Mobile | 0dB |
| A09-QBMP6I | 1/4 wave antenna | MMCX | 1.9 dBi | Mobile | 0dB |
| A09-QI | 1/4 wave integrated wire antenna | Integrated | 1.9 dBi | Mobile | 0dB |
| 29000187 | Helical | Integrated | -2.0 dBi | Fixed/Mobile | 0dB |
| A09-QW | Quarter-wave wire | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QRAMM | 3 "Quarter-wave | MMCX | 2.1 dBi | Fixed / Mobile | 0dB |
| A09-QSM-3 | Quarter-wave | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QSM-3H | Heavy duty quarter- | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QBMM-P6I | Quarter-wave w/ 6" | MMCX | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QHRN | Miniature Helical | Permanent | -1 dBi | Fixed / Mobile | 0dB |
| A09-QHSN | Miniature Helical | Permanent | -1 dBi | Fixed / Mobile | 0dB |
| A09-QHSM-2 | 2" Straight | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QHRSM-2 | 2" Right angle | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QHRSM-170 | 1.7" Right angle | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QRSM-380 | 3.8" Right angle | RPSMA | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QAPM-520 | 5.2" Articulated | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QSPM-3 | 3" Straight screw | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QAPM-3 | 3" Articulated screw | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-QAPM-3H | 3" Articulated screw | Permanent | 1.9 dBi | Fixed / Mobile | 0dB |
| A09-DPSM-P12F | omni directional | RPSMA | 3.0 dBi | Fixed | 0dB |
| A09-D3NF-P12F | omni directional | RPN | 3.0 dBi | Fixed | 0dB |
| A09-D3SM-P12F | omni directional w/ | RPSMA | 3.0 dBi | Fixed | 0dB |
| A09-D3PNF | omni directional | RPN | 3.0 dBi | Fixed | 0dB |
| A09-D3TM-P12F | omni directional w/ | RPTNC | 3.0 dBi | Fixed | 0dB |
| A09-D3PTM | omni directional | RPTNC | 3.0 dBi | Fixed | 0dB |
| A09-M0SM | Mag Mount | RPSMA | 0 dBi | Fixed | 0dB |

| | | | | | |
|----------|-----------|-------|---------|-------|-----|
| A09-M2SM | Mag Mount | RPSMA | 2.1 dBi | Fixed | 0dB |
| A09-M3SM | Mag Mount | RPSMA | 3.1 dBi | Fixed | 0dB |
| A09-M5SM | Mag Mount | RPSMA | 5.1 dBi | Fixed | 0dB |
| A09-M7SM | Mag Mount | RPSMA | 7.1 dBi | Fixed | 0dB |
| A09-M8SM | Mag Mount | RPSMA | 8.1 dBi | Fixed | 0dB |
| A09-M0TM | Mag Mount | RPTNC | 0 dBi | Fixed | 0dB |
| A09-M2TM | Mag Mount | RPTNC | 2.1 dBi | Fixed | 0dB |
| A09-M3TM | Mag Mount | RPTNC | 3.1 dBi | Fixed | 0dB |
| A09-M5TM | Mag Mount | RPTNC | 5.1 dBi | Fixed | 0dB |
| A09-M7TM | Mag Mount | RPTNC | 7.1 dBi | Fixed | 0dB |
| A09-M8TM | Mag Mount | RPTNC | 8.1 dBi | Fixed | 0dB |

