XBee®/XBee-PRO S2C DigiMesh® 2.4

Radio Frequency (RF) Modules

User Guide

Revision history—90001506

Revision	Date	Description
D	June 2017	Modified regulatory and certification information as required by RED (Radio Equipment Directive).
E	February 2018	Added Brazil certification information.
F	May 2018	Added note on range estimation. Changed IC to ISED.
G	May 2019	Removed Brazilian certification information.
H	June 2019	Added FCC publication 996369 related information.

Trademarks and copyright

Disclaimers

Warranty

To view product warranty information, go to the following website:

www.digi.com/howtobuy/terms

Customer support

Gather support information: Before contacting Digi technical support for help, gather the following information:

- Product name and model
- Product serial number (s)
- Firmware version
- Operating system/browser (if applicable)
- Logs (from time of reported issue)
- Trace (if possible)
- Description of issue

Steps to reproduce

Contact Digi technical support: Digi offers multiple technical support plans and service packages. Contact us at +1 952.912.3444 or visit us at www.digi.com/support.

Feedback

To provide feedback on this document, email your comments to

techcomm@digi.com

Include the document title and part number (XBee S2C DigiMesh 2.4 User Guide, 90001506 D) in the subject line of your email.

Contents

Network commissioning and diagnostics

Sleep support

AT commands

XBee S2C DigiMesh 2.4 User Guide

This manual describes the operation of the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module, which consists of DigiMesh firmware loaded onto XBee S2C and XBee-PRO S2C hardware.

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module supports the unique needs of low-cost, low-power, wireless sensor networks. The devices require minimal power and provide reliable data delivery between remote devices. The devices operate within the ISM 2.4 GHz frequency band.

You can build networks of up to 32 nodes using these devices. For larger networks of up to 1,000 or more nodes, we offer technical support to assist with proper network configuration. For information on Technical Support plans and pricing, contact us at 877.912.3444 or visit us at www.digi.com/support.

Applicable firmware and hardware

This manual supports the following firmware:

- 9x0x

It supports the following hardware:

- XBee S2C

Firmware release notes

You can view the current release notes in the Firmware Explorer section of XCTU. For instructions on downloading and using XCTU, go to: digi.com/products/xbee-rf-solutions/xctu-software/xctu.

Technical specifications

The following tables provide the device's technical specifications.

Performance specifications

The following table describes the performance specifications for the devices.

Note Range figure estimates are based on free-air terrain with limited sources of interference. Actual range will vary based on transmitting power, orientation of transmitter and receiver, height of transmitting antenna, height of receiving antenna, weather conditions, interference sources in the area, and terrain between receiver and transmitter, including indoor and outdoor structures such as walls, trees, buildings, hills, and mountains.

Specification	XBee value	XBee-PRO value
Indoor / urban range	Up to 200 ft (60 m)	Up to 300 ft. (90 m)
Outdoor RF line-of-sight range	Up to 4000 ft (1200 m)	Up to 2 miles (3200 m)
Transmit power output (software selectable)	6.3 mW (8 dBm), Boost mode[1] 3.1 mW (5 dBm), Normal mode Channel 26 max power is 0.3 mW (-5 dBm)	63 mW (18 dBm)[2]
RF data rate	250,000 b/s	250,000 b/s
Maximum data throughput	TBD	TBD
UART interface data rate	1200 b/s to 250,000 b/s	1200 b/s to 250,000 b/s
SPI data rate	Up to 5 Mb/s (burst)	Up to 5 Mb/s (burst)
Receiver sensitivity	-102 dBm, Boost mode -100 dBm, Normal mode	-101 dBm

Power requirements

The following table describes the power requirements for the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module.

Specification	XBee	XBee-PRO
Supply voltage	2.1 - 3.6 V	2.7 - 3.6 V
Transmit current (typical, VCC = 3.3 V)	45 mA (8 dBm, Boost mode) 33 mA (5 dBm, Normal mode)	120 mA (18 dBm)
Idle / receive current (typical, VCC = 3.3 V)	31 mA (Boost mode) 28 mA (Normal mode)	31 mA
Power-down current	<1 uA @ 25C	<1 uA @ 25C

[1]Boost mode enabled by default; see PM (Power Mode).

[2]See Regulatory information for region-specific certification requirements.

General specifications

The following table describes the general specifications for the devices.

Specification	XBee	XBee-PRO
Operating frequency	ISM 2.4 GHz	
Supported channels	11 - 26	12 - 23
Form factor	TH: 2.438 x 2.761 cm (0.960 x 1.087 in) SMT: 2.199 x 3.4 x 0.305 cm (0.866 x 1.33 x 0.120 in)	TH: 2.438 x 3.294 cm (0.960 x 1.297 in) SMT: 2.199 x 3.4 x 0.305 cm (0.866 x 1.33 x 0.120 in)
Operating temperature	-40 to 85 ºC (industrial)	
Antenna options	TH: PCB antenna, U.FL connector, RPSMA connector, or integrated wire SMT: RF pad, PCB antenna, or U.FL connector	

Networking and security specifications

The following table describes the networking and security specifications for the devices.

Specification	XBee	XBee-PRO
Supported network topologies	Mesh, point-to-point, point-to-multipoint, peer-to-peer	Mesh, point-to-point, point-to-multipoint, peer-to-peer
Number of channels (software selectable)	16 direct sequence channels	12 direct sequence channels
Addressing options	PAN ID, channel and 64-bit addresses	PAN ID, channel and 64-bit addresses
Encryption	256-bit Advanced Encryption Standard (AES) Counter Mode or 128-bit AES Codebook (for legacy support).	256-bit AES Counter Mode or 128-bit AES Codebook (for legacy support).

Regulatory conformity summary

This table describes the agency approvals for the devices.

Country	XBee (surface-mount)	XBee-PRO (surface-mount)	XBee (through-hole)	XBee-PRO (through-hole)
United States (FCC Part 15.247)	FCC ID: MCQ-XBS2C	FCC ID: MCQ-PS2CSM	FCC ID: MCQ-S2CTH	FCC ID: MCQ-PS2CTH
Innovation, Science and Economic Development Canada (ISED)	IC: 1846A-XBS2C	IC: 1846A-PS2CSM	IC: 1846A-S2CTH	IC: 1846A-PS2CTH
FCC/IC test transmit power output range	-26 to +8 dBm	-0.7 to +19.4 dBm	-26 to +8 dBm	+1 to +19 dBm
Europe (CE)	Yes	-	Yes	-
Australia	RCM	RCM	RCM	RCM
Japan	R201WW10215369		R210- 105563	
South Korea	MSIP-CRM-DIG-XBee-S2C		MSIP-CRM-DIG-XBee-S2C-TH	
RoHS	Compliant			

Serial communication specifications

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module supports both Universal Asynchronous Receiver / Transmitter (UART) and Serial Peripheral Interface (SPI) serial connections.

UART pin assignments

The SC1 (Serial Communication Port 1) of the Ember 357 is connected to the UART port. The following table provides the UART pin assignments.

Specifications	Module pin number	
UART pins	XBee (surface-mount)	XBee (through-hole)
DOUT	3	2
DIN / $\overline{\text{CONFIG}}$	4	3
$\overline{\text{CTS}}$ / DIO7	25	12
$\overline{\text{RTS}}$ / DIO6	29	16

SPI pin assignments

The SC2 (Serial Communication Port 2) of the Ember 357 is connected to the SPI port.

Specifications	**Module pin number**	
SPI pins	**XBee (surface-mount)**	**XBee (through-hole)**
SPI_SCLK	14	18
SPI_$\overline{\text{SSEL}}$	15	17
SPI_MOSI	16	11
SPI_MISO	17	4
SPI_$\overline{\text{ATTN}}$	12	19

GPIO specifications

XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules have 15 General Purpose Input / Output (GPIO) ports available. The exact list depends on the device configuration, as some GPIO pads are used for purposes such as serial communication.

GPIO Electrical Specification	Value
Low Schmitt switching threshold	0.42 - 0.5 x VCC
High Schmitt switching threshold	0.62 - 0.8 x VCC
Input current for logic 0	-0.5 μA
Input current for logic 1	0.5 μA
Input pull-up resistor value	29 kΩ
Input pull-down resistor value	29 kΩ
Output voltage for logic 0	0.18 x VCC (maximum)
Output voltage for logic 1	0.82 x VCC (minimum)
Output source/sink current for pad numbers 3, 4, 5, 10, 12, 14, 15, 16, 17, 25, 26, 28, 29, 30, and 32 on the SMT modules	4 mA
Output source/sink current for pin numbers 2, 3, 4, 9, 12, 13, 15, 16, 17, and 19 on the TH modules	4 mA
Output source/sink current for pad numbers 7, 8, 24, 31, and 33 on the SMT modules	8 mA
Output source/sink current for pin numbers 6, 7, 11, 18, and 20 on the TH modules	8 mA
Total output current (for GPIO pads)	40 mA

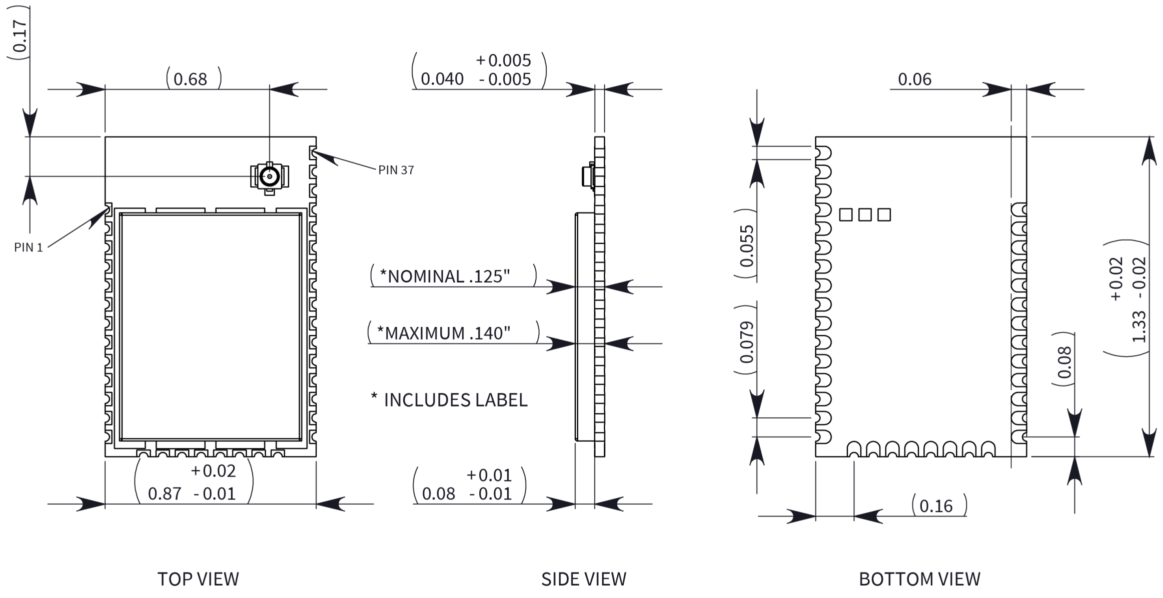
Hardware

Antenna options

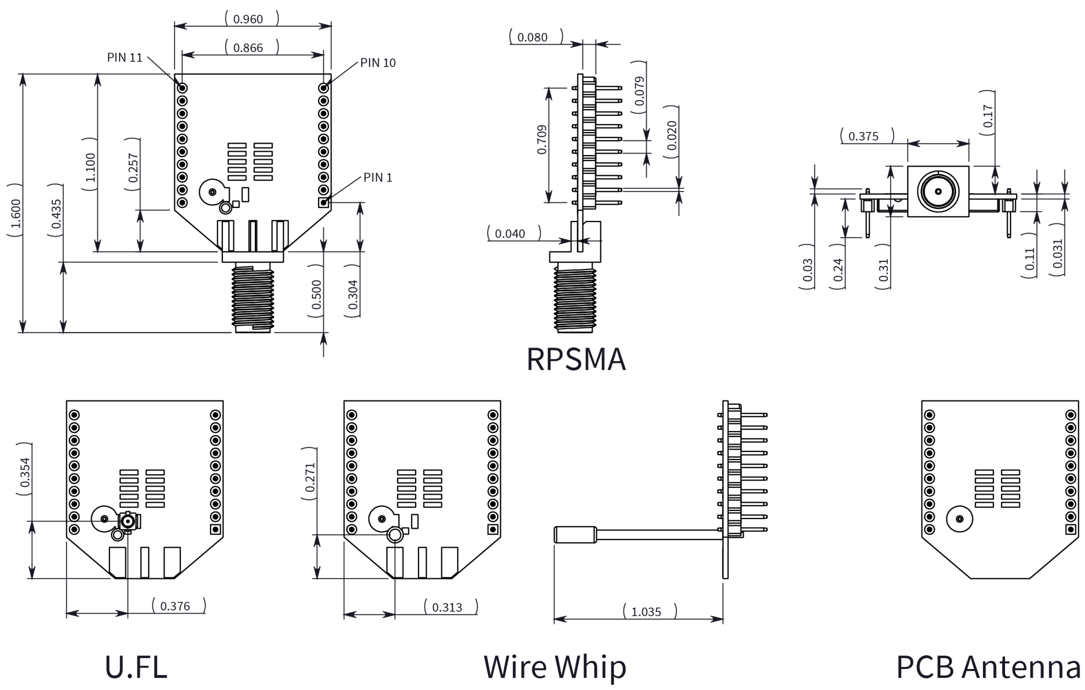
The ranges specified are typical for the integrated whip (1.5 dBi) and dipole (2.1 dBi) antennas. The printed circuit board (PCB) antenna option provides advantages in its form factor; however, it typically yields shorter range than the whip and dipole antenna options when transmitting outdoors. For more information, see XBee and XBee-PRO OEM RF Module Antenna Considerations Application Note.

Mechanical drawings

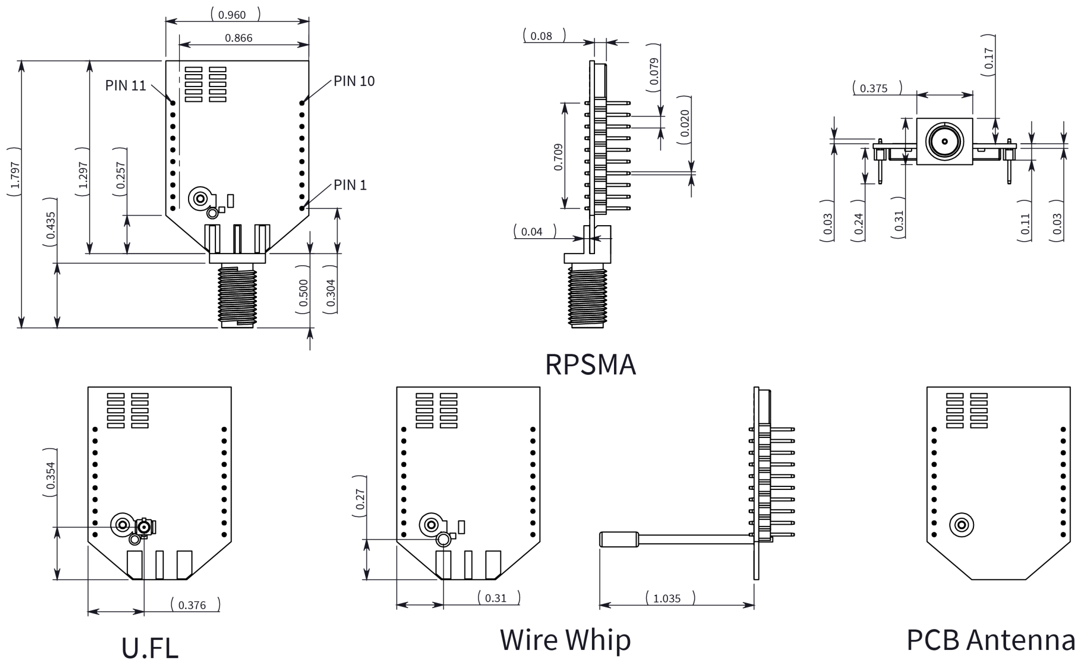
The following mechanical drawings of the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module show all dimensions in inches. The first drawing shows the surface-mount device (antenna options not shown).



The following drawings show the standard (non-PRO) through-hole device.



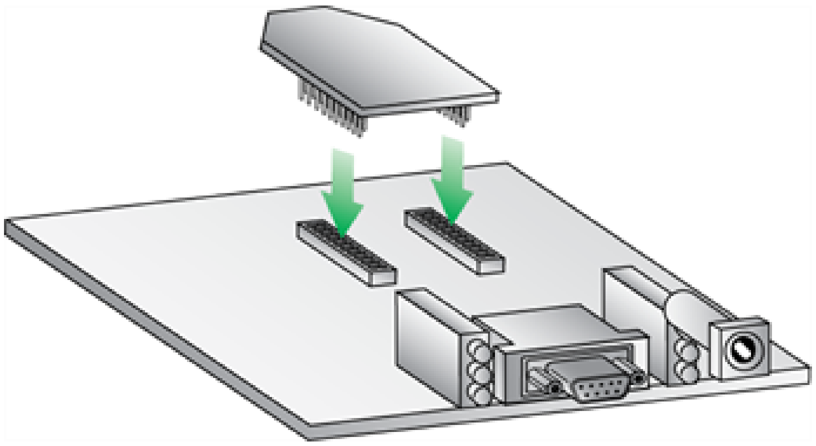
The following drawings show the XBee-PRO through-hole device.



Mounting considerations

We design the through-hole module to mount into a receptacle so that you do not have to solder the module when you mount it to a board. The development kits may contain RS-232 and USB interface boards that use two 20-pin receptacles to receive modules.

The following illustration shows the module mounting into the receptacle on the RS-232 interface board.

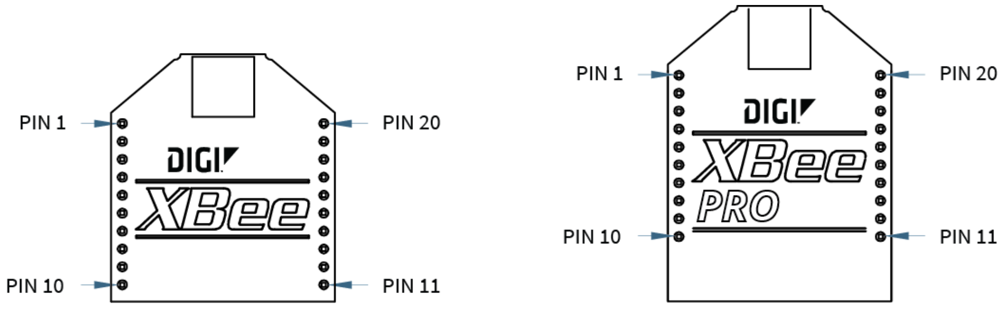


- Through-hole single-row receptacles: Samtec part number: MMS-110-01-L-SV (or equivalent)
- Surface-mount double-row receptacles: Century Interconnect part number: CPRMSL20-D-0-1 (or equivalent)
- Surface-mount single-row receptacles: Samtec part number: SMM-110-02-SM-S

Note We recommend that you print an outline of the module on the board to indicate the correct orientation for mounting the module.

Pin signals

The following image shows the pin numbers; it shows the device's top sides, the shields are on the bottom.



The following table shows the pin assignments for the through-hole device. In the table, low-asserted signals have a horizontal line above signal name.