Yagi antennas

| | | | | | |
|-----------|-----------------|-------|----------|----------------|-------|
| A09-Y6 | 2 Element Yagi | RPN | 6.1 dBi | Fixed / Mobile | 0dB |
| A09-Y7 | 3 Element Yagi | RPN | 7.1 dBi | Fixed / Mobile | 0dB |
| A09-Y8 | 4 Element Yagi | RPN | 8.1 dBi | Fixed / Mobile | 0dB |
| A09-Y9 | 4 Element Yagi | RPN | 9.1 dBi | Fixed / Mobile | 0dB |
| A09-Y10 | 5 Element Yagi | RPN | 10.1 dBi | Fixed / Mobile | 0dB |
| A09-Y11 | 6 Element Yagi | RPN | 11.1 dBi | Fixed / Mobile | 0dB |
| A09-Y12 | 7 Element Yagi | RPN | 12.1 dBi | Fixed / Mobile | 0dB |
| A09-Y13 | 9 Element Yagi | RPN | 13.1 dBi | Fixed / Mobile | 0.8dB |
| A09-Y14 | 10 Element Yagi | RPN | 14.1 dBi | Fixed / Mobile | 1.8dB |
| A09-Y14 | 12 Element Yagi | RPN | 14.1 dBi | Fixed / Mobile | 1.8dB |
| A09-Y15 | 13 Element Yagi | RPN | 15.1 dBi | Fixed / Mobile | 2.8dB |
| A09-Y15 | 15 Element Yagi | RPN | 15.1 dBi | Fixed / Mobile | 2.8dB |
| A09-Y6TM | 2 Element Yagi | RPTNC | 6.1 dBi | Fixed / Mobile | 0dB |
| A09-Y7TM | 3 Element Yagi | RPTNC | 7.1 dBi | Fixed / Mobile | 0dB |
| A09-Y8TM | 4 Element Yagi | RPTNC | 8.1 dBi | Fixed / Mobile | 0dB |
| A09-Y9TM | 4 Element Yagi | RPTNC | 9.1 dBi | Fixed / Mobile | 0dB |
| A09-Y10TM | 5 Element Yagi | RPTNC | 10.1 dBi | Fixed / Mobile | 0dB |
| A09-Y11TM | 6 Element Yagi | RPTNC | 11.1 dBi | Fixed / Mobile | 0dB |
| A09-Y12TM | 7 Element Yagi | RPTNC | 12.1 dBi | Fixed / Mobile | 0dB |
| A09-Y13TM | 9 Element Yagi | RPTNC | 13.1 dBi | Fixed / Mobile | 0.8dB |
| A09-Y14TM | 10 Element Yagi | RPTNC | 14.1 dBi | Fixed / Mobile | 1.8dB |
| A09-Y14TM | 12 Element Yagi | RPTNC | 14.1 dBi | Fixed / Mobile | 1.8dB |
| A09-Y15TM | 13 Element Yagi | RPTNC | 15.1 dBi | Fixed / Mobile | 2.8dB |
| A09-Y15TM | 15 Element Yagi | RPTNC | 15.1 dBi | Fixed / Mobile | 2.8dB |

Transmitters with Detachable Antennas

This radio transmitter (IC: 1846A-XBEEEXSC or IC: 1846A-XBPS3B) has been approved by Industry Canada to operate with the antenna types listed in the table above with the maximum permissible

gain and required antenna impedance for each antenna type indicated. Antenna types not included in this list, having a gain greater than the maximum gain indicated for that type, are strictly prohibited for use with this device.

Le présent émetteur radio (IC: 1846A-XBPS3B ou IC: 1846A-XBPS3B) a été approuvé par Industrie Canada pour fonctionner avec les types d'antenne énumérés ci-dessous et ayant un gain admissible maximal et l'impédance requise pour chaque type d'antenne. Les types d'antenne non

inclus dans cette liste, ou dont le gain est supérieur au gain maximal indiqué, sont strictement interdits pour l'exploitation de l'émetteur.

Detachable Antenna

Under Industry Canada regulations, this radio transmitter may only operate using an antenna of a type and maximum (or lesser) gain approved for the transmitter by Industry Canada. To reduce potential radio interference to other users, the antenna type and its gain should be so chosen that the equivalent isotropically radiated power (e.i.r.p.) is not more than that necessary for successful communication.

Conformément à la réglementation d'Industrie Canada, le présent émetteur radio peut fonctionner avec une antenne d'un type et d'un gain maximal (ou inférieur) approuvé pour l'émetteur par Industrie Canada. Dans le but de réduire les risques de brouillage radioélectrique à l'intention des autres utilisateurs, il faut choisir le type d'antenne et son gain de sorte que la puissance isotrope rayonnée équivalente (p.i.r.e.) ne dépasse pas l'intensité nécessaire à l'établissement d'une communication satisfaisante.

Appendix D: Additional Information

1-Year Warranty

XBee RF Modules from Digi International, Inc. (the "Product") are warranted against defects in materials and workmanship under normal use, for a period of 1-year from the date of purchase. In the event of a product failure due to materials or workmanship, Digi will repair or replace the defective product. For warranty service, return the defective product to Digi International, shipping prepaid, for prompt repair or replacement.

The foregoing sets forth the full extent of Digi International's warranties regarding the Product. Repair or replacement at Digi International's option is the exclusive remedy. THIS WARRANTY IS GIVEN IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, AND DIGI SPECIFICALLY DISCLAIMS ALL WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL DIGI, ITS SUPPLIERS OR LICENSORS BE LIABLE FOR DAMAGES IN EXCESS OF THE PURCHASE PRICE OF THE PRODUCT, FOR ANY LOSS OF USE, LOSS OF TIME, INCONVENIENCE, COMMERCIAL LOSS, LOST PROFITS OR SAVINGS, OR OTHER INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT, TO THE FULL EXTENT SUCH MAY BE DISCLAIMED BY LAW. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES. THEREFORE, THE FOREGOING EXCLUSIONS MAY NOT APPLY IN ALL CASES. This warranty provides specific legal rights. Other rights which vary from state to state may also apply.