Pin	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART data out
3	DIN/$\overline{\text{CONFIG}}$	Input	UART data In

Pin	Name	Direction	Description
4	DIO12/SPI_MISO	Both	Digital I/O 12 / Serial Peripheral Interface (SPI) Data Out
5	$\overline{\text{RESET}}$	Input	Module reset (reset pulse must be at least 200 ns). This must be driven as an open drain/collector. The device drives this line low when a reset occurs. Never drive this line high.
6	DIO10/PWM0/RSSI PWM	Both	Digital I/O 10 / PWM output 0 / RX signal strength indicator
7	DIO11/PWM1	Both	Digital I/O 11 / PWM output 1
8	[Reserved]	-	Do not connect
9	DIO8/SLEEP_ RQ/$\overline{\text{DTR}}$	Both	Digital I/O 8 / Pin sleep control line
10	GND	-	Ground
11	DIO4/SPI_MOSI	Both	Digital I/O 4 / SPI Data In
12	DIO7/$\overline{\text{CTS}}$	Both	Digital I/O 7 / Clear-to-send flow control
13	ON/$\overline{\text{SLEEP}}$	Output	Device sleep status indicator
14	V_{REF}	-	Feature not supported on this device. Used on other XBee devices for analog voltage reference.
15	DIO5/ASSOC	Both	Digital I/O 5 / Associated indicator
16	DIO6/$\overline{\text{RTS}}$	Both	Digital I/O 6 / Request-to-send flow control
17	DIO3/AD3/SPI_ $\overline{\text{SSEL}}$	Both	Digital I/O 3 / Analog input 3 / SPI select
18	DIO2/AD2/SPI_CLK	Both	Digital I/O 2 / Analog input 2 / SPI clock
19	DIO1/AD1/SPI_ $\overline{\text{ATTN}}$	Both	Digital I/O 1 / Analog input 1 / SPI Attention
20	DIO0/AD0	Both	Digital I/O 0 / Analog input 0

The following table shows the pin assignments for the surface-mount device.

Pin	Name	Direction	Function
1	GND	-	Ground
2	VCC	-	Power supply
3	DOUT	Output	UART data out
4	DIN/$\overline{\text{CONFIG}}$	Input	UART data in
5	DIO12	Both	Digital I/O 12

Pin	Name	Direction	Function
6	$\overline{\text{RESET}}$	Input	Module reset (reset pulse must be at least 200 ns). This must be driven as an open drain/collector. The device drives this line low when a reset occurs. Never drive this line high.
7	DIO10/PWM0/RSSI PWM	Both	Digital I/O 10 / PWM output 0 / RX signal strength indicator
8	DIO11/PWM1	Both	Digital I/O 11 / PWM output 1
9	[Reserved]	-	Do not connect
10	DIO8/SLEEP_ RQ/DTR	Both	Digital I/O 8 / Pin sleep control line
11	GND	-	Ground
12	SPI_ ATTN/$\overline{\text{BOOTMODE}}$	Output	SPI Attention. Do not tie low on reset.
13	GND	-	Ground
14	SPI_CLK	Input	SPI clock
15	SPI_$\overline{\text{SSEL}}$	Input	SPI select
16	SPI_MOSI	Input	SPI Data In
17	SPI_MISO	Output	SPI Data Out
18	[Reserved]	-	Do not connect
19	[Reserved]	-	Do not connect
20	[Reserved]	-	Do not connect
21	[Reserved]	-	Do not connect
22	GND	-	Ground
23	[Reserved]	-	Do not connect
24	DIO4	Both	Digital I/O 4
25	DIO7/$\overline{\text{CTS}}$	Both	Digital I/O 7 / Clear-to-send flow control
26	ON/$\overline{\text{SLEEP}}$	Output	Device sleep status indicator
27	V_{REF}	-	Feature not supported on this device. Used on other XBee devices for analog voltage reference.
28	DIO5/ASSOC	Both	Digital I/O 5 / Associated indicator
29	DIO6/$\overline{\text{RTS}}$	Both	Digital I/O 6 / Request-to-send flow control
30	DIO3/AD3	Both	Digital I/O 3 / Analog input 3

Pin	Name	Direction	Function
31	DIO2/AD2	Both	Digital I/O 2 / Analog input 2
32	DIO1/AD1	Both	Digital I/O 1 / Analog input 1
33	DIO0/AD0	Both	Digital I/O 0 / Analog input 0
34	[Reserved]	-	Do not connect
35	GND	-	Ground
36	RF	Both	RF connection
37	[Reserved]	-	Do not connect

Notes

Minimum connections: VCC, GND, DOUT and DIN.

Minimum connections for updating firmware: VCC, GND, DIN, DOUT, RTS and DTR.

The table specifies signal direction with respect to the device.

Use the **PR** (Pull-up/Down Resistor Enable) command to configure several of the input pull-ups.

You can connect other pins to external circuitry for convenience of operation including the Associate LED pin (pin 15). The Associate LED flashes differently depending on the state of the device.

Leave any unused pins disconnected.

Design notes

The following guidelines help to ensure a robust design.

Power supply design

A poor power supply can lead to poor device performance, especially if you do not keep the supply voltage within tolerance or if it is excessively noisy. To help reduce noise, place a 1.0 µF and 8.2 pF capacitor as near as possible to pin 1 on the PCB. If you are using a switching regulator for the power supply, switch the frequencies above 500 kHz. Limit the power supply ripple to a maximum 100 mV peak to peak.

Board layout

We design XBee devices to be self sufficient and have minimal sensitivity to nearby processors, crystals or other printed circuit board (PCB) components. Keep power and ground traces thicker than signal traces and make sure that they are able to comfortably support the maximum current specifications. There are no other special PCB design considerations to integrate XBee devices, with the exception of antennas.

Antenna performance

Antenna location is important for optimal performance. The following suggestions help you achieve optimal antenna performance. Point the antenna up vertically (upright). Antennas radiate and receive the best signal perpendicular to the direction they point, so a vertical antenna's omnidirectional radiation pattern is strongest across the horizon.

Position the antennas away from metal objects whenever possible. Metal objects between the transmitter and receiver can block the radiation path or reduce the transmission distance. Objects that are often overlooked include:

- metal poles
- metal studs
- structure beams
- concrete, which is usually reinforced with metal rods

If you place the device inside a metal enclosure, use an external antenna. Common objects that have metal enclosures include:

- vehicles
- elevators
- ventilation ducts
- refrigerators
- microwave ovens
- batteries
- tall electrolytic capacitors

Do not place XBee devices with the chip or integrated PCB antenna inside a metal enclosure.

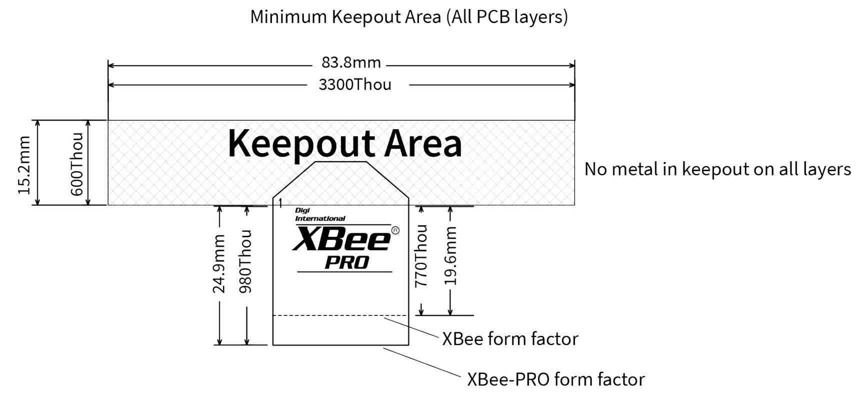
Do not place any ground planes or metal objects above or below the antenna.

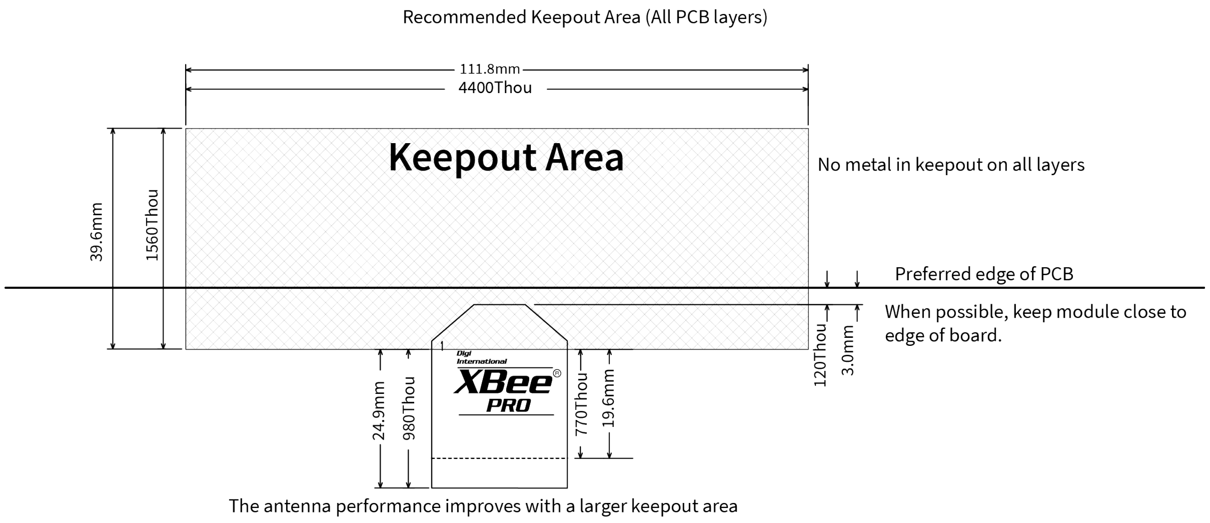
For the best results, mount the device at the edge of the host PCB. Ensure that the ground, power, and signal planes are vacant immediately below the antenna section.

Keepout area

We recommend that you allow a “keepout” area, which the following drawings show.

Through-hole keepout



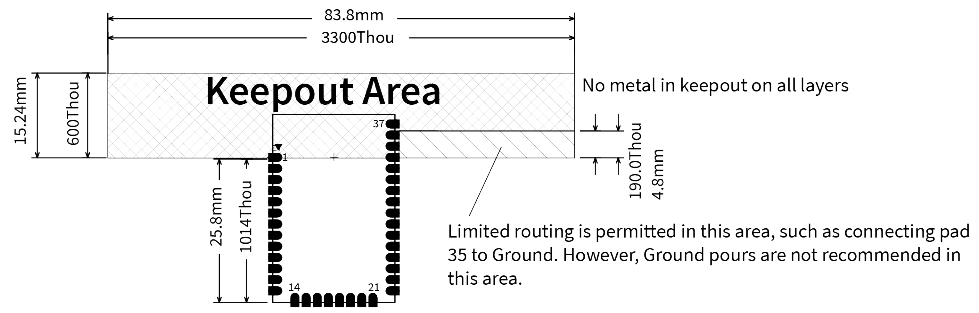

Recommended Keepout Area (All PCB layers)
111.8mm
4400Thou
39.6mm
1560Thou
Keepout Area
No metal in keepout on all layers
Preferred edge of PCB
When possible, keep module close to edge of board.
120Thou
3.0mm
24.9mm
980Thou
770Thou
19.6mm
Digi International
XBee PRO
The antenna performance improves with a larger keepout area

Notes

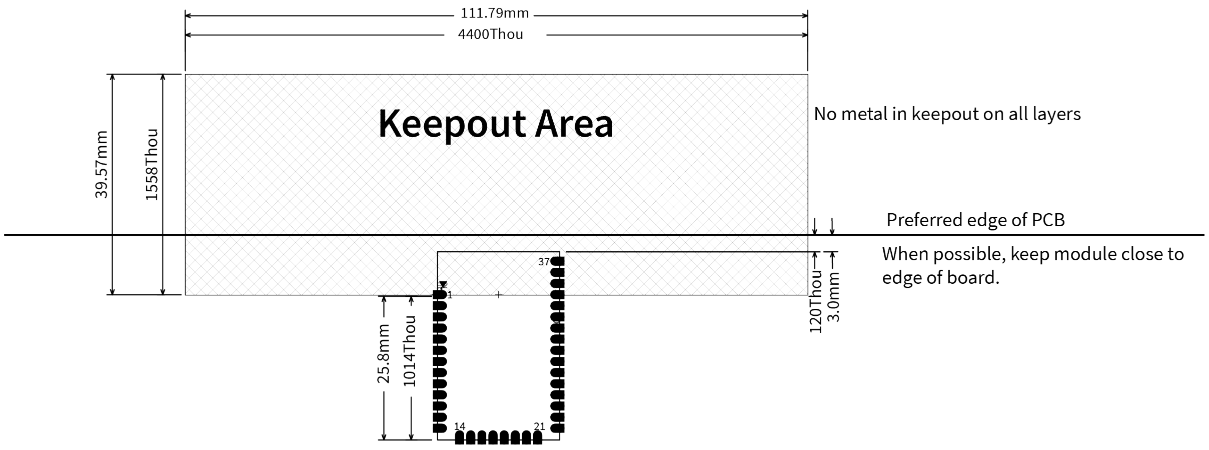
1. We recommend non-metal enclosures. For metal enclosures, use an external antenna.
2. Keep metal chassis or mounting structures in the keepout area at least 2.54 cm (1 in) from the antenna.
3. Maximize the distance between the antenna and metal objects that might be mounted in the keepout area.
4. These keepout area guidelines do not apply for wire whip antennas or external RF connectors. Wire whip antennas radiate best over the center of a ground plane.

Surface-mount keepout

Minimum Keepout Area for PCB Antenna (All layers)


83.8mm
3300Thou
Keepout Area
No metal in keepout on all layers
15.24mm
600Thou
37
1
14
21
25.8mm
1014Thou
190.0Thou
4.8mm
Limited routing is permitted in this area, such as connecting pad 35 to Ground. However, Ground pours are not recommended in this area.

Recommended Keepout Area for PCB Antenna (All layers)



The antenna performance improves with a larger keepout area

Notes

1. We recommend non-metal enclosures. For metal enclosures, use an external antenna.
2. Keep metal chassis or mounting structures in the keepout area at least 2.54 cm (1 in) from the antenna.
3. Maximize the distance between the antenna and metal objects that might be mounted in the keepout area.
4. These keepout area guidelines do not apply for wire whip antennas or external RF connectors. Wire whip antennas radiate best over the center of a ground plane.

RF pad version

The RF pad is a soldered antenna connection on the surface-mount device. The RF signal travels from pin 36 on the module to the antenna through a single ended RF transmission line on the PCB. This line should have a controlled impedance of 50 Ω.

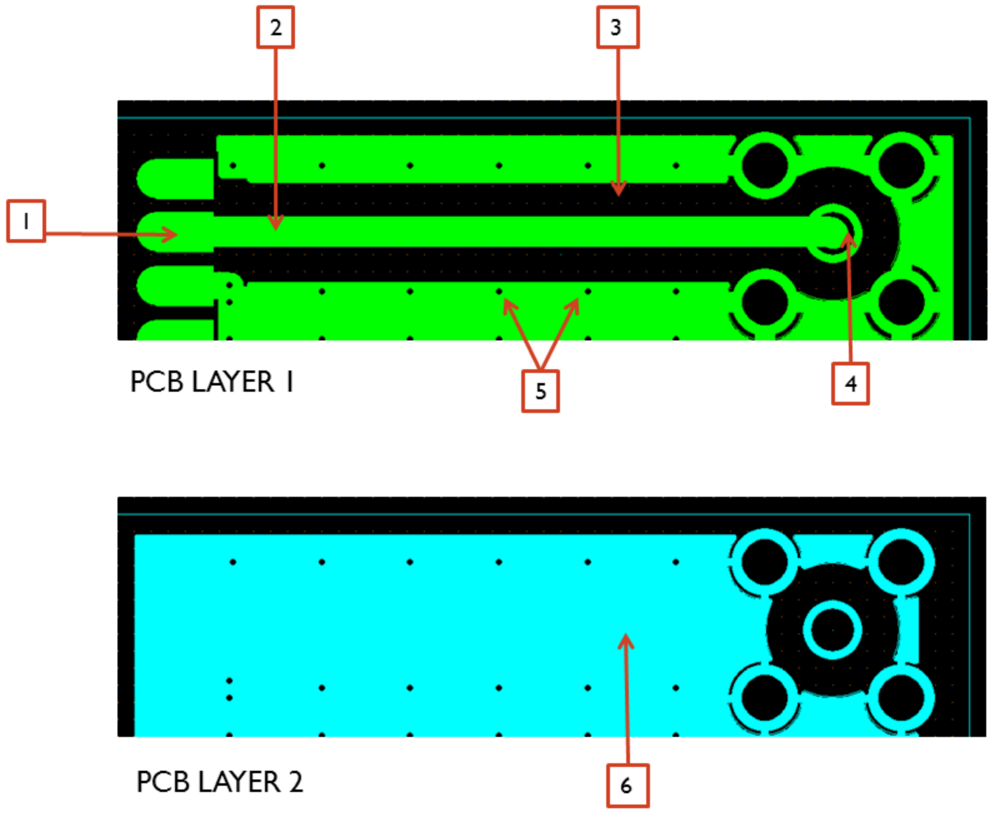
For the transmission line, we recommend either a microstrip or coplanar waveguide trace on the PCB. We provide a microstrip example below, because it is simpler to design and generally requires less area on the host PCB than coplanar waveguide.

We do not recommend using a stripline RF trace because that requires routing the RF trace to an inner PCB layer, and via transitions can introduce matching and performance problems.

The following figure shows a layout example of a microstrip connecting an RF pad module to a through-hole RPSMA RF connector.

- The top two layers of the PCB have a controlled thickness dielectric material in between. The second layer has a ground plane which runs underneath the entire RF pad area. This ground plane is a distance *d*, the thickness of the dielectric, below the top layer.
- The top layer has an RF trace running from pin 36 of the device to the RF pin of the RPSMA connector. The RF trace's width determines the impedance of the transmission line with relation to the ground plane. Many online tools can estimate this value, although you should consult the PCB manufacturer for the exact width. Assuming *d* = 0.025 in, and that the dielectric has a relative permittivity of 4.4, the width in this example will be approximately 0.045 in for a 50 Ω trace. This trace width is a good fit with the module footprint's 0.060 in pad width.

We do not recommend using a trace wider than the pad width, and using a very narrow trace can cause unwanted RF loss. You can minimize the length of the trace by placing the RPSMA jack close to the module. All of the grounds on the jack and the module are connected to the ground planes directly or through closely placed vias. Space any ground fill on the top layer at least twice the distance *d* (in this case, at least 0.050 in) from the microstrip to minimize their interaction.



Number	Description
1	XBee surface-mount pin 36
2	50 Ω microstrip trace
3	Back off ground fill at least twice the distance between layers 1 and 2
4	RF connector
5	Stitch vias near the edges of the ground plane
6	Pour a solid ground plane under the RF trace on the reference layer

Implementing these design suggestions helps ensure that the RF pad device performs to specifications.

ADC characteristics

The following table displays the ADC timing and performance characteristics.

Parameter	Condition	Min	Typical	Max	Units
Internal voltage reference		1.17	1.2	1.23	V
Analog input voltage range[1]		0	-	1.2	V
Input impedance		1	-	-	MΩ
Number of bits			10		
Differential non-linearity	Codes peak Codes RMS	-	0.044 0.014	-	LSB
Integral non-linearity	Codes peak Codes RMS	-	0.306 0.176	-	LSB

[1]Analog input must be within range for valid conversion. Values greater than 1.2 V convert to $3FF.

Configure the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module

Software libraries

One way to communicate with the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module is by using a software library. The libraries available for use with the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module include:

- XBee Java library
- XBee Python library
- XBee ANSI C library
- XBee mbed library

The XBee Java Library is a Java API. The package includes the XBee library, its source code and a collection of samples that help you develop Java applications to communicate with your XBee devices.

The XBee Python Library is a Python API that dramatically reduces the time to market of XBee projects developed in Python and facilitates the development of these types of applications, making it an easy process.

The XBee ANSI C Library project is a collection of portable ANSI C code for communicating with the devices in API mode.

The XBee mbed library is a ready-to-import mbed extension that dramatically reduces development time for XBee projects on mbed platforms.

Configure the device using XCTU

XBee Configuration and Test Utility (XCTU) is a multi-platform program that enables users to interact with Digi radio frequency (RF) devices through a graphical interface. The application includes built-in tools that make it easy to set up, configure, and test Digi RF devices.

For full support of the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module, you must use XCTU version 6.3.2 or higher.

For instructions on downloading and using XCTU, see the XCTU User Guide.

Click **Discover devices** and follow the instructions. XCTU should discover the connected XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules using the provided settings.

Click **Add selected devices**.The devices appear in the **Radio Modules** list. You can click a module to view and configure its individual settings. For more information on these items, see AT commands.

Over-the-air (OTA) firmware update

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module supports OTA firmware updates using XCTU version 6.3.2 or higher. For instructions on performing an OTA firmware update with XCTU, see How to update the firmware of your modules in the XCTU User Guide.

XBee Network Assistant

The XBee Network Assistant is an application designed to inspect and manage RF networks created by Digi XBee devices. Features include:

- Join and inspect any nearby XBee network to get detailed information about all the nodes it contains.
- Update the configuration of all the nodes of the network, specific groups, or single devices based on configuration profiles.

- Geo-locate your network devices or place them in custom maps and get information about the connections between them.
- Export the network you are inspecting and import it later to continue working or work offline.
- Use automatic application updates to keep you up to date with the latest version of the tool.

See the *XBee Network Assistant User Guide* for more information.

To install the XBee Network Assistant:

1. Navigate to digi.com/xbeenetworkassistant.
2. Click **General Diagnostics, Utilities and MIBs**.
3. Click the **XBee Network Assistant - Windows x86** link.
4. When the file finishes downloading, run the executable file and follow the steps in the XBee Network Assistant Setup Wizard.

XBee Multi Programmer

The XBee Multi Programmer is a combination of hardware and software that enables partners and distributors to program multiple Digi Radio frequency (RF) devices simultaneously. It provides a fast and easy way to prepare devices for distribution or large networks deployment.

The XBee Multi Programmer board is an enclosed hardware component that allows you to program up to six RF modules thanks to its six external XBee sockets. The XBee Multi Programmer application communicates with the boards and allows you to set up and execute programming sessions. Some of the features include:

- Each XBee Multi Programmer board allows you to program up to six devices simultaneously. Connect more boards to increase the programming concurrency.
- Different board variants cover all the XBee form factors to program almost any Digi RF device.

Download the XBee Multi Programmer application from: digi.com/support/productdetail?pid=5641

See the *XBee Multi Programmer User Guide* for more information.

Modes

Serial modes

The firmware operates in several different modes. Two top-level modes establish how the device communicates with other devices through its serial interface: Transparent operating mode and API operating mode. Use the **AP** command to choose Serial mode. XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules use Transparent operation as the default serial mode.

The following modes describe how the serial port sends and receives data.

Transparent operating mode

Devices operate in this mode by default. The device acts as a serial line replacement when it is in Transparent operating mode. The device queues all UART data it receives through the DIN pin for RF transmission. When a device receives RF data, it sends the data out through the DOUT pin. You can set the configuration parameters using Command mode.

Note Transparent operating mode is not available when using the SPI interface; see SPI port.

The device buffers data in the serial receive buffer until one of the following causes the data to be packetized and transmitted:

- The device receives no serial characters for the amount of time determined by the **RO** (Packetization Timeout) parameter. If **RO** = 0, packetization begins when a character is received.
- The device receives the Command Mode Sequence (**GT** + **CC** + **GT**). Any character buffered in the serial receive buffer before the sequence is transmitted.
- The device receives the maximum number of characters that fits in an RF packet (100 bytes). See NP (Maximum Packet Payload Bytes).

API operating mode

Application programming interface (API) operating mode is an alternative to Transparent mode. It is helpful in managing larger networks and is more appropriate for performing tasks such as collecting data from multiple locations or controlling multiple devices remotely. API mode is a frame-based protocol that allows you to direct data on a packet basis. It can be particularly useful in large networks where you need control over the operation of the radio network or when you need to know which node a data packet is from. The device communicates UART or SPI data in packets, also known as API frames. This mode allows for structured communications with serial devices.

For more information, see API mode overview.

Command mode

Command mode is a state in which the firmware interprets incoming characters as commands. It allows you to modify the device’s configuration using parameters you can set using AT commands. When you want to read or set any parameter of the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module using this mode, you have to send an AT command. Every AT command starts with the letters **AT** followed by the two characters that identify the command and then by some optional configuration values.

The operating modes of the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module are controlled by the AP (API Enable) setting, but Command mode is always available as a mode the device can enter while configured for any of the operating modes.

Command mode is available on the UART interface for all operating modes. You cannot use the SPI interface to enter Command mode.

Enter Command mode

To get a device to switch into Command mode, you must issue the following sequence: **+++** within one second. There must be at least one second preceding and following the **+++** sequence. Both the command character (**CC**) and the silence before and after the sequence (**GT**) are configurable. When the entrance criteria are met the device responds with **OK\r** on UART signifying that it has entered Command mode successfully and is ready to start processing AT commands.

If configured to operate in Transparent operating mode, when entering Command mode the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module knows to stop sending data and start accepting commands locally.

Note Do not press **Return** or **Enter** after typing **+++** because it interrupts the guard time silence and prevents you from entering Command mode.

When the device is in Command mode, it listens for user input and is able to receive AT commands on the UART. If **CT** time (default is 10 seconds) passes without any user input, the device drops out of Command mode and returns to the previous operating mode. You can force the device to leave Command mode by sending CN (Exit Command Mode).

You can customize the command character, the guard times and the timeout in the device’s configuration settings. For more information, see CC (Command Character), CT (Command Mode Timeout) and GT (Guard Times).

Troubleshooting

Failure to enter Command mode is often due to baud rate mismatch. Ensure that the baud rate of the connection matches the baud rate of the device. By default, BD (Baud Rate) = **3** (9600 b/s).

There are two alternative ways to enter Command mode:

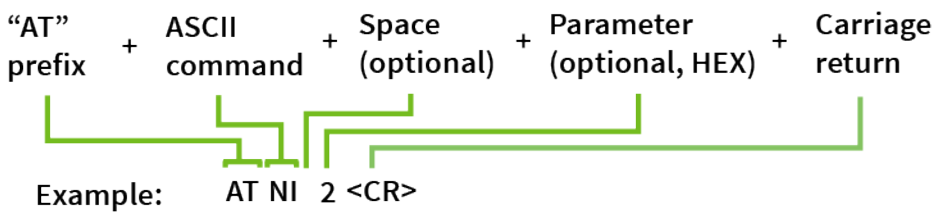
- A serial break for six seconds enters Command mode. You can issue the "break" command from a serial console, it is often a button or menu item.
- Asserting DIN (serial break) upon power up or reset enters Command mode. XCTU guides you through a reset and automatically issues the break when needed.

Both of these methods temporarily set the device's baud rate to 9600 and return an **OK** on the UART to indicate that Command mode is active. When Command mode exits, the device returns to normal operation at the baud rate that **BD** is set to.

Send AT commands

Once the device enters Command mode, use the syntax in the following figure to send AT commands. Every AT command starts with the letters **AT**, which stands for "attention." The AT is followed by two characters that indicate which command is being issued, then by some optional configuration values.

To read a parameter value stored in the device’s register, omit the parameter field.



The preceding example changes NI (Node Identifier) to **My XBee**.

Multiple AT commands

You can send multiple AT commands at a time when they are separated by a comma in Command mode; for example, **ATNIMy XBee,AC<cr>**.

The preceding example changes the **NI (Node Identifier)** to **My XBee** and makes the setting active through AC (Apply Changes).

Parameter format

Refer to the list of AT commands for the format of individual AT command parameters. Valid formats for hexidecimal values include with or without a leading **0x** for example **FFFF** or **0xFFFF**.

Response to AT commands

When using AT commands to set parameters the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module responds with **OK<cr>** if successful and **ERROR<cr>** if not.

For devices with a file system:

ATAP1<cr>

OK<cr>

When reading parameters, the device returns the current parameter value instead of an **OK** message.

ATAP<cr>

1<cr>

Apply command changes

Any changes you make to the configuration command registers using AT commands do not take effect until you apply the changes. For example, if you send the **BD** command to change the baud rate, the actual baud rate does not change until you apply the changes. To apply changes:

1. Send AC (Apply Changes).
2. Send WR (Write).
   or:
3. Exit Command mode.

Make command changes permanent

Send a WR (Write) command to save the changes. **WR** writes parameter values to non-volatile memory so that parameter modifications persist through subsequent resets.

Send as RE (Restore Defaults) to wipe settings saved using **WR** back to their factory defaults.

Note You still have to use **WR** to save the changes enacted with **RE**.

Exit Command mode

1. Send CN (Exit Command Mode) followed by a carriage return.
   or:
2. If the device does not receive any valid AT commands within the time specified by CT (Command Mode Timeout), it returns to Transparent or API mode. The default Command mode timeout is 10 seconds.

For an example of programming the device using AT Commands and descriptions of each configurable parameter, see AT commands.

Transceiver modes

The following modes describe how the transceiver sends and receives over-the-air (OTA) data.

Idle mode

When not receiving or transmitting data, the device is in Idle mode. During Idle mode, the device listens for valid data on both the RF and serial ports.

Transmit mode

Transmit mode is the mode in which the device is transmitting data. This typically happens when data is received from the serial port.

Receive mode

This is the default mode for the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module. The device is in Receive mode when it is not transmitting data. If a destination node receives a valid RF packet, the destination node transfers the data to its serial transmit buffer.

Serial communication

Select a serial port

The device has two serial ports and only one is active at a time. To be active, a port must be enabled and in use.

The UART is always enabled. The SPI is enabled if it is configured. To be configured, SPI_MISO, SPI_MOSI, SPI_SSEL , and SPI_CLK must all be configured as peripherals. On the surface-mount device, these lines are configured as peripherals by setting P5, P6, P7, and P8 to 1. This is also the default configuration for surface-mount devices.

On the through-hole device, those pins are not available and SPI is disabled by default. Therefore, to configure the SPI pins on a through-hole device, hold DOUT low during a reset. If the UART is not hooked up, then DOUT can be treated as an input to force the device into SPI mode. It is best to follow this special operation by a **WR** operation so that the SPI port will still be enabled on future resets without forcing DOUT low.

Once the SPI port is enabled by either means, it is still not active until the external SPI master asserts SPI_SSEL low. After the SPI port is active, the device continues to use the SPI port until the next reset.

Serial receive buffer

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

Serial transmit buffer

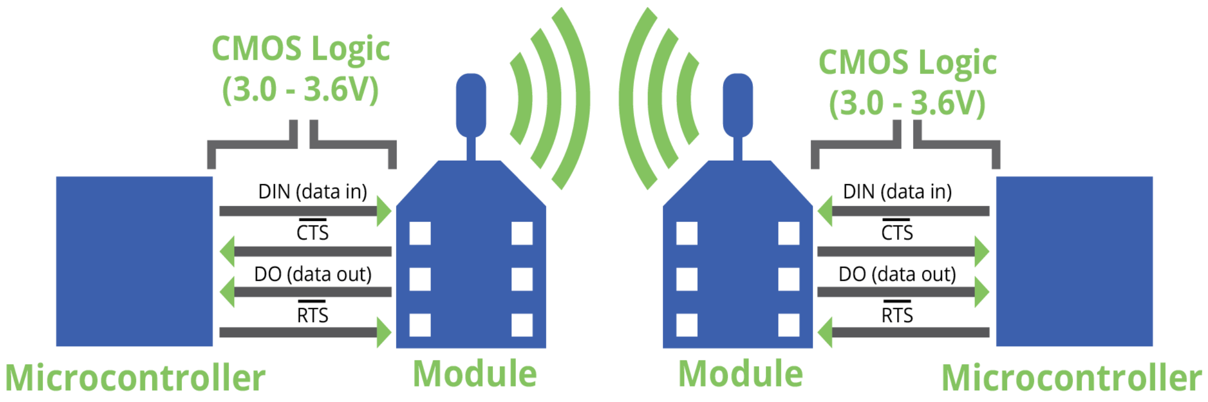
When the device receives RF data, it moves the data into the serial transmit buffer and sends it out the UART or SPI port. If the serial transmit buffer becomes full and the system buffers are also full, then it drops the entire RF data packet. Whenever the device receives data faster than it can process and transmit the data out the serial port, there is a potential of dropping data.

UART port

UART data flow

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module device’s UART performs tasks such as checking timing and parity, which is required for data communications.

Devices that have a UART interface connect directly to the pins of the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module as shown in the following figure. The figure shows system data flow in a UART-interfaced environment. Low-asserted signals have a horizontal line over the signal name.

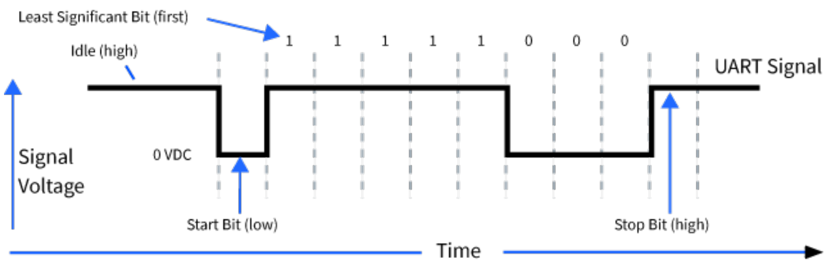


Serial data

A device sends data to the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module's UART through TH pin 3/SMT pin 4 DIN as an asynchronous serial signal. When the device is not transmitting data, the signals should idle high.

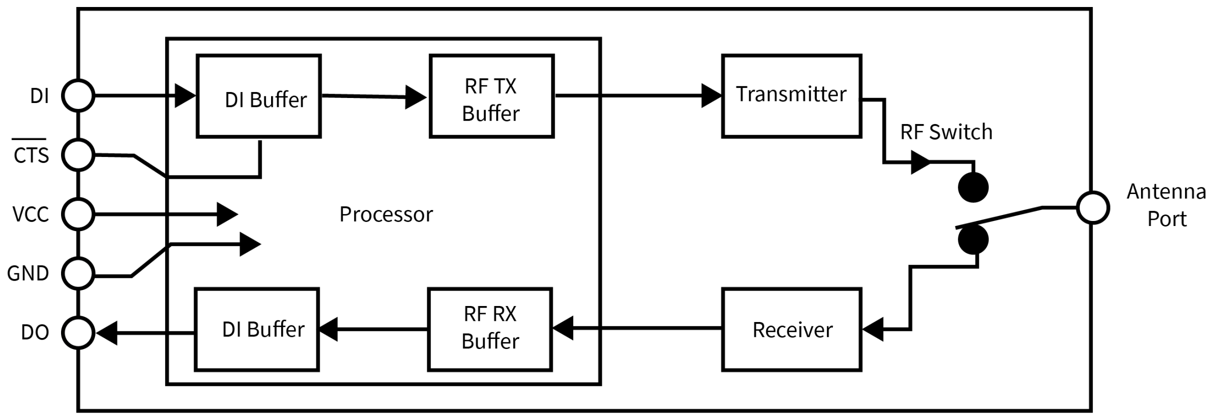
For serial communication to occur, you must configure the UART of both devices (the microcontroller and the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module) with compatible settings for the baud rate, parity, start bits, stop bits, and data bits.

Each data byte consists of a start bit (low), 8 data bits (least significant bit first) and a stop bit (high). The following diagram illustrates the serial bit pattern of data passing through the device. The diagram shows UART data packet 0x1F (decimal number 31) as transmitted through the device.



Flow control

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module maintains buffers to collect serial and RF data that it receives. The serial receive buffer collects incoming serial characters and holds them until the device can process them. The serial transmit buffer collects the data it receives via the RF link until it transmits that data out the serial port. The following figure shows the process of device buffers collecting received serial data.



$\overline{\text{CTS}}$ flow control

If you enable $\overline{\text{CTS}}$ flow control (by setting **D7** to 1), when the serial receive buffer is 7 bytes away from being full, the device de-asserts $\overline{\text{CTS}}$(sets it high) to signal to the host device to stop sending serial data. The device reasserts $\overline{\text{CTS}}$ after the serial receive buffer has 14 bytes of space. The maximum space available for receiving serial data is 109 bytes, which is enough to hold 1.5 full packets of data.

Flow control threshold

Use the **FT** parameter to set the flow control threshold. Since the receive serial buffer is 109 bytes, you cannot set **FT** to more than 109-7 = 102 bytes. This allows up to 7 bytes of data to come in after CTS is de-asserted before data is dropped. The default value of **FT** is 81, leaving space for an external device that responds slowly to $\overline{\text{CTS}}$ being de-asserted. The minimum value of **FT** is 7, which is the minimal operational level.

$\overline{\text{RTS}}$ flow control

If you send the **D6** command to enable $\overline{\text{RTS}}$ flow control, the device does not send data in the serial transmit buffer out the DOUT pin as long as $\overline{\text{RTS}}$ is de-asserted (set high). Do not de-assert $\overline{\text{RTS}}$ for long periods of time or the serial transmit buffer will fill. If the device receives an RF data packet and the serial transmit buffer does not have enough space for all of the data bytes, it discards the entire RF data packet.

If the device sends data out the UART when $\overline{\text{RTS}}$ is de-asserted (set high) the device could send up to five characters out the UART port after RTS is de-asserted.

SPI port

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

SPI signals

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module supports 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 device:

Signal	SMT pin #	SMT applicable AT command	TH Pin #	TH applicable AT command
SPI_MOSI (Master out, Slave in)	16	**P6**	11	**D4**
SPI_MISO (Master in, Slave out)	17	**P5**	4	**P2**
SPI_SCLK (Serial clock)	14	**P8**	18	**D2**
SPI_$\overline{\text{SSEL}}$ (Slave select)	15	**P7**	17	**D3**
SPI_$\overline{\text{ATTN}}$ (Attention)	12	**P9**	19	**D1**

By default, the inputs have pull-up resistors enabled. On through-hole devices, you can use 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 device requires the pull-ups for proper UART operation.

Signal description

SPI_MISO: When SPI_CLK is active, the device outputs the data on SPI_MISO at the SPI_CLK rate. If there are other SPI slave devices connected to the same SPI master, then the SPI_MISO output from XBee device must be externally tri-stated when SPI_SSEL is de-asserted to prevent multiple devices from driving SPI_MISO.

SPI_MOSI: The SPI master outputs data on this line at the SPI_CLK rate after it selects the desired slave. When you configure the device for SPI operations, this pin is an input.

SPI_SCLK: The SPI master outputs a clock on this pin, and the rate must not exceed the maximum allowed, 5 Mb/s. This signal clocks data transfers on MOSI and MISO.

SPI_$\overline{\text{SSEL}}$: The SPI master outputs a low signal on this pin to select the device as an SPI slave. When you configure the device for SPI operations, this pin is an input. This signal enables serial communication with the slave.

SPI_$\overline{\text{ATTN}}$: The device asserts this pin low when it has data to send to the SPI master. When you configure this pin for SPI operations, it is an output (not tri-stated). This signal alerts the master that the slave has data queued to send. The device asserts this pin as soon as data is available to send to the SPI master and it remains asserted until the SPI master has clocked out all available data.

SPI parameters

Most host processors with SPI hardware allow you to set the bit order, clock phase and polarity. For communication with all XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules, 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

All XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules use SPI mode 0 and MSB first. 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.

SPI and API mode

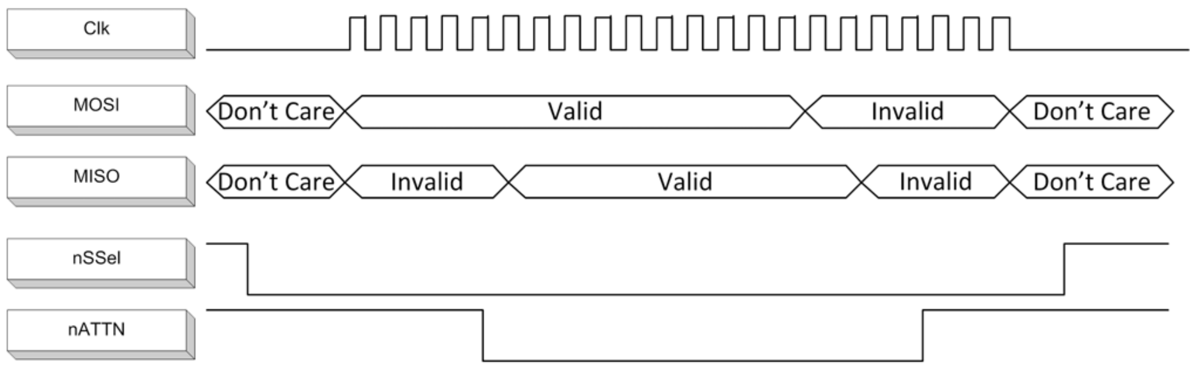
The SPI only operates in API mode 1. The SPI does not support Transparent mode or API mode 2 (with escaped characters). This means that the **AP** configuration only applies to the UART interface and is ignored while using the SPI.

Full duplex operation

When using SPI on the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module the device uses API operation without escaped characters to packetize data. The device ignores the configuration of **AP** because SPI does not operate in any other mode. SPI is a full duplex protocol, even when data is only available in one direction. This means that whenever a device receives data, it also transmits, and that data is normally invalid. Likewise, whenever a device transmits data, invalid data is probably received. To determine whether or not received data is invalid, the firmware places the data in 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 sends data to the slave and the slave has valid data to send in the middle of receiving data from the master, a full duplex operation occurs, 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.

The following figure illustrates the SPI interface while valid data is being sent in both directions.

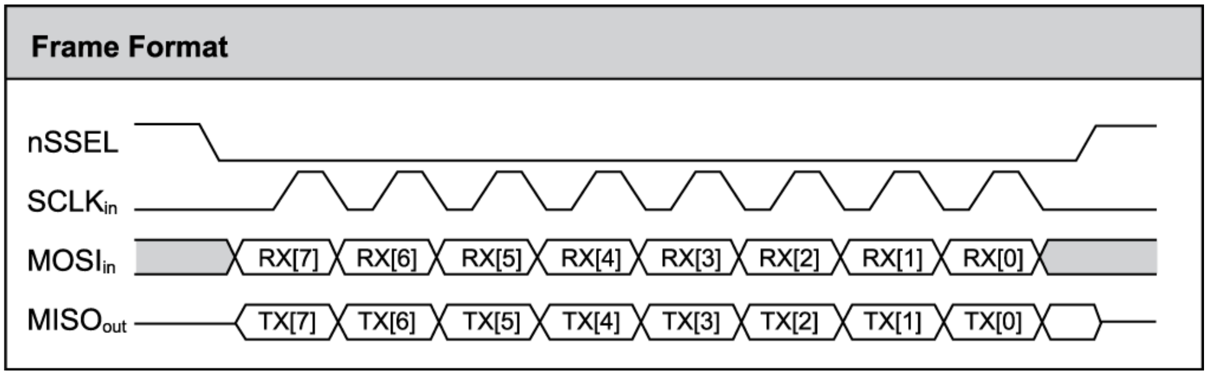


Slave mode characteristics

In slave mode, the following apply:

- SPI Clock rates up to 5 MHz (5 Mb/s) are possible.
- Data is MSB first.
- It uses Frame Format Mode 0. This means CPOL= 0 (idle clock is low) and CPHA = 0 (data is sampled on the clock's leading edge). The picture below diagrams Mode 0.
- The SPI port is setup for API mode and is equivalent to **AP** = 1.

The following picture shows the frame format for SPI communications.


Frame Format
nSSEL
SCLKin
MOSIin
MISOout
RX[7]
RX[6]
RX[5]
RX[4]
RX[3]
RX[2]
RX[1]
RX[0]
TX[7]
TX[6]
TX[5]
TX[4]
TX[3]
TX[2]
TX[1]
TX[0]

I/O support

Digital I/O line support

Digital I/O is available on lines DIO0 through DIO12 (D0 - D9 and P0 - P2). Each of these pins may be configured as 3, 4, or 5 with the following meanings:

- 3 is digital input
- 4 is digital output low
- 5 is digital output high

Function	Pin	Command
DIO0	TH pin 20/SMT pin 33	D0 (DIO0/AD0)
DIO1	TH pin 19/SMT pin 32	D1 (DIO1/AD1)
DIO2	TH pin 18/SMT pin 31	D2 (DIO2/AD2)
DIO3	TH pin 17/SMT pin 30	D3 (DIO3/AD3)
DIO4	TH pin 11/SMT pin 24	D4 (DIO4)
DIO5	TH pin 15/SMT pin 28	D5 (DIO5/ASSOCIATED_INDICATOR)
DIO6	TH pin 16/SMT pin 29	D6 (DIO6/RTS)
DIO7	TH pin 12/SMT pin 25	D7 (DIO7/CTS)
DIO8	TH pin 9/SMT pin 10	D8 (DIO8/DTR/SLEEP_REQUEST)
DI09		D9 (ON_SLEEP)
DIO10	TH pin 6/SMT pin 7	P0 (DIO10/RSSI/PWM0 Configuration)
DIO11	TH pin 7/SMT pin 8	P1 (DIO11/PWM1 Configuration)
DIO12	TH pin 4/SMT pin 5	P2 (DIO12/SPI_MISO Configuration)

Analog input

Analog input is available on D0 through D3. To use analog input, set these parameters to 2.

Monitor I/O lines

You can use IS (Force Sample) to query the current state of all digital input and ADC lines on the device. If no inputs are defined, the command returns an ERROR.

If you send the **IS** command from Command mode, then the device returns a carriage return delimited list containing the following fields.

Field	Name	Description
1	Sample sets	Number of sample sets in the packet. Always set to 1.

Field	Name	Description
2	Digital channel mask	Indicates which digital I/O lines have sampling enabled. Each bit corresponds to one digital I/O line on the device. 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 For example, a digital channel mask of 0x002F means DIO0, 1, 2, 3 and 5 are enabled as digital I/O.
1	Analog channel mask	Indicates which lines have analog inputs enabled for sampling. Each bit in the analog channel mask corresponds to one analog input channel. bit 0 = AD0/DIO0 bit 1 = AD1/DIO1 bit 2 = AD2/DIO2 bit 3 = AD3/DIO3
Variable	Sampled data set	If you enable any digital I/O lines, the first two bytes of the data set indicate the state of all enabled digital I/O. Only digital channels that you enable in the digital channel mask bytes have any meaning in the sample set. If you do not enable any digital I/O on the device, it omits these two bytes. Following the digital I/O data (if there is any), each enabled analog channel returns two bytes. The data starts with AD0 and continues sequentially for each enabled analog input channel up to AD3.

If you issue the **IS** command using a local or remote AT Command API frame, then the device returns an AT Command Response (0x88 or 0x97) frame with the I/O data included in the command data portion of the packet.

Example	Sample AT response
0x01	[1 sample set]
0x0C0C	[Digital inputs: DIO 2, 3, 10, 11 enabled]
0x03	[Analog inputs: A/D 0, 1 enabled]
0x0408	[Digital input states: DIO 3, 10 high, DIO 2, 11 low]
0x03D0	[Analog input: ADIO 0 = 0x3D0]
0x0124	[Analog input: ADIO 1 = 0x120]

On demand I/O sampling

You can use the **IS** (Force Sample) command to sample pins configured as digital I/O and analog input. If no pins are configured in this manner (with the **DO** - **D8** commands set to 2, 3, 4, or 5), then the **IS** command returns an error.

In Command mode, the output is:

Output	Description
01	Indicates one sample. That is the only possibility for Command mode.
20E	Mask to indicate which lines are sampled (A0, D3, D2, and D1).
00A	Digital sample indicates D3 high, D2 low, and D1 high.
3FF	Analog sample for A0 indicates that A0 is reading maximum voltage of 1.2 V.

In API mode, the output is:

```
7E 00 0C 83 00 00 00 00 01 03 3E 01 2A 02 10 FD
```

In this example, note the following:

83 indicates RX Packet: 16-bit Address I/O frame (0x83).

00 00 indicates 16-bit source address.

00 indicates RSSI (does not apply).

00 indicates options.

01 indicates the number of samples.

03 3E mask to indicate which lines are sampled (A0, D8, D5, D4, D3, D2, and D1).

01 2A digital sample that indicates that D8 is high, D5 is high, D4 is low, D3 is high, D2 is low, and D1 is high.

02 10 indicates that A0 has input voltage nearly half of capacity, where 03 FF would indicate the full voltage of 1.2 V = 1200 mV.

For a remote **IS** command sent to the device listed above with the same configuration, the output is:

```
7E 00 16 97 01 00 13 A2 00 40 E3 C0 15 00 00 49 53 00 01 03 3E 01 2A 02 10 9F
```

In this example, note the following:

97 indicates Remote AT Command Response frame (0x97).

01 is the frame ID.

00 13 A2 00 40 E3 C0 15 is the 64-bit source address.

00 00 indicates 16-bit source address.

49 53 (IS) indicates command response to the **IS** command.

00 indicates the status is OK.

01 indicates the number of samples.

03 3E mask to indicate which lines are sampled (A0, D8, D5, D4, D3, D2, and D1).

01 2A digital sample that indicates that D8 is high, D5 is high, D4 is low, D3 is high, D2 is low, and D1 is high.

02 10 indicates that A0 has input voltage about half of capacity, where 03 FF would indicate full voltage of 1.2 V = 1200 mV.

Periodic I/O sampling

Periodic sampling allows a device to take an I/O sample and transmit it to a remote device at a periodic rate. Use the **IR** command to set the periodic sample rate.

- To disable periodic sampling, set **IR** to **0**.
- For all other **IR** values, the firmware samples data when **IR** milliseconds elapse and the sample data transmits to a remote device.

The **DH** and **DL** commands determine the destination address of the I/O samples.

Only devices with API operating mode enabled send I/O data samples out their serial interface. Devices that are in Transparent mode (**AP** = **0**) discard the I/O data samples they receive. You must configure at least one pin as a digital or ADC input to generate sample data.

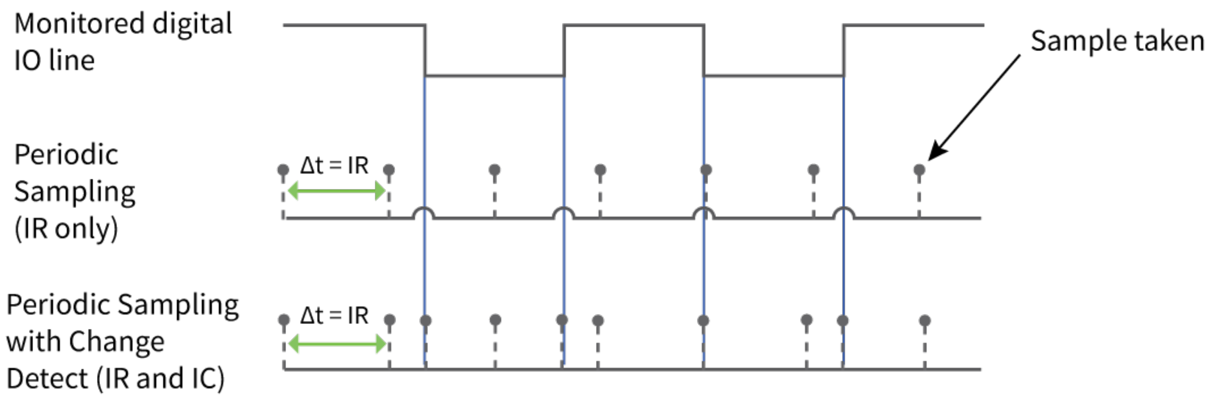
Although samples may be taken every millisecond, **IR** should be at least 20 milliseconds. This allows time for OTA transmission and output on the serial port of the receiving device.

A device with sleep enabled transmits periodic I/O samples at the **IR** rate until the **ST** time expires and the device can resume sleeping. For more information, see Sleep support.

Detect digital I/O changes

You can configure devices to transmit a data sample immediately whenever a monitored digital I/O pin changes state. The **IC** command is a bitmask that you use to set which digital I/O lines to monitor for a state change. If you set one or more bits in **IC**, the device transmits an I/O sample as soon it observes a state change in one of the monitored digital I/O lines using edge detection.

The figure below shows how I/O change detection can work with periodic sampling. In the figure, the gray dashed lines with a dot on top represent samples taken from the monitored DIO line. The top graph shows only **IR** samples, the bottom graph shows a combination of **IR** samples and **IC** (Change Detect). In the top graph, the humps indicate that the sample was not taken at that exact moment and needed to wait for the next **IR** sample period.


Monitored digital IO line
Sample taken
Periodic Sampling (IR only)
Δt = IR
Periodic Sampling with Change Detect (IR and IC)
Δt = IR

Note Use caution when combining Change Detect sampling with sleep modes. **IC** only causes a sample to be generated if the change takes place during a wake period. If the device is sleeping when the digital input transition occurs, then no change is detected and an I/O sample is not generated. Use **IR** in conjunction with **IC** in this instance, since **IR** generates an I/O sample upon wakeup and ensures that the change is properly observed.

I/O line passing

You can configure XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules to perform analog and digital line passing. When a device receives an RF I/O sample data packet, you can set up the receiving device to update any enabled outputs (PWM and DIO) based on the data it receives.

Digital I/O lines are mapped in pairs; pins configured as digital input on the transmitting device affect the corresponding digital output pin on the receiving device. For example: DI5 (pin 25) can only update DO5 (pin 25).

For Analog Line Passing, the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module has two PWM output pins that simulate the voltage measured by the ADC lines AD0 and AD1. For example, when configured as an ADC, AD0 (pin 33) updates PWM0 (pin 7); AD1 (pin 32) updates PWM1 (pin 8).

The default setup is for outputs to not be updated. Instead, a device sends I/O sample data out the serial interface in API mode, even if the destination node is not configured for API mode. You can use the **IU** command to disable sample data output.

On the destination node, the **IU** parameter enables the serial port to output I/O samples it receives. **IU** is set to 1 by default. If **IU** is set and the destination node is not in Command mode, it displays samples it receives on its serial port in API format. The AP parameter is ignored in this case.

To enable updating the outputs, set the **IA** (I/O Input Address) parameter with the address of the device that has the appropriate inputs enabled. This effectively binds the outputs to a particular device’s input. This does not affect the ability of the device to receive I/O line data from other devices - only its ability to update enabled outputs. Set the **IA** parameter to 0xFFFF (broadcast address) to set up the device to accept I/O data for output changes from any device on the network.

For line passing to function, the device configured with inputs must generate sample data.

I/O line passing details

The same message is received for both I/O sampling and for I/O line passing. But I/O line passing only occurs if **IA** matches the address of the sending node or if **IA** is 0xFFFF to match a sample from any node. The default value of **IA** is 0xFFFFFFFFFFFFFFFF, which prevents I/O line passing from occurring on the node because no node has that address. Additionally, the receiving device must have a matching value for output. For example, if an ADC0 sample is received, then **P0** must be configured with 2 for PWM output. Otherwise, the analog signal will not be reflected with a matching PWM signal. Likewise, if the sample indicates that D2 is high, but **D2** is not set to 4 or 5 on the receiving device, then the D2 pin will not be affected by I/O line passing.

When a digital output pin is set to something different than its configured value, that pin may return to its configured value after the time specified for the corresponding timer. **T0** specifies how long D0 will hold its non-configured value and **T1** specifies how long D1 will hold its non-configured value (**Q1** is for P1 and so forth). A value of 0 indicates that a pin holds the value of the input of the corresponding device indefinitely and a value greater than 0xFF specifies how many tenth second units the pin holds the non-configured value.

For PWM outputs, PT timer applies to both PWM0 and PWM1. A value of 0x00 allows the PWM pin to output a duty cycle reflective of the analog input indefinitely and a value larger than 0 indicates how many 10th second units before PWM output reverts to the duty cycle specified by **M0** or **M1**.

Networking

Network identifiers

You define DigiMesh networks with a unique network identifier. Use the **ID** command to set this identifier. For devices to communicate, you must configure them with the same network identifier and the same operating channel. For devices to communicate, the **CH** and **ID** commands must be equal on all devices in the network.

The **ID** command directs the devices to talk to each other by establishing that they are all part of the same network. The **ID** parameter allows multiple DigiMesh networks to co-exist on the same physical channel.

Operating channels

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module operates over the 2.4 GHz band using direct sequence spread spectrum (DSSS) modulation. DSSS modulation allows the device to operate over a channel or frequency that you specify.

The 2.4 GHz frequency band defines 16 operating channels. XBee devices support all 16 channels and XBee-PRO devices support 12 of the 16 channels.

Use the **CH** command to select the operating channel on a device. **CH** tells the device the frequency to use to communicate.

For devices to communicate, the **CH** and **ID** commands must be equal on all devices in the network.

Note these requirements for communication:

- A device can only receive data from other devices within the same network (with the same **ID** value) and using the same channel (with the same **CH** value).
- A device can only transmit data to other devices within the same network (with the same **ID** value) and using the same channel (with the same **CH** value).

Delivery methods

The TO (Transmit Options) command sets the default delivery method that the device uses when in Transparent mode. In API mode, the TxOptions field of the API frame overrides the **TO** command, if non-zero.

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module supports three delivery methods:

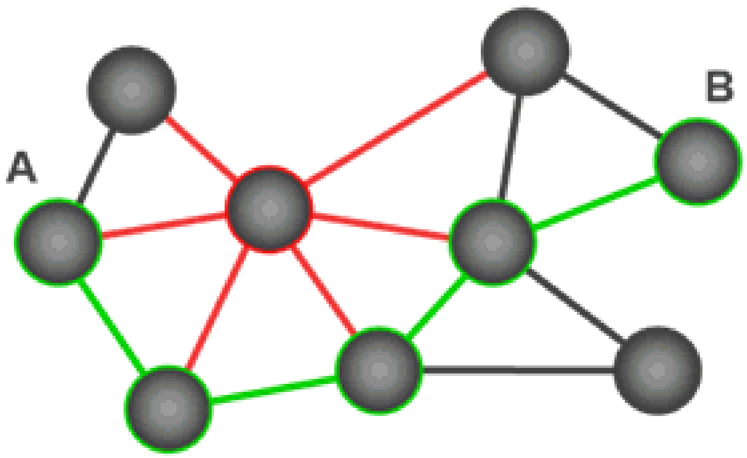
- Point-to-multipoint (**TO** = 0x40).
- Repeater (directed broadcast) (**TO** = 0x80).
- DigiMesh (**TO** = 0xC0).

DigiMesh networking

A mesh network is a topology in which each node in the network is connected to other nodes around it. Each node cooperates in transmitting information. Mesh networking provides these important benefits:

- **Routing**. With this technique, the message is propagated along a path by hopping from node to node until it reaches its final destination.
- **Ad-hoc network creation**. This is an automated process that creates an entire network of nodes on the fly, without any human intervention.

- **Self-healing**. This process automatically figures out if one or more nodes on the network is missing and reconfigures the network to repair any broken routes.
- **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 acknowledgments**. Only the destination node will reply to route requests.
- **Reliable delivery**. Reliable delivery of data is accomplished by means of acknowledgments.
- **Sleep modes.** Low power sleep modes with synchronized wake are supported with variable sleep and wake times.



With mesh networking, the distance between two nodes does not matter as long as there are enough nodes in between to pass the message along. When one node wants to communicate with another, the network automatically calculates the best path.

A mesh network is also reliable and offers redundancy. For example, If a node can no longer operate because it has been removed from the network or because a barrier blocks its ability to communicate, the rest of the nodes can still communicate with each other, either directly or through intermediate nodes.

Note Mesh networks use more bandwidth for administration and therefore have less available for payloads.

Broadcast addressing

All of the routers in a network receive and repeat broadcast transmissions. Broadcast transmissions do not use ACKs, so the sending device sends the broadcast multiple times. By default, the sending device sends a broadcast transmission four times. The transmissions become automatic retries without acknowledgments. This results in all nodes repeating the transmission four times as well.

In order to avoid RF packet collisions, the network inserts a random delay before each router relays the broadcast message. You can change this random delay time with the **NN** parameter.

Sending frequent broadcast transmissions can quickly reduce the available network bandwidth. Use broadcast transmissions sparingly.

The broadcast address is a 64 bit address with the lowest 16 bits set to 1. The upper bits are set to 0. To send a broadcast transmission:

- Set **DH** to 0.
- Set **DL** to 0xFFFF.

In API operating mode, this sets the destination address to 0x000000000000FFFF.

Unicast addressing

When devices transmit using DigiMesh unicast, the network uses retries and acknowledgments (ACKs) for reliable data delivery. In a retry and acknowledgment scheme, for every data packet that a device sends, the receiving device must send an acknowledgment back to the transmitting device to let the sender know that the data packet arrived at the receiver. If the transmitting device does not receive an acknowledgment then it re-sends the packet. It sends the packet a finite number of times before the system times out.

The **MR** (Mesh Network Retries) parameter determines the number of mesh network retries. The sender device transmits RF data packets up to **MR** + 1 times across the network route, and the receiver transmits ACKs when it receives the packet. If the sender does not receive a network ACK within the time it takes for a packet to traverse the network twice, the sender retransmits the packet.

If a device sends a unicast that uses both MAC and NWK retries and acknowledgments:

- Use MAC retries and acknowledgments for transmissions between adjacent devices in the route.
- Use NWK retries and acknowledgments across the entire route.

To send unicast messages while in Transparent operating mode, set the **DH** and **DL** on the transmitting device to match the corresponding **SH** and **SL** parameter values on the receiving device.

Route discovery

Route discovery is a process that occurs when:

1. The source node does not have a route to the requested destination.
2. A route fails. This happens when the source node uses up its network retries without receiving an ACK.

Route discovery begins by the source node broadcasting a route request (RREQ). We call any router that receives the RREQ and is not the ultimate destination, 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, the node saves, updates and broadcasts the RREQ.

When the ultimate destination receives the RREQ, it unicasts a route reply (RREP) back to the source node along the path of the RREQ. It does this regardless of route quality and regardless of how many times it has seen an RREQ 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 uses for the queued packet and for subsequent packets with the same destination address.

Routing

A device within a mesh network determines reliable routes using a routing algorithm and table. The routing algorithm uses a reactive method derived from Ad-hoc On-demand Distance Vector (AODV). The firmware uses an associative routing table to map a destination node address with its next hop. A device sends a message to the next hop address, and the message either reaches its destination or forwards to an intermediate router that routes the message on to its destination.

If a message has a broadcast address, it is broadcast to all neighbors, then all routers that receive the message rebroadcast the message **MT**+1 times. Eventually, the message reaches the entire network.

Packet tracking prevents a node from resending a broadcast message more than **MT**+1 times. This means that a node that relays a broadcast will only relay it after it receives it the first time and it will discard repeated instances of the same packet.

Routers and end devices

You can use the **CE** command to configure devices in a DigiMesh network to act as routers or end devices. All devices in a DigiMesh network act as routers by default. Any devices that you configure as routers actively relay network unicast and broadcast traffic.

Repeater/directed broadcast

All of the routers in a network receive and repeat directed broadcast transmissions. Because it does not use ACKs, the originating node sends the broadcast multiple times. By default a broadcast transmission is sent four times—the extra transmissions become automatic retries without acknowledgments. This results in all nodes repeating the transmission four times. Sending frequent broadcast transmissions can quickly reduce the available network bandwidth, so use broadcast transmissions sparingly.

MAC layer

The MAC layer is the building block that is used to build repeater capability. To implement Repeater mode, we use 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, the device sends both unicast and broadcast packets out as broadcasts that are always repeated. All repeated packets are sent to every device. The devices that receive the broadcast send broadcast data out their serial port.

When a device sends a unicast, it specifies a destination address in the network header. Then, only the device that has the matching destination address sends the unicast out its serial port. This is called a directed broadcast.

Any node that has a **CE** parameter set to router rebroadcasts the packet if its **BH** (broadcast hops) or broadcast radius values are not depleted. If a node has already seen a repeated broadcast, it ignores the broadcast.

The **NH** parameter sets the maximum number of hops that a broadcast transmission is repeated. The device always uses the **NH** value unless you specify a **BH** value 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. If you change the **CE** parameter, you can limit which nodes repeat packets, which helps dense networks from becoming overly congested while packets are being repeated.

Transmission timeout calculations for Repeater/directed broadcast mode are the same as for DigiMesh broadcast transmissions.

Point-to-multipoint

To select point-to-multipoint, set the transmit options to 0x40.

In Transparent mode, use the **TO** (Transmit Options) command to set the transmit options.

In API mode, use the Transmit Request (0x10) and Explicit Addressing Command (0x11) frames to set the transmit options. However, if the transmit options in the API frame are zero, then the transmit options in the **TO** command apply.

Point-to-multipoint transmissions occur between two adjacent nodes within RF range. No route discovery and no routing occur for these types of transmissions. The networking layer is entirely skipped.

Point-to-multipoint has an advantage over DigiMesh for two adjacent devices due to less overhead. However, it cannot work over multiple hops.

Encryption

Set EE (Encryption Enable) to **1** to enable encryption. Use KY (AES Encryption Key) to set an encryption key and the same key must be set on each device in the network.

Starting with firmware version 9002, the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module supports 256-bit AES counter mode encryption. We recommend using this enhanced security mode to provide greater security against replay attacks and attempts to determine the plaintext.

Use **C8 (Compatibility Options)** bit 2 to select an encryption mode.

256-bit AES Counter Mode encryption

This security mode uses Counter (CTR) mode encryption instead of Electronic Codebook (ECB) mode encryption. Since the counter is passed over-the-air (OTA) and changes with each frame, the same text is always encrypted differently and there are no known attacks to determine the plaintext from the ciphertext.

A side effect of this implementation is that the maximum payload is reduced by the size of the counter (8 bytes). Therefore, no frames can exceed 65 bytes with encryption enabled. The maximum payload is still 73 bytes with encryption disabled.

Also effective starting with version 9002, the key is 256 bits rather than 128 bits. 256 bits is 32 bytes. Since the key is entered with ASCII HEX characters in Command mode, up to 64 ASCII HEX characters may be entered for the **KY** command.

This security mode is compatible with other XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules running version 9002 or greater and XBee3 DigiMesh modules. This security mode is not enabled by default. To enable this enhanced security mode, clear C8 (802.15.4 Compatibility) bit 2.

128-bit AES Electronic Codebook encryption

This mode is enabled by default, however we recommend using 256-bit AES CTR mode encryption whenever possible.

For compatibility with nodes in the same network that do not support CTR mode encryption, setting **C8** bit 2 enables legacy mode 128-bit ECB mode encryption as supported previously. In this case, only the last 32 ASCII HEX characters of the key are used, even if more characters were previously entered for the key.

128-bit encryption refers to the length of the encryption key entered with the **KY** command (128 bits = 16 bytes).

802.15.4 security modes

The 802.15.4 protocol specifies eight security modes, enumerated as shown in the following table.

Level	Name	Encrypted?	Length of message integrity check	Packet length overhead
0	N/A	No	0 (no check)	0
1	MIC-32	No	4	9
2	MIC-64	No	8	13
3	MIC-128	No	16	21
4	ENC	Yes	0 (no check)	5
5	ENC-MIC-32	Yes	4	9
6	ENC-MIC-64	Yes	8	13
7	ENC-MIC-128	Yes	16	21

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module only supports security levels 0 and 4. It does not support message integrity checks. **EE** 0 selects security level 0 and **EE** 1 selects security level 4. When using encryption, all devices in the network must use the same 16-byte encryption key for valid data to get through. Mismatched keys will corrupt the data output on the receiving device. Mismatched **EE** parameters will prevent the receiving device from outputting received data.

Maximum payload

There is a maximum payload that you can send at one time. Use the **NP** (Maximum Packet Payload Bytes) command to read the device's maximum payload.

These maximums only apply in API mode. If you attempt to send an API packet with a larger payload than specified, the device responds with a Transmit Status frame (0x89) with the Status field set to 74 (Data payload too large).

In Transparent mode, the firmware splits the data as necessary to cope with maximum payloads.

DigiMesh throughput

Throughput in a DigiMesh network can vary due to a number of variables, including:

- The number of hops.
- If you enable or disable encryption.
- Sleeping end devices.
- Failures and route discoveries.

Our empirical testing shows the following throughput performance in a robust operating environment with low interference.

Configuration	Data throughput
1 hop, encryption disabled	27.0 kb/s
3 hop, encryption disabled	10.9 kb/s

Configuration	Data throughput
6 hop, encryption disabled	5.78 kb/s
1 hop, encryption enabled	20.5 kb/s
3 hop, encryption enabled	9.81 kb/s
6 hop, encryption enabled	4.70 kb/s

We performed data throughput measurements with the serial interface rate set to 115200 b/s, and measured the time to send 100,000 bytes from the source to the destination. During the test, there were no route discoveries or failures.

Network commissioning and diagnostics

We call the process of discovering and configuring devices in a network for operation, "network commissioning." Devices include several device discovery and configuration features. In addition to configuring devices, you must develop a strategy to place devices to ensure reliable routes. To accommodate these requirements, modules include features to aid in placing devices, configuring devices, and network diagnostics.

Local configuration

You can configure devices locally using serial commands in Transparent or API mode, or remotely using remote API commands. Devices that are in API mode can send configuration commands to set or read the configuration settings of any device in the network.

Remote configuration

When you do not have access to the device's serial port, you can use a separate device in API mode to remotely configure it. To remotely configure devices, use the following steps.

Send a remote command

To send a remote command, populate the Remote AT Command Request frame - 0x17 with:

1. The 64-bit address of the remote device.
2. The correct command options value.
3. Optionally, the command and parameter data.
4. If you want a command response, set the Frame ID field to a non-zero value.

The firmware only supports unicasts of remote commands. You cannot broadcast remote commands.

XCTU has a Frames Generator tool that can assist you with building and sending a remote AT frame; see Frames generator tool in the *XCTU User Guide*.

Apply changes on remote devices

When you use remote commands to change the command parameter settings on a remote device, you must apply the parameter changes or they do not take effect. For example, if you change the **BD** parameter, the actual serial interface rate does not change on the remote device until you apply the changes. You can apply the changes using remote commands in one of three ways:

1. Set the apply changes option bit in the API frame.
2. Send an **AC** command to the remote device.
3. Send the **WR** command followed by the **FR** command to the remote device to save the changes and reset the device.

Remote command response

If a local device sends a command request to a remote device, and the API frame ID is non-zero, the remote device sends a remote command response transmission back to the local device.

When the local device receives a remote command response transmission, it sends a remote command response API frame out its UART. The remote command response indicates:

1. The status of the command, which is either success or the reason for failure.
2. In the case of a command query, it includes the register value.

The device that sends a remote command does not receive a remote command response frame if:

1. It could not reach the destination device.
2. You set the frame ID to 0 in the remote command request.

Establish and maintain network links

Build aggregate routes

In many applications, many or all of the nodes in the network must 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, you can use the **AG** command to automatically build routes to an aggregate node in a DigiMesh network.

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

Upon deploying a DigiMesh network, you can send the **AG** command on the desired aggregator node to cause all nodes in the network to build routes to the aggregator node. You can optionally use the **AG** command 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 device; typically you should replace this value with the 64-bit address of the node sending the **AG** broadcast. However, if you do not want to update the **DH**/**DL** of the device receiving the **AG** broadcast you can use the invalid address of 0xFFFE. The receiving nodes that are configured in API mode output an Aggregator Update API frame (0x8E) if they update their **DH**/**DL** address; for a description of the frame, see Aggregate Addressing Update frame - 0x8E.

All devices that receive an **AG** broadcast update their routing table information to build a route to the sending device, regardless of whether or not their **DH**/**DL** address is updated. The devices use this routing information for future DigiMesh unicast transmissions.

DigiMesh routing examples

Example one:

In a scenario where you deploy a network, and then you want to update the **DH** and **DL** registers of all the devices in the network so that they use the MAC address of the aggregator node, which has the MAC address 0x0013A200 4052C507, you could use the following technique.

1. Deploy all devices in the network with the default **DH**/**DL** of 0xFFFF.
2. Serially, send an ATAGFFFF command to the aggregator node so it sends the broadcast transmission to the rest of the nodes.

All the nodes in the network that receive the **AG** broadcast set their **DH** to 0x0013A200 and their **DL** to 0x4052C507. These nodes automatically build a route to the aggregator node.

Example two:

If you want all of the nodes in the network to build routes to an aggregator node with a MAC address of 0x0013A200 4052C507 without affecting the **DH** and **DL** registers of any nodes in the network:

1. Send the ATAGFFFE command to the aggregator node. This sends an **AG** broadcast to all of the nodes in the network.

2. All of the nodes internally update only their routing table information to contain a route to the aggregator node.
3. None of the nodes update their **DH** and **DL** registers because none of the registers are set to the 0xFFFE address.

Replace nodes

You can use the **AG** command to update the routing table and **DH**/**DL** registers in the network after you replace a device. To update only the routing table information without affecting the **DH** and **DL** registers, use the process in example two, above.

To update the **DH** and **DL** registers of the network, use example three, below.

Example three:

This example shows how to cause all devices to update their **DH** and **DL** registers to the MAC address of the sending device. In this case, assume you are using a device with a serial number of 0x0013A200 4052C507 as a network aggregator, and the sending device has a MAC address of 0x0013A200 F5E4D3B2 To update the **DH** and **DL** registers to the sending device's MAC address:

1. Replace the aggregator with 0x0013A200 F5E4D3B2.
2. Send the ATAG0013A200 4052C507 command to the new device.

Test links between adjacent devices

It often helps to test the quality of a link between two adjacent modules in a network. You can use the Test Link Request Cluster ID to send a number of test packets between any two devices in a network. To clarify the example, we refer to "device A" and "device B" in this section.

To request that device B perform a link test against device A:

1. Use device A in API mode (**AP** = **1**) to send an Explicit Addressing Command (0x11) frame to device B.
2. Address the frame to the Test Link Request Cluster ID (0x0014) and destination endpoint: 0xE6.
3. Include a 12-byte payload in the Explicit Addressing Command frame with the following format:

Number of bytes	Field name	Description
8	Destination address	The address the device uses to test its link. For this example, use the device A address.
2	Payload size	The size of the test packet. Use the **NP** command to query the maximum payload size for the device.
2	Iterations	The number of packets to send. This must be a number between 1 and 4000.

4. Device B should transmit test link packets.
5. When device B completes transmitting the test link packets, it sends the following data packet to device A's Test Link Result Cluster (0x0094) on endpoint (0xE6).
6. Device A outputs the following information as an API Explicit RX Indicator (0x91) frame:

Number of bytes	Field name	Description
8	Destination address	The address the device used to test its link.
2	Payload size	The size of the test packet device A sent to test the link.
2	Iterations	The number of packets that device A sent.
2	Success	The number of packets that were successfully acknowledged.
2	Retries	The number of MAC retries used to transfer all the packets.
1	Result	0x00 - the 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 you want to test the link between device A (**SH**/**SL** = 0x0013A200 40521234) and device B (**SH**/**SL**=0x0013A 200 4052ABCD) by transmitting 1000 40-byte packets:

Send the following API packet to the serial interface of device A.

In the following example packet, whitespace marks 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

When the test is finished, the following API frame may be received:

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

This means:

- 999 out of 1000 packets were successful.
- The device made 100 retries.
- **RR** = 10.
- maxRSSI = -80 dBm.
- minRSSI = -83 dBm.
- avgRSSI = -82 dBm.

If the Result field does not equal zero, an error has occurred. Ignore the other fields in the packet.

If the Success field equals zero, ignore the RSSI fields.

The device that sends the request for initiating the Test link and outputs the result does not need to be the sender or receiver of the test. It is possible for a third node, "device C", to request device A to perform a test link against device B and send the results back to device C to be output. It is also possible for device B to request device A to perform the previously mentioned test. In other words, the frames can be sent by either device A, device B or device C and in all cases the test is the same: device A sends data to device B and reports the results.

Trace route option

In many networks, it is useful to determine the route that a DigiMesh unicast takes to its destination; particularly, when you set up a network or want to diagnose problems within a network.

Note Because of the large number of Route Information Packet frames that a unicast with trace route enabled can generate, we suggest you only use the trace route option for occasional diagnostic purposes and not for normal operations.

The Transmit Request (0x10 and 0x11) frames contain a trace route option, which transmits routing information packets to the originator of the unicast using the intermediate nodes.

When a device sends a unicast with the trace route option enabled, the unicast transmits to its destination devices, which forward the unicast to its eventual destination. The destination device transmits a Route Information Packet (0x8D) frame back along the route to the unicast originator.

The Route Information Packet frame contains:

- Addressing information for the unicast.
- Addressing information for the intermediate hop.
- Timestamp
- Other link quality information.

For a full description of the Route Information Packet frame, see Route Information Packet frame - 0x8D.

Trace route example

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

- After the data packet makes a successful MAC transmission from device A to device B, device A outputs a Route Information Packet frame indicating that the transmission of the data packet from device A to device E was successful in forwarding one hop from device A to device B.
- After the data packet makes a successful MAC transmission from device B to device C, device B transmits a Route Information Packet frame to device A. When device A receives the Route Information packet, it outputs it over its serial interface.
- After the data packet makes a successful MAC transmission from device C to device D, device C transmits a Route Information Packet frame to device A (through device B). When device A receives the Route Information packet, it outputs it over its serial interface.
- After the data packet makes a successful MAC transmission from device D to device E, device D transmits a Route Information Packet frame to device A (through device C and device B). When device A receives the Route Information packet, it outputs it over its serial interface.

There is no guarantee that Route Information Packet frames will arrive in the same order as the route taken by the unicast packet. On a weak route, it is also possible for the transmission of Route Information Packet frames to fail before arriving at the unicast originator.

NACK messages

Transmit Request (0x10 and 0x11) frames contain a negative-acknowledge character (NACK) API option (Bit 2 of the Transmit Options field).

If you use this option when transmitting data, when a MAC acknowledgment failure occurs on one of the hops to the destination device, the device generates a Route Information Packet (0x8D) frame and sends it to the originator of the unicast.

This information is useful because it allows you to identify and repair marginal links.

RSSI indicators

The received signal strength indicator (RSSI) measures the amount of power present in a radio signal. It is an approximate value for signal strength received on an antenna.

You can use the **DB** command to measure the RSSI on a device. **DB** returns the RSSI value measured in -dBm of the last packet the device received. This number can be misleading in multi-hop 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.

To determine the **DB** value in hardware:

1. Set **PO** to 1 to enable the RSSI pulse-width modulation (PWM) functionality.
2. Use the DIO10/RSSI/PWM0 module pin (pin 6 in through-hole, pin 7 in surface-mount). When the device receives data, it sets the RSSI PWM duty cycle to a value based on the RSSI of the packet it receives.

This value only indicates the quality of the last hop of a multi-hop transmission. You could connect this pin to an LED to indicate if the link is stable or not.

Associate LED

The Associate pin (pin 15) provides an indication of the device's sleep status and diagnostic information. To take advantage of these indications, connect an LED to the Associate pin.

To enable the Associate LED functionality, set the **D5** command to 1; it is enabled by default. If enabled, the Associate pin is configured as an output. This section describes the behavior of the pin.

The pin functions as a power indicator.

Use the **LT** command to override the blink rate of the Associate pin. If you set **LT** to 0, the device uses the default blink time of 250 ms.

The following table describes the Associate LED functionality.

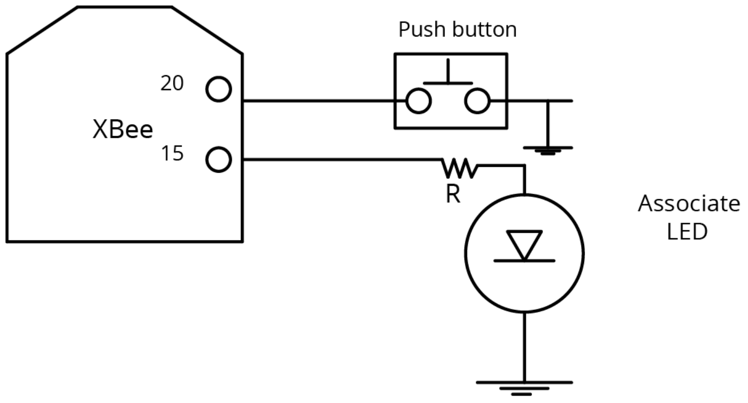
Sleep mode	LED Status	Meaning
0	On, blinking	The device has power and is operating properly
1, 4, 5	Off	The device is asleep
1, 4, 5	On, blinking	The device has power, is awake and is operating properly

Diagnostics support

The Associate pin works with the Commissioning Pushbutton to provide additional diagnostic behaviors to aid in deploying and testing a network. If you press the Commissioning Pushbutton once, the device transmits a broadcast Node Identification Indicator (0x95) frame at the beginning of the next wake cycle if the device is sleep compatible, or immediately if the device is not sleep compatible. If you enable the Associate LED functionality using the **D5** command, a device that receives this transmission blinks its Associate pin rapidly for one second.

The Commissioning Pushbutton

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module supports a set of commissioning and LED functions to help you deploy and commission devices. These functions include the Commissioning Pushbutton definitions and the associated LED functions. The following diagram shows how the hardware can support these features.



To support the Commissioning Pushbutton and its associated LED functions, connect a pushbutton and an LED to device pins 20 and 15 respectively.

Definitions

To enable the Commissioning Pushbutton functionality on pin 20, set the **D0** command to 1. The functionality is enabled by default.

You must perform multiple button presses within two seconds.

The following table provides the pushbutton definitions.

Button presses	Action
1	Immediately sends a Node Identification broadcast transmission. All devices that receive this transmission blink their Associate LED rapidly for one second. All devices in API operating mode that receive this transmission send a Node Identification Indicator frame (0x95) out their UART.
1	If the device is configured for asynchronous sleep, this wakes it for 30 seconds. Immediately sends a Node Identification broadcast transmission. All devices that receive this transmission blink their Associate LED rapidly for one second. All devices in API operating mode that receive this transmission send a Node Identification Indicator frame (0x95) out their UART.
4	Sends an **RE** command to restore device parameters to default values.

Use the Commissioning Pushbutton

Use the **CB** command to simulate button presses in software. Send **CB** with a parameter set to the number of button presses to perform. For example, if you send **ATCB1**, the device performs the action (s) associated with a single button press.

Node Identification Indicator frame - 0x95 is similar to Remote Command Response frame - 0x97 – it contains the device's address, node identifier string (**NI** command), and other relevant data. All devices in API operating mode that receive the Node Identification Indicator frame send it out their UART as a Node Identification Indicator frame.

Node discovery

Node discovery has three variations as shown in the following table:

Commands	Syntax	Description
Node Discovery	**ND**	Seeks to discover all nodes in the network (on the current Network ID).
Directed Node Discovery	**ND** <NI String>	Seeks to discover if a particular node named <NI String> is found in the network.
Destination Node	**DN** <NI String>	Sets **DH**/**DL** to point to the MAC address of the node whose <NI String> matches.

The node discovery command (without an NI string designated) sends out a broadcast to every node in the Network ID. Each node in the network sends a response back to the requesting node.

When the node discovery command is issued in Command mode, all other AT commands are inhibited until the node discovery command times out, as determined by the **N?** parameter. After the timeout, an extra CR is output to the terminal window, indicating that new AT commands can be entered. This is the behavior whether or not there were any nodes that responded to the broadcast.

When the node discovery command is issued in API mode, the behavior is the same except that the response is output in API mode. If no nodes respond, there will be no responses at all to the node discover command. The requesting node is not able to process a new AT command until **N?** times out.

Discover all the devices on a network

You can use the **ND** (Network Discovery) command to discover all devices on a network. When you send the **ND** command:

1. The device sends a broadcast **ND** command through the network.
2. All devices that receive the command send a response that includes their addressing information, node identifier string and other relevant information. For more information on the node identifier string, see NI (Node Identifier).

ND is useful for generating a list of all device addresses in a network.

When a device receives the network discovery command, it waits a random time before sending its own response. You can use the **NT** command to set the maximum time delay on the device that you use to send the **ND** command.

- The device that sends the **ND** includes its **NT** setting in the transmission to provide a delay window for all devices in the network.
- The default **NT** value is 0x82 (13 seconds).

Directed node discovery

The directed node discovery command (**ND** with an **NI** string parameter) sends out a broadcast to find a node in the network with a matching **NI** string. If such a node exists, it sends a response with its information back to the requesting node.

In Transparent mode, the requesting node outputs an extra carriage return following the response from the designated node and the command terminates; it is then ready to accept a new AT command. In the event that the requested node does not exist or is too slow to respond, the requesting node outputs an ERROR response after **N?** expires.

In API mode, the response from the requesting node will be output in API mode and the command will terminate immediately. If no response comes from the requested node, the requesting node outputs an error response in API mode after **N?** expires. The device's software assumes that each node has a unique **NI** string.

The directed node discovery command terminates after the first node with a matching **NI** string responds. If that **NI** string is duplicated in multiple nodes, the first responding node may not always be the same node or the desired node.

Destination Node

The Destination Node command (**DN** with an **NI** string parameter) sends out a broadcast containing the **NI** string being requested. The responding node with a matching **NI** string sends its information back to the requesting node. The local node then sets **DH**/**DL** to match the address of the responding node. As soon as this response occurs, the command terminates successfully. If the device is in AT command mode, an OK string is output and command mode exits. In API mode, you may enter another AT command.

If an **NI** string parameter is not provided, the **DN** command terminates immediately with an error. If a node with the given NI string does not respond, the **DN** command terminates with an error after **N?** times out.

In Transparent mode, unlike **ND** (with or without an **NI** string), **DN** does not cause the information from the responding node to be output; rather it simply sets **DH**/**DL** to the address of the responding node.

In API mode, the response from the requesting node outputs in API mode and the command terminates immediately. If no response comes from the requested node, the requesting node outputs an error response in API mode after **N?** expires.

The device's software assumes that each node has a unique **NI** string. The directed destination node command terminates after the first node with a matching **NI** string responds. If that **NI** string is duplicated in multiple nodes, **DH**/**DL** may not be set to the desired value.

Discover devices within RF range

The **FN** (Find Neighbor) command works the same as the **ND** (Node Discovery) except that it is limited to neighboring devices (devices that are only one hop away). See FN (Find Neighbors) for details.

- You can use the **FN** (Find Neighbors) command to discover the devices that are immediate neighbors (within RF range) of a particular device.
- **FN** is useful in determining network topology and determining possible routes.

You can send **FN** locally on a device in Command mode or you can use a local AT Command frame - 0x08.

To use **FN** remotely, send the target node a Remote AT Command Request frame - 0x17 using **FN** as the name of the AT command.

The device you use to send **FN** transmits a zero-hop broadcast to all of its immediate neighbors. All of the devices that receive this broadcast send an RF packet to the device that transmitted the **FN** command. If you sent **FN** remotely, the target devices respond directly to the device that sent the **FN** command. The device that sends **FN** outputs a response packet in the same format as an AT Command Response frame - 0x88.

The FN (Find Neighbors) command

The **FN** (Find Neighbors) command works exactly the same as the **ND** (Network Discover) command except that it is limited to neighboring devices (devices that are only one hop away). See FN (Find Neighbors).

Sleep support

Sleep modes

Sleep modes enable the device to enter states of low-power consumption when not in use. In order to enter Sleep mode, one of the following conditions must be met (in addition to the device having a non-zero **SM** parameter value):

- SLEEP_RQ/$\overline{\text{DTR}}$ (pin 9 on through-hole devices, pin 10 on surface-mount devices) is asserted and the device is in a pin sleep mode (**SM** = 1, or 5)
- The device is idle (no data transmission or reception) for the amount of time defined by the **ST** (Time before Sleep) parameter.

Note **ST** is only active when **SM** = 4 or 5.

The following table shows the sleep mode configurations.

Sleep mode	Description
SM 0	No sleep
SM 1	Pin sleep
SM 4	Cyclic sleep
SM 5	Cyclic sleep with pin wake-up

Pin Sleep mode (SM = 1)

Pin Sleep mode minimizes quiescent power (power consumed when in a state of rest or inactivity). This mode is voltage level-activated; when Sleep_RQ (pin 9 for through-hole, pin 10 for surface-mount) is asserted, the device finishes any transmit or receive activities, enters Idle mode, and then enters a state of sleep. The device does not respond to either serial or RF activity while in pin sleep.

To wake a sleeping device operating in Pin Sleep mode, de-assert Sleep_RQ. The device wakes when Sleep_RQ is de-asserted and is ready to transmit or receive when the CTS line is low. When waking the device, the pin must be de-asserted at least two 'byte times' after CTS goes low. This assures that there is time for the data to enter the DI buffer.

Cyclic Sleep mode (SM = 4)

The Cyclic Sleep modes allow devices to periodically check for RF data. When the **SM** parameter is set to 4, the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module is configured to sleep, then wakes once per cycle to check for data from a messaging coordinator. The Cyclic Sleep Remote sends a poll request to the messaging coordinator at a specific interval set by the **SP** (Cyclic Sleep Period) parameter. The messaging coordinator transmits any queued data addressed to that specific remote upon receiving the poll request.

If no data is queued for the remote, the messaging coordinator does not transmit and the remote returns to sleep for another cycle. If queued data is transmitted back to the remote, it stays awake to allow for back and forth communication until the **ST** (Time before Sleep) timer expires.

If configured, $\overline{\text{CTS}}$ goes low each time the remote wakes, allowing for communication initiated by the remote host if desired. If ON_SLEEP is configured it goes high (ON) after **SN** (Number of Cycles Between ON_SLEEP) sleep periods. Change **SN** to allow external circuitry to sleep for longer periods if no data is received.

Cyclic Sleep with Pin Wake-up mode (SM = 5)

Use this mode to wake a sleeping remote device through either the RF interface or by de-asserting SLEEP_RQ for event-driven communications. The cyclic sleep mode works as described previously with the addition of a pin-controlled wake-up at the remote device. The SLEEP_RQ pin is level-triggered. The device wakes when a low is detected then set CTS low as soon as it is ready to transmit or receive.

Any activity resets the **ST** (Time before Sleep) timer, so the device goes back to sleep only after there is no activity for the duration of the timer. Once the device wakes (pin-controlled), it ignores further pin activity. The device transitions back into sleep according to the **ST** time regardless of the state of the pin.

Sleep parameters

The following AT commands are associated with the sleep modes. See the linked commands for the parameter's description, range and default values.

- SM (Sleep Mode)
- SN (Number of Cycles Between ON_SLEEP)
- SO (Sleep Options)
- ST (Wake Time)
- SP (Sleep Time)
- WH (Wake Host Delay)

Sleep current

The following table shows the sleep current during the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module sleep modes.

Sleep mode	SM command setting	Sleep current
Pin sleep	1	<1 µA @ 25ºC
Cyclic sleep	4	<1 µA @ 25ºC
Cyclic sleep with pin wake-up	5	<1 µA @ 25ºC

You can make devices use low sleep current by driving PWM outputs high during sleep and by using internal pull-ups/pull-downs on disabled/unused pins. The sleep pins are set up for sleeping as specified in Sleep pins. Additionally, pins that are outputs (other than PWM outputs) continue to output the same levels during sleep. Normally, this means that pins configured for output high or low will output high or low accordingly. However, if the output is overridden by I/O line passing, then the overridden output level is maintained during the sleep time.

Sleep pins

The following table describes the three external device pins associated with sleep.

Pin name	Pin number	Description
SLEEP_RQ	TH pin 9/SMT pin 10	For **SM** = 1, high puts the device to sleep and low wakes it up. For **SM** = 5, a high to low transition wakes the device up for **ST** time. The device ignores a low to high transition in **SM** = 5.
$\overline{\text{CTS}}$	TH pin 12/SMT pin 25	If **D7** = 1, high indicates that the device is asleep and low indicates that it is awake and ready to receive serial data.
ON_$\overline{\text{SLEEP}}$	TH pin 13/SMT pin 26	Low indicates that the device is asleep and high indicates that it is awake and ready to receive serial data.

Indirect messaging and polling

Indirect messaging

Indirect messaging is a communication mode designed for communicating with asynchronous sleeping devices. A device 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 device holds onto the data until it is requested via a poll. On receiving a poll, the indirect messaging coordinator sends a queued data packet (if available) to the requestor.

Because it is possible for a 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. We suggest setting 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 queues up as many data packets as it has buffers available. After the coordinator uses all of its available buffers, it holds transmission requests unprocessed on the serial input queue. After the serial input queue is full, the device de-asserts $\overline{\text{CTS}}$ (if hardware flow control is enabled). After receiving a poll or purging data from the indirect messaging queue the buffers become available again.

Indirect messaging only functions with P2MP unicast messages. 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. To enable polling on a device, configure it as an indirect messaging poller with the **CE** command and set its **DH**:**DL** registers to match the **SH**:**SL** registers of the device that will function as the Indirect Messaging Coordinator. When you enable polling, the device sends a P2MP poll request regularly to the address specified by the **DH**:**DL** registers. When the device sends a P2MP unicast to the destination specified by the **DH**:**DL** of a polling device, the data also functions as a poll.

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

AT commands

Special commands

The following commands are special commands.

FR (Software Reset)

Resets the device. The device responds immediately with an **OK** and performs a reset 100 ms later.

If you issue **FR** while the device is in Command Mode, the reset effectively exits Command mode.

Parameter range

N/A

Default

N/A

RE (Restore Defaults)

Restore device parameters to factory defaults.

Parameter range

N/A

Default

N/A

AC (Apply Changes)

Immediately applies new settings without exiting Command mode.

Parameter range

N/A

Default

N/A

WR (Write)

Writes parameter values to non-volatile memory so that parameter modifications persist through subsequent resets.

Writing parameters to non-volatile memory does not apply the changes immediately. However, since the device uses non-volatile memory to determine initial configuration following reset, the written parameters are applied following a reset.

Note Once you issue a **WR** command, do not send any additional characters to the device until after you receive the **OK** response.

Parameter range

N/A

Default

N/A

MAC/PHY commands

The following AT commands are MAC/PHY commands.

CH (Operating Channel)

Set or read the operating channel devices used to transmit and receive data. The channel is one of two addressing configurations available to the device. The other configuration is the Network ID (**ID** command).

In order for devices to communicate with each other, they must share the same channel number. A network can use different channels to prevent devices in one network from listening to the transmissions of another. Adjacent channel rejection is 23 dB.

The command uses 802.15.4 channel numbers. Center frequency = 2405 MHz + (**CH** - 11 decimal) * 5 MHz.

Parameter range

0xB - 0x1A (XBee)

0x0C - 0x17 (XBee-PRO)

Default

0xC (12 decimal)

ID (Network ID)

Set or read the user network identifier.

Devices can only communicate with other devices that have the same network identifier and channel configured.

If using Original equipment manufacturer (OEM) network IDs, **0xFFFF** uses the factory value.

Parameter range

0 - 0xFFFF

Default

0x7FFF

RR (Unicast Mac Retries)

Set or read the maximum number of MAC level packet delivery attempts for unicasts. If **RR** is non-zero, the sent unicast packets request an acknowledgment from the recipient. Unicast packets can be retransmitted up to **RR** times if the transmitting device does not receive a successful acknowledgment.

Parameter range

0 - 0xF

Default

0xA (10 retries)

MT (Broadcast Multi-Transmits)

Set or read the number of additional MAC-level broadcast transmissions. All broadcast packets are transmitted **MT**+1 times to ensure they are received.

Parameter range

0 - 0xF

Default

3

PL (TX Power Level)

Sets or displays the power level at which the device transmits conducted power. Power levels are approximate.

For XBee, **PL** = 4, **PM** = 1 is tested at the time of manufacturing. Other power levels are approximate. On channel 26, transmitter power will not exceed -4 dBm.

Parameter range

0 - 4

The following table shows the TX power versus the **PL** setting.

XBee modules

PL setting	PM setting	Channel(s)	TX power* (dBm)
4	1	11 to 25	8
4	0	11 to 25	5
3	1	11 to 25	6
3	0	11 to 25	3
2	1	11 to 25	4
2	0	11 to 25	1
1	1	11 to 25	2
1	0	11 to 25	-1
0	1	11 to 25	-2
0	0	11 to 25	-5
X	1	26	-5
X	0	26	-8
* Highest power level is tested during manufacturing. Other power levels are approximate.			

XBee-PRO modules

PL setting	Channel(s)	TX power* (dBm)
0	12 to 23	0
1	12 to 23	12
2	12 to 23	15
3	12 to 23	16
4	12 to 23	18
* Highest power level is tested during manufacturing. Other power levels are approximate.		

Default

4

PM (Power Mode)

Set or read the power mode of the device. Enabling boost mode improves the receive sensitivity by 2dB and increase the transmit power by 3dB.

Parameter range

0 - 1

Setting	Meaning
0	Boost mode disabled
1	Boost mode enabled

Default

1

CA (CCA Threshold)

Set or read the Clear Channel Assessment (CCA) threshold. Prior to transmitting a packet, the device performs a CCA to detect energy on the channel. If the device detects energy above the CCA threshold, it will not transmit the packet.

The **CA** parameter is measured in units of -dBm.

Note If device is operating in Europe, this value must be set to 0x34 to comply with EN 300 328 Listen Before Talk requirements. Alternatively the device can be set to **PL3** as explained in Europe.

Parameter range

0, 0x28 - 0x50

Default

0x0 (CCA disabled)

ED (Energy Detect)

Starts an energy detect scan. This command accepts an argument to specify the time in milliseconds to scan all channels. The device loops through all the available channels until the time elapses. It returns the maximal energy on each channel, a comma follows each value, and the list ends with a carriage return. The values returned reflect the energy level that **ED** detects in -dBm units.

Parameter range

0 - 0x3A98 (15 seconds)

Default

0xA (10 ms)

TP (Board Temperature)

The current module temperature in degrees Celsius in 8-bit two's compliment format. For example 0x1A = 26 °C, and 0xF6 = -10 °C.

Note This command is only available on the XBee-PRO device.

Parameter range

This is a read-only parameter

Default

N/A

%V (Voltage Supply Monitoring)

Displays the supply voltage of the device in mV units.

Parameter range

This is a read-only parameter

Default

N/A

%H (MAC Unicast One Hop Time)

The MAC unicast one hop time timeout in milliseconds. If you change the MAC parameters it can change this value.

The time to send a unicast between two nodes in the network should not exceed the product of the unicast one hop time (**%H**) and the number of hops between those two nodes.

Parameter range

[read-only]

Default

N/A

0x267

%8 (MAC Broadcast One Hop Time)

The time to send a broadcast between two nodes in the network should not exceed the product of the broadcast one hop time (**%8**) and the number of hops between those two nodes.

Parameter range

[read-only]

Default

N/A

DB (Last Packet RSSI)

Reports the RSSI in -dBm of the last received RF data packet. **DB** returns a hexadecimal value for the -dBm measurement.

For example, if **DB** returns 0x60, then the RSSI of the last packet received was -96 dBm.

DB only indicates the signal strength of the last hop. It does not provide an accurate quality measurement for a multihop link.

If the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module has been reset and has not yet received a packet, **DB** reports **0**.

This value is volatile (the value does not persist in the device's memory after a power-up sequence).

Parameter range

N/A

Default

0

UA (Unicasts Attempted Count)

The number of unicast transmissions expecting an acknowledgment (when **RR** > 0).

This value is volatile (the value does not persist in the device's memory after a power-up sequence).

Parameter range

0 - 0xFFFF

Default

0

GD (Good Packets Received)

This count increments when a device receives a good frame with a valid MAC header on the RF interface. Received MAC ACK packets do not increment this counter. Once the number reaches 0xFFFF, it does not count further events.

To reset the counter to any 16-bit unsigned value, append a hexadecimal parameter to the command.

This value is volatile (the value does not persist in the device's memory after a power-up sequence).

Parameter range

0 - 0xFFFF

Default

N/A (0 after reset)

BC (Bytes Transmitted)

The number of RF bytes transmitted. The firmware counts every byte of every packet, including MAC/PHY headers and trailers.

You can reset the counter to any 32-bit value by appending a hexadecimal parameter to the command.

Parameter range

0 - 0xFFFFFFFF

Default

N/A (0 after reset)

EA (MAC ACK Failure Count)

The number of unicast transmissions that time out awaiting a MAC ACK. This can be up to **RR** +1 timeouts per unicast when **RR** > 0.

This count increments whenever a MAC ACK timeout occurs on a MAC-level unicast. When the number reaches **0xFFFF**, the firmware does not count further events.

To reset the counter to any 16-bit unsigned value, append a hexadecimal parameter to the command. This value is volatile (the value does not persist in the device's memory after a power-up sequence).

Parameter range

0 - 0xFFFF

Default

N/A (0 after reset)

EC (CCA Failures)

Sets or displays the number of frames that were blocked and not sent due to CCA failures or receptions in progress. If CCA is disabled (**CA** is 0), then this count only increments for frames that are blocked due to receive in progress. When this count reaches its maximum value of 0xFFFF, it stops counting.

You can reset **EC** to 0 (or any other value) at any time to make it easier to track errors.

Parameter range

0 - 0xFFFF

Default

N/A (0 after reset)

TR (Transmission Failure Count)

This count increments whenever a MAC transmission attempt exhausts all MAC retries without ever receiving a MAC acknowledgment message from the destination node. Once the number reaches **0xFFFF**, it does not count further events.

To reset the counter to any 16-bit value, append a hexadecimal parameter to the command.

This value is volatile (the value does not persist in the device's memory after a power-up sequence).

Parameter range

0 - 0xFFFF

Default

N/A (0 after reset)

Network commands

The following commands are network commands.

CE (Routing / Messaging Mode)

The routing and messaging mode of the device.

A routing device repeats broadcasts. Indirect Messaging Coordinators do not transmit point-to-multipoint unicasts until an end device requests them. Setting a device as a poller causes it to regularly send polls to its Indirect Messaging Coordinator. Nodes can also be configured to route, or not route, multi-hop packets.

Parameter range

0 - 6

Parameter	Description	Routes packets
0	Standard router	Yes
1	Indirect message coordinator	Yes
2	Non-routing device	No
3	Non-routing coordinator	No
4	Indirect message poller	Yes
5	N/A	N/A
6	Non-routing poller	No

Default

0

TO (Transmit Options)

The bitfield that configures the transmit options for Transparent mode.

The device's transmit options. The device uses these options for all transmissions. API transmissions can override this using the TxOptions field in the API frame.

Parameter range

0 - 0xFF

Bit	Meaning	Description
6,7	Delivery method	b’00 = <invalid option> b’01 = Point-multipoint (0x40) b'10 = Directed Broadcast (0x80) b’11 = DigiMesh (0xC0)
5	Reserved	<set this bit to 0>
4	Reserved	<set this bit to 0>
3	Trace Route	Enable a Trace Route on all DigiMesh API packets
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 2: Set **TO** to **0xC1** to send transmissions using DigiMesh, with network acknowledgments disabled.

When you set **BR** to **0** the **TO** option has the DigiMesh and Repeater mode disabled automatically.

Default

0xC0

BH (Broadcast Hops)

The maximum transmission hops for broadcast data transmissions.

If you set **BH** greater than **NH**, the device uses the value of **NH**.

When working in API mode, the **Broadcast Radius** field in the API frame is used instead of this configuration.

Parameter range

0 - 0x20

Default

0

NH (Network Hops)

Sets or displays the maximum number of hops across the network. This parameter limits the number of hops. You can use this parameter to calculate the maximum network traversal time.

You must set this parameter to the same value on all nodes in the network.

Parameter range

1 - 0x20 (1 - 32 hops)

Default

7

NN (Network Delay Slots)

Set or read the maximum random number of network delay slots before rebroadcasting a network packet.

One network delay slot is approximately 13 ms.

Parameter range

1 - 0xA network delay slots

Default

3

MR (Mesh Unicast Retries)

Set or read the maximum number of network packet delivery attempts. If **MR** is non-zero, the packets a device sends request a network acknowledgment, and can be resent up to **MR**+1 times if the device does not receive an acknowledgment.

Changing this value dramatically changes how long a route request takes.

We recommend that you set this value to **1**.

If you set this parameter to **0**, it disables network ACKs. Initially, the device can find routes, but a route will never be repaired if it fails.

Parameter range

0 - 7 mesh unicast retries

Default

1

C8 (802.15.4 Compatibility)

Sets or displays the operational compatibility with a legacy DigiMesh 2.4 device (S1 or older versions of S2C firmware). This parameter should only be set when operating in a mixed network that contains XBee Series 1 or XBee S2C devices that cannot be updated to version 9002.

Parameter range

0, 4

Bit field:

Bit	Meaning	Setting	Description
2	TX compatibility	0	When encryption is enabled, AES Counter mode is used with a 256-bit key.
		1	When encryption is enabled AES ECB mode is used with a 128-bit key. This is compatible with legacy versions of DigiMesh 2.4.

Default

4

Addressing commands

The following AT commands are addressing commands.

SH (Serial Number High)

Displays the upper 32 bits of the unique IEEE 64-bit extended address assigned to the product family in the factory.

The 64-bit source address is always enabled. This value is read-only and it never changes.

Parameter range

0 - 0xFFFFFFFF [read-only]

Default

Set in the factory

SL (Serial Number Low)

Displays the lower 32 bits of the unique IEEE 64-bit RF extended address assigned to the product family in the factory.

The 64-bit source address is always enabled. This value is read-only and it never changes.

Parameter range

0 - 0xFFFFFFFF [read-only]

Default

Set in the factory

DH (Destination Address High)

Set or read the upper 32 bits of the 64-bit destination address. When you combine **DH** with **DL**, it defines the destination address that the device uses for transmissions in Transparent mode.

The destination address is also used for I/O sampling in both Transparent and API modes.

0x000000000000FFFF is the broadcast address. It is also used as the polling address when the device functions as end device.

Parameter range

0 - 0xFFFFFFFF

Default

0

DL (Destination Address Low)

Set or display the lower 32 bits of the 64-bit destination address. When you combine **DH** with **DL**, it defines the destination address that the device uses for transmissions in Transparent mode.

The destination address is also used for I/O sampling in both Transparent and API modes.

0x000000000000FFFF is the broadcast address. It is also used as the polling address when the device functions as end device.

Parameter range

0 - 0xFFFFFFFF

Default

0xFFFF

CI (Cluster ID)

The application layer cluster ID value. The device uses this value as the cluster ID for all data transmissions in Transparent mode and for all transmissions performed with the Transmit Request frame - 0x10 in API mode. In API mode, transmissions performed with the Explicit Addressing Command frame - 0x11 ignore this parameter.

If you set this value to 0x12 (loopback Cluster ID), the destination node echoes any transmitted packet back to the source device.

Parameter range

0 - 0xFFFF

Default

0x11 (Transparent data cluster ID)

Diagnostic commands

The following AT commands are diagnostic commands. Diagnostic commands are typically volatile and will not persist across a power cycle.

AG (Aggregator Support)

The **AG** command sends a broadcast through the network that has the following effects on nodes that receive the broadcast:

- The receiving node establishes a DigiMesh route back to the originating node, if there is space in the routing table.
- The **DH** and **DL** of the receiving node update to the address of the originating node if the **AG** parameter matches the current **DH**/**DL** of the receiving node.
- API-enabled devices with updated **DH** and **DL** send an Aggregate Addressing Update frame (0x8E) out the serial port.

Parameter range

Any 64-bit address

Default

N/A

DM (DigiMesh Options)

A bit field mask that you can use to enable or disable DigiMesh features.

Bit:

0: Disable aggregator updates. When set to 1, the device does not issue or respond to **AG** requests.

1: Disable Trace Route and NACK responses. When set to 1, the device does not generate or respond to Trace Route or NACK requests.

Parameter range

0 - 0x03 (bit field)

Default

0

DN (Discover Node)

Resolves an **NI** (Node identifier) string to a physical address (case sensitive).

The following events occur after **DN** discovers the destination node:

When **DN** is sent in Command mode:

1. The requesting node sets**DL** and **DH** to the address of the device with the matching **NI** string. The address selected (either 16-bit short address or 64-bit extended address) is chosen based on the destination device's **MY** command configuration.
2. The requesting node returns **OK** (or **ERROR**).
3. The requesting node exits Command mode to allow for immediate communication. If an ERROR is received, then Command mode does not exit.

When **DN** is sent as a local AT Command frame - 0x08:

1. The requesting node returns 0xFFFE followed by its 64-bit extended addresses in an AT Command Response frame - 0x88.
2. The device returns an **ERROR** message if it is given without a destination node (that is without a parameter) or if the given destination node does not respond within **N?** milliseconds.

Parameter range

20-byte ASCII string

Default

N/A

ND (Network Discover)

The command reports the following information after a jittered time delay:

SH<CR> (4 bytes)
SL<CR> (4 bytes)
DB<CR> (Contains the detected signal strength of the response in negative dBm units)
NI <CR> (variable, 0-20 bytes plus 0x00 character)
DEVICE_TYPE<CR> (1 byte: **0** = Coordinator, **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<CR> (1 byte. Optionally included based on **NO** settings.)
<CR>

If you send FN (Find Neighbors) in Command mode, after (**NT***100) ms + overhead time, the command ends by returning a carriage return, represented by <CR>.

ND accepts an NI (Node Identifier) as an argument. For more details, see Directed node discovery.

Broadcast an **ND** command to the network. If the command includes an optional node identifier string parameter, only those devices with a matching **NI** string respond without a random offset delay. If the command does not include a node identifier string parameter, all devices respond with a random offset delay.

The NT (Network Discovery Back-off) setting determines the range of the random offset delay. The NO (Network Discovery Options) setting sets options for the Node Discovery. For more information about options that affect the behavior of the **ND** command, see the description of **NO**.

WARNING! If the **NT** setting is small relative to the number of devices on the network, responses may be lost due to channel congestion. Regardless of the **NT** setting, because the random offset only mitigates transmission collisions, getting responses from all devices in the network is not guaranteed.

Parameter range

20-byte printable ASCII string

Default

[read-only]

FN (Find Neighbors)

Discovers and reports all devices found within immediate (1 hop) RF range. **FN** reports the following information for each device it discovers:

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** = Coordinator, **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 (Network Discovery Options) settings.)

RSSI OF LAST HOP<CR> (1 byte. Optionally included based on NO (Network Discovery Options) settings.)

<CR>

If you send the **FN** command in Command mode, after (**NT***100) ms + overhead time, the command ends by returning a carriage return, represented by <CR>.

If you send the **FN** command through a local AT Command (0x08) API frame, each response returns as a separate AT Command Response (0x88) or Remote Command Response (0x97) frame, respectively. The data consists of the bytes in the previous list without the carriage return delimiters. The **NI** string ends in a 0x00 null character.

FN accepts a **NI** (Node Identifier) as an argument.

See The FN (Find Neighbors) command for more details.

Parameter range

0 to 20 ASCII characters

Default

N/A

NI (Node Identifier)

Stores the node identifier string for a device, which is a user-defined name or description of the device. This can be up to 20 ASCII characters.

- XCTU prevents you from exceeding the string limit of 20 characters for this command. If you are using another software application to send the string, you can enter longer strings, but the software on the device returns an error.

Use the **ND** (Network Discovery) command with this string as an argument to easily identify devices on the network.

The **DN** command also uses this identifier.

Parameter range

A string of case-sensitive ASCII printable characters from 0 to 20 bytes in length. A carriage return or a comma automatically ends the command.

Default

0x20 (an ASCII space character)

NT (Network Discovery Back-off)

Sets or displays the network discovery back-off parameter for a device. This sets the maximum value for the random delay that the device uses to send network discovery responses.

The **ND** and **FN** commands use **NT**. The read-only **N?** command increases and decreases with **NT**.

Parameter range

0x20 - 0x2EE0 (x 100 ms)

Default

0x82 (13 seconds)

N? (Network Discovery Timeout)

The maximum response time, in milliseconds, for **ND** (Network Discovery) responses and **DN** (Discover Node) responses. The timeout is the sum of **NT** (Network Discovery Back-off Time) and the network propagation time.

Parameter range

This is a read-only parameter, however, its value increases or decreases as **NT** increases or decreases and you can modify **NT**.

Default

N/A

NO (Network Discovery Options)

Set or read the network discovery options value for the **ND** (Network Discovery) command on a particular device. The options bit field value changes the behavior of the **ND** command and what optional values the local device returns when it receives an **ND** command or API Node Identification Indicator (0x95) frame.

Use **NO** to suppress or include a self-response to **ND** (Node Discover) commands. When **NO** bit 1 = 1, a device performing a Node Discover includes a response entry for itself.

Parameter range

0x0 - 0x7 (bit field)

Bit field

Option	Description
0x01	Append the **DD** (Digi Device Identifier) value to **ND** responses or API node identification frames.
0x02	Local device sends **ND** response frame out the serial interface when **ND** is issued.
0x04	Append the RSSI of the last hop to **ND**, **FN**, and responses or API node identification frames.

Default

0x0

Security commands

The following AT commands are security commands.

EE (Encryption Enable)

Enables or disables Advanced Encryption Standard (AES) encryption.

Set this command parameter the same on all devices in a network.

Parameter range

0 - 1

Parameter	Description
0	Encryption Disabled
1	Encryption Enabled

Default

0

KY (AES Encryption Key)

The Link Key used for encryption and decryption. If C8 (802.15.4 Compatibility) bit 2 is cleared, encryption/decryption uses the 256 bits of the **KY** value (all 64 ASCII characters of the **KY** value). **C8** bit 2 sets encryption/decryption, and uses the last 32 ASCII characters of the 256-bit **KY** value entered.

This command is write-only. If you attempt to read **KY**, the device returns an **OK** status.

Set this command parameter the same on all devices in a network.

Parameter range

256-bit value (up to 32 hex bytes/64 ASCII bytes)

Default

0

Serial interfacing commands

The following AT commands are serial interfacing commands.

BD (Baud Rate)

To request non-standard baud rates with values above 0x80, you can use the Serial Console toolbar in XCTU to configure the serial connection (if the console is connected), or click the **Connect** button (if the console is not yet connected).

When you send non-standard baud rates to a device, it stores the closest interface data rate represented by the number in the **BD** register. Read the **BD** command by sending **ATBD** without a parameter value, and the device returns the value stored in the **BD** register.

Parameter range

Standard baud rates: 0x0 - 0x8

Non-standard baud rates: 0x4B0 - 0x3D090

Parameter	Description
0x0	1200 b/s
0x1	2400 b/s
0x2	4800 b/s
0x3	9600 b/s
0x4	19200 b/s
0x5	38400 b/s
0x6	57600 b/s
0x7	115200 b/s
0x8	230400 b/s
0x4B0 to 0x3D090 (non-standard baud rates)	

Default

0x03 (9600 b/s)

NB (Parity)

Set or read the serial parity settings for UART communications.

Parameter range

0x00 - 0x03

Parameter	Description
0x00	No parity
0x01	Even parity
0x02	Odd parity
0x03	Mark parity (forced high)

Default

0x00

RO (Packetization Timeout)

Set or read the number of character times of inter-character silence required before transmission begins when operating in Transparent mode.

Set **RO** to **0** to transmit characters as they arrive instead of buffering them into one RF packet.

Parameter range

0 - 0xFF (x character times)

Default

3

FT (Flow Control Threshold)

Set or display the flow control threshold.

The device de-asserts $\overline{\text{CTS}}$ and/or send XOFF when **FT** bytes are in the UART receive buffer. It re-asserts $\overline{\text{CTS}}$ when less than **FT**-16 bytes are in the UART receive buffer.

Parameter range

0x07 - 0x66 bytes

Default

0x51

AP (API Enable)

Set or read the API mode setting. The device can format the RF packets it receives into API frames and send them out the serial port.

When you enable API, you must format the serial data as API frames because Transparent operating mode is disabled.

Parameter range

0 - 2

Parameter	Description
0	API disabled (operate in Transparent mode)
1	API enabled
2	API enabled (with escaped control characters)

Default

0

AO (API Options)

The API data frame output format for RF packets received.

Use **AO** to enable different API output frames.

Parameter range

0 - 2

Parameter	Description
0	API Rx Indicator - 0x90, this is for standard data frames.
1	API Explicit Rx Indicator - 0x91, this is for Explicit Addressing data frames.

Default

0

I/O settings commands

The following AT commands are I/O settings commands.

CB (Commissioning Pushbutton)

Use **CB** to simulate commissioning pushbutton presses in software.

Set the parameter value to the number of button presses that you want to simulate. For example, send **CB1** to perform the action of pressing the Commissioning Pushbutton once.

Parameter range

1, 4

Default

N/A

D0 (DIO0/AD0)

Sets or displays the DIO0/AD0 configuration (TH pin 20/SMT pin 33).

Parameter range

0 - 5

Parameter	Description
0	Disabled
1	Commissioning Pushbutton
2	ADC
3	Digital input
4	Digital output, low
5	Digital output, high

Default

1

D1 (DIO1/AD1)

Sets or displays the DIO1/AD1 configuration (TH pin 19/SMT pin 32).

Parameter range

0 - 5

Parameter	Description
0	Disabled
1	SPI_$\overline{\text{ATTN}}$ for the through-hole device N/A for the surface-mount device
2	ADC
3	Digital input
4	Digital output, low
5	Digital output, high

Default

0

D2 (DIO2/AD2)

Sets or displays the DIO2/AD2 configuration (TH pin 18/SMT pin 31).

Parameter range

0 - 5

Parameter	Description
0	Disabled
1	SPI_CLK for through-hole devices N/A for surface-mount devices
2	ADC
3	Digital input
4	Digital output, low
5	Digital output, high

Default

0

D3 (DIO3/AD3)

Sets or displays the DIO3/AD3 configuration (TH pin 17/SMT pin 30).

Parameter range

0 - 5

Parameter	Description
0	Disabled
1	SPI_$\overline{\text{SSEL}}$ for the through-hole device N/A for surface-mount device
2	ADC
3	Digital input
4	Digital output, low
5	Digital output, high

Default

0

D4 (DIO4)

Sets or displays the DIO4 configuration (TH pin 11/SMT pin 24).

Parameter range

0, 2 - 5

0 - 5

Parameter	Description
0	Disabled
1	SPI_MOSI for the through-hole device N/A for the surface-mount device
2	DI04
3	Digital input
4	Digital output, low
5	Digital output, high

Default

0

D5 (DIO5/ASSOCIATED_INDICATOR)

Sets or displays the DIO5/ASSOCIATED_INDICATOR configuration (TH pin 15/SMT pin 28).

Parameter range

0 - 5

Parameter	Description
0	Disabled
1	Associate LED indicator - blinks when associated
2	DIO5/ASSOCIATED_INDICATOR
3	Digital input
4	Digital output, default low
5	Digital output, default high

Default

1

D6 (DIO6/RTS)

Sets or displays the DIO6/$\overline{\text{RTS}}$ configuration (TH pin 16/SMT pin 29).

Parameter range

0, 1, 3 - 5

Parameter	Description
0	Disabled

Parameter	Description
1	$\overline{\text{RTS}}$ flow control
2	N/A
3	Digital input
4	Digital output, low
5	Digital output, high

Default

0

D7 (DIO7/CTS)

Sets or displays the DIO7/$\overline{\text{CTS}}$ configuration (TH pin 12/SMT pin 25).

Parameter range

0, 1, 3 - 7

Parameter	Description
0	Disabled
1	$\overline{\text{CTS}}$ flow control
2	N/A
3	Digital input
4	Digital output, low
5	Digital output, high
6	RS-485 Tx enable, low Tx (0 V on transmit, high when idle)
7	RS-485 Tx enable high, high Tx (high on transmit, 0 V when idle)

Default

0x1

D8 (DIO8/DTR/SLEEP_REQUEST)

Sets or displays the DIO8/$\overline{\text{DTR}}$/SLP_RQ configuration (TH pin 9/SMT pin 10).

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module does not support sleep. The SLEEP_REQUEST option is provided for compatibility purposes and does not affect the device.

Parameter range

0, 1, 3 - 5

Parameter	Description
0	Disabled
1	SLEEP_REQUEST input
2	N/A
3	Digital input
4	Digital output, low
5	Digital output, high

Default

1

D9 (ON_SLEEP)

Sets or displays the ON/$\overline{\text{SLEEP}}$ configuration (TH pin 13/SMT pin 26).

Parameter range

0, 1, 3 - 5

Parameter	Description
0	Disabled
1	ON/$\overline{\text{SLEEP}}$ output
2	N/A
3	Digital input
4	Digital output, low
5	Digital output, high

Default

1

P0 (DIO10/RSSI/PWM0 Configuration)

Sets or displays the RSSI/PWM0 configuration (TH pin 6/SMT pin 7).

When configured as a PWM output, you can use **M0** to set the PWM duty cycle.

Parameter range

0 - 5

Parameter	Description
0	Disabled

Parameter	Description
1	RSSI PWM output
2	PWM0 output. Value is controlled by **M0** parameter or by I/O line passing.
3	Digital input
4	Digital output, low
5	Digital output, high

Default

1

P1 (DIO11/PWM1 Configuration)

Sets or displays the DIO11/PWM1 configuration (TH pin 7/SMT pin 8).

Parameter range

0 - 5

Parameter	Description
0	Disabled
1	N/A
2	PWM1 output. Value is controlled by **M1** parameter or by I/O line passing
3	Digital input
4	Digital output, low
5	Digital output, high

Default

0

P2 (DIO12/SPI_MISO Configuration)

Sets or displays the DIO12/SPI_MISO configuration (TH pin 4/SMT pin 5).

Parameter range

1, 3 - 5

Parameter	Description
0	Disabled
1	SPI_MISO for the through-hole device N/A for the surface-mount device

Parameter	Description
2	N/A
3	Digital input
4	Digital output, low
5	Digital output, high

Default

0

P5 (SPI_MISO)

Sets or displays the DIO15/SPI_MISO configuration.

This only applies to surface-mount devices.

Parameter range

0, 1

Parameter	Description
0	Disabled
1	SPI_MISO

Default

1

P6 (SPI_MOSI Configuration)

Sets or displays the DIO16/SPI_MOSI configuration.

This only applies to surface-mount devices.

Parameter range

0, 1

Parameter	Description
0	Disabled
1	SPI_MOSI

Default

1

P7 (SPI_SSEL)

Sets or displays the DIO17/SPI_$\overline{\text{SSEL}}$ configuration.

This only applies to surface-mount devices.

Parameter range

1, 2

Parameter	Description
0	Disabled
1	SPI_$\overline{\text{SSEL}}$

Default

1

P8 (SPI_SCLK)P8 (SPI_SCLK)

Sets or displays the SPI_SCLK configuration.

This only applies to surface-mount devices.

Parameter range

1, 2

Parameter	Description
0	Disabled
1	SPI_SCLK

Default

1

P9 (SPI_ATTN)

Sets or displays the SPI_$\overline{\text{ATTN}}$ configuration.

This only applies to surface-mount devices.

Parameter range

0, 1

Parameter	Description
0	Disabled
1	SPI_$\overline{\text{ATTN}}$

Default

1

PD (Pull Up/Down Direction)

The resistor pull direction bit field (1 = pull-up, 0 = pull-down) for corresponding I/O lines that are set by the **PR** command.

See PR (Pull-up/Down Resistor Enable) for the bit mappings.

Parameter range

0x0 - 0x7FFF

Default

0x1FFF

PR (Pull-up/Down Resistor Enable)

PR and **PD** only affect lines that are configured as digital inputs or disabled.

The following table defines the bit-field map for **PR** and **PD** commands.

Bit	I/O line
0	DIO4 (pin 11 for through-hole, pin 24 for surface-mount)
1	DIO3/AD3 (pin 17 for through-hole, pin 30 for surface-mount)
2	DIO2/AD2 (pin 18 for through-hole, pin 31 for surface-mount)
3	DIO1/AD1 (pin 19 for through-hole, pin 32 for surface-mount)
4	DIO0/AD0 (pin 20 for through-hole, pin 33 for surface-mount)
5	DIO6/$\overline{\text{RTS}}$ (pin 16 for through-hole, pin 29 for surface-mount)
6	DIO8/SLEEP_REQUEST (pin 9 for through-hole, pin 10 for surface-mount)
7	DIO14/DIN/$\overline{\text{CONFIG}}$ (pin 3 for through-hole, pin 4 for surface-mount)
8	DIO5/ASSOCIATE (pin 15 for through-hole, pin 28 for surface-mount)
9	DIO9/ON_$\overline{\text{SLEEP}}$(pin 13 for through-hole, pin 26 for surface-mount)
10	DIO12/SPI_MISO (pin 4 for through-hole), DIO12 (pin 5 for surface-mount)
11	DIO10/RSSI/PWM0 (pin 6 for through-hole, pin 7 for surface-mount)
12	DIO11/PWM1 (pin 7 for through-hole, pin 8 for surface-mount)
13	DIO7/$\overline{\text{CTS}}$ (pin 12 for through-hole, pin 25 for surface-mount)
14	DOUT (pin 2)

Parameter range

0 - 0x7FFF (bit field)

Default

0x1FFF

M0 (PWM0 Duty Cycle)

The duty cycle of the PWM0 line (TH pin 6/SMT pin 7).

If the **IA** (I/O Input Address) parameter is correctly set and **P0** is configured as PWM0 output, incoming AD0 samples automatically modify the PWM0 value.

See PT (PWM Output Timeout).

To configure the duty cycle of PWM0:

1. Enable PWM0 output (**P0** = **2**).
2. Change **M0** to the desired value.
3. Apply settings (use **CN** or **AC**).

The PWM period is 64 µs and there are 0x03FF (1023 decimal) steps within this period. When **M0** = **0** (0% PWM), 0x01FF (50% PWM), 0x03FF (100% PWM), and so forth.

Parameter range

0 - 0x3FF

Default

0

M1 (PWM1 Duty Cycle)

The duty cycle of the PWM1 line (TH pin 7/SMT pin 8).

If the **IA** (I/O Input Address) parameter is correctly set and **P1** is configured as PWM1 output, incoming AD1 samples automatically modify the PWM1 value. See PT (PWM Output Timeout).

To configure the duty cycle of PWM1:

1. Enable PWM1 output (**P1** = 2).
2. Change **M1** to the desired value.
3. Apply settings (use **CN** or **AC**).

The PWM period is 64 µs and there are 0x03FF (1023 decimal) steps within this period. When **M1** = **0** (0% PWM), 0x01FF (50% PWM), 0x03FF (100% PWM), and so forth.

Parameter range

0 - 0x3FF

Default

0

LT (Associate LED Blink Time)

Set or read the Associate LED blink time. If you use the **D5** command to enable the Associate LED functionality (DIO5/Associate pin), this value determines the on and off blink times for the LED when the device has joined the network.

If **LT** = **0**, the device uses the default blink rate of 250 ms.

For all other **LT** values, the firmware measures **LT** in 10 ms increments.

Set or read the Associate LED blink time. If you use the **D5** command to enable the Associate LED functionality (DIO5/Associate pin), this value determines the on and off blink times for the LED. The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module does not use network authentication or synchronized sleep support, so the Associate LED steadily blinks regardless of the current network status.

Parameter range

0x14 - 0xFF (x 10 ms)

Default

0

RP (RSSI PWM Timer)

The PWM timer expiration in 0.1 seconds. **RP** sets the duration of pulse width modulation (PWM) signal output on the RSSI pin. The signal duty cycle updates with each received packet and shuts off when the timer expires.

When **RP** = **0xFF**, the output is always on.

Parameter range

0 - 0xFF (x 100 ms)

Default

0x28 (four seconds)

I/O line passing commands

The following AT commands are I/O line passing commands.

I/O Line Passing allows the digital and analog inputs of a remote device to affect the corresponding outputs of the local device.

You can perform Digital Line Passing on any of the Digital I/O lines. Digital Inputs directly map to Digital Outputs of each digital pin.

Analog Line Passing can be performed only on the first two ADC lines:

- ADC0 corresponds with PWM0
- ADC1 corresponds with PWM1

IA (I/O Input Address)

The source address of the device to which outputs are bound. Setting all bytes to 0xFF disables I/O line passing. Setting **IA** to 0xFFFF allows any I/O packet addressed to this device (including broadcasts) to change the outputs.

The source address of the device to which outputs are bound. If an I/O sample is received from the address specified, any pin that is configured as a digital output or PWM changes its state to match that of the I/O sample.

Set **IA** to 0xFFFFFFFFFFFFFFFF to disable I/O line passing.

Set **IA** to 0xFFFF to allow any I/O packet addressed to this device (including broadcasts) to change the outputs.

Parameter range

0 - 0xFFFF FFFF FFFF FFFF

Default

0xFFFFFFFFFFFFFFFF (I/O line passing disabled)

IU (Send I/O Sample to Serial Port)

Indicates whether or not I/O samples should be sent to the serial port. 0 suppresses output; 1 allows output (only if the device is in API mode).

Enable or disable the serial output of received I/O sample data when I/O line passing is enabled. **IU** only affects the device's behavior when **IA** is set to a non-default value.

When **IU** is enabled, any received I/O sample data is sent out the UART/SPI interface using an API frame. Sample data is only generated if the local device is operating in API mode (**AP** = 1 or 2).

Parameter range

0 - 1

Parameter	Description
0	Disabled
1	Enabled

Default

1

T0 (D0 Timeout)

Specifies how long pin D0 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T1 (D1 Output Timeout)

Specifies how long pin D1 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T2 (D2 Output Timeout)

Specifies how long pin D2 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T3 (D3 Output Timeout)

Specifies how long pin D0 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 3, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T4 (D4 Output Timeout)

Specifies how long pin D4 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T5 (D5 Output Timeout)

Specifies how long pin D5 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T6 (D6 Output Timeout)

Specifies how long pin D6 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T7 (D7 Output Timeout)

Specifies how long pin D7 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T8 (D8 Timeout)

Specifies how long pin D8 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T9 (D9 Timeout)

Specifies how long pin D9 holds a given value (due to I/O line passing) before it reverts to configured value. If set to **0**, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

Q0 (P0 Timeout)

Specifies how long pin **P0** holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

Q1 (P1 Timeout)

Specifies how long pin P1 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

Q2 (P2 Timeout)

Specifies how long pin P2 holds a given value (due to I/O line passing) before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

PT (PWM Output Timeout)

Specifies how long both PWM outputs (**P0**, **P1**) output a given PWM signal (due to I/O line passing) before it reverts to the configured value (**M0**/**M1**). If set to 0, there is no timeout. This timeout only affects these pins when they are configured as PWM output.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0xFF

I/O sampling commands

The following AT commands configure I/O sampling parameters.

IC (DIO Change Detect)

Set or read 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 you enable a pin as a digital I/O, use the **IC** command to force an immediate I/O sample transmission when the DIO state changes. If sleep is enabled, the edge transition must occur during a wake period to trigger a change detect.

The data transmission contains only DIO data.

IC is a bitmask you can use to enable or disable edge detection on individual digital I/O lines. Only DIO0 through DIO12 can be sampled using a Change Detect.

Set unused bits to 0.

Bit	I/O line	Surface-mount pin	Through-hole pin
0	DIO0	33	20
1	DIO1	32	19
2	DIO2	31	18
3	DIO3	30	17
4	DIO4	24	11
5	DIO5	28	15
6	DIO6	29	16
7	DIO7	25	12

Bit	I/O line	Surface-mount pin	Through-hole pin
8	DIO8	10	9
9	DIO9	26	13
10	DIO10	7	6
11	DIO11	8	7
12	DIO12	5	4

Parameter range

0 - 0x1FFF

Default

0

IF (Sleep Sample Rate)

Set or read the number of sleep cycles that must elapse between periodic I/O samples. This allows the firmware to take I/O samples only during some wake cycles. During those cycles, the firmware takes I/O samples at the rate specified by **IR**.

Parameter range

1 - 0xFF

Default

1

IR (Sample Rate)

Set or read the I/O sample rate to enable periodic sampling. When set, this parameter causes the device to sample all enabled DIO and ADC at a specified interval.

To enable periodic sampling, set **IR** to a non-zero value, and enable the analog or digital I/O functionality of at least one device pin (see D0 (DIO0/AD0) -D9 (ON_SLEEP), P0 (DIO10/RSSI/PWM0 Configuration)- P2 (DIO12/SPI_MISO Configuration).

WARNING! If you set **IR** to 1 or 2, the device will not keep up and many samples will be lost.

Set or read the I/O sample rate to enable periodic sampling.

When set, this parameter causes the device to sample all enabled digital I/O and analog inputs at a specified interval. Samples will be sent to the address specified by the **DH** and **DL** commands. The target device must be operating in API mode in order to output the received sample data.

To enable periodic sampling, set **IR** to a non-zero value, and enable the analog or digital I/O functionality of at least one device pin (**D0** – **D9**, **P0** – **P9**).

Parameter range

0 - 0xFFFF (x 1 ms)

Default

0

IS (Force Sample)

Forces a read of all enabled digital and analog input lines. The data is returned through the UART or SPI.

See Monitor I/O lines.

Parameter range

N/A

Default

N/A

Sleep commands

The following AT commands are sleep commands.

SM (Sleep Mode)

Sets or displays the sleep mode of the device.

Normal mode is always awake. Pin sleep modes allow you to wake the device with the SLEEP_ REQUEST line. Asynchronous cyclic mode sleeps for **SP** time and briefly wakes, checking for activity. The device does not support synchronous sleep.

Parameter range

0 - 5

Parameter	Description
0	No sleep (disabled)
1	Pin sleep
2	Reserved
3	Reserved
4	Cyclic Sleep Remote
5	Cyclic Sleep Remote with pin wakeup

Default

0

SO (Sleep Options)

Set or read the sleep options bit field of a device. This command is a bitmask.

You can set or clear any of the available sleep option bits.

Parameter range

0x0000, 0x0100

Bit	Option
8	Always wake for **ST** time

Default

0

SN (Number of Cycles Between ON_SLEEP)

Set or read 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. This allows external circuitry to sleep longer than the **SP** time.

Parameter range

1 - 0xFFFF

Default

1

Example

Set to 1 to set ON_SLEEP high after each **SP** time (default).

If **SN** = 3, the ON_SLEEP line asserts only every third wakeup; **SN** = 9, every ninth wakeup; and so forth.

SP (Sleep Time)

Sets or displays the device's sleep time. This command defines the amount of time the device sleeps 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 end device. The coordinator will hold the message for (2.5 * **SP**).

Parameter range

0x1 - 0x15F900 (x 10 ms)

Default

0xC8

ST (Wake Time)

Sets or displays the wake time of the device.

For devices in cyclic sleep, **ST** defines the amount of time that a device stays awake after it receives RF or serial data.

Parameter range

0x1 - 0x36EE80 (x 1 ms)

Default

0x7D0 (3 seconds)

WH (Wake Host Delay)

Sets or displays the wake host timer value. You can use **WH** to give a sleeping host processor sufficient time to power up after the device asserts the ON_SLEEP line.

If you set **WH** to a non-zero value, this timer specifies a time in milliseconds that the device delays after waking from sleep before sending data out the UART or transmitting an I/O sample. If the device receives serial characters, the **WH** timer stops immediately.

Parameter range

0 - 0xFFFF (x 1 ms)

Default

0

Command mode options

The following commands are Command mode option commands.

CC (Command Character)

The character value the device uses to enter Command mode.

The default value (**0x2B**) is the ASCII code for the plus (**+**) character. You must enter it three times within the guard time to enter Command mode. To enter Command mode, there is also a required period of silence before and after the command sequence characters of the Command mode sequence (**GT** + **CC** + **GT**). The period of silence prevents inadvertently entering Command mode.

Parameter range

0 - 0xFF

Recommended: 0x20 - 0x7F (ASCII)

Default

0x2B (the ASCII plus character: **+**)

CT (Command Mode Timeout)

Sets or displays the Command mode timeout parameter. If a device does not receive any valid commands within this time period, it returns to Transparent mode or API mode.

Parameter range

2 - 0x1770 (x 100 ms)

Default

0x64 (10 seconds)

CN (Exit Command Mode)

Immediately exits Command Mode and applies pending changes.

Parameter range

N/A

Default

N/A

GT (Guard Times)

Set the required period of silence before and after the command sequence characters of the Command mode sequence (**GT** + **CC** + **GT**). The period of silence prevents inadvertently entering Command mode.

Parameter range

0x2 - 0xCE4 (x 1 ms)

Default

0x3E8 (one second)

Firmware version/information commands

The following AT commands are firmware commands.

VL (Version Long)

Shows detailed version information including the application build date and time.

Parameter range

N/A

Default

N/A

VR (Firmware Version)

Reads the firmware version on a device.

Parameter range

0x9000 - 0x90FF [read-only]

Default

Set in the factory

HV (Hardware Version)

Display the hardware version number of the device.

Parameter range

0 - 0xFFFF [read-only]

Default

Set in firmware

DD (Device Type Identifier)

Stores the Digi device type identifier value. Use this value to differentiate between multiple XBee devices.

If you change **DD**, RE (Restore Defaults) will not restore defaults. The only way to get **DD** back to default values is to explicitly set it to defaults.

Parameter range

0 - 0xFFFFFFFF

Default

0x50000

NP (Maximum Packet Payload Bytes)

Reads the maximum number of RF payload bytes that you can send in a transmission.

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module firmware returns a fixed number of bytes: 0x49 = 73 bytes.

Parameter range

0 - 0xFFFF (bytes) [read-only]

Default

N/A

CK (Configuration CRC)

Displays the cyclic redundancy check (CRC) of the current AT command configuration settings.

This command allows you to detect an unexpected configuration change on a device. Use the code that the device returns to determine if a node has the configuration you want.

After a firmware update this command may return a different value.

Parameter range

0 - 0xFFFF

Default

N/A

Operate in API mode

API mode overview

As an alternative to Transparent operating mode, you can use API operating mode. API mode provides a structured interface where data is communicated through the serial interface in organized packets and in a determined order. This enables you to establish complex communication between devices without having to define your own protocol. The API specifies how commands, command responses and device status messages are sent and received from the device using the serial interface or the SPI interface.

We may add new frame types to future versions of firmware, so build the ability to filter out additional API frames with unknown frame types into your software interface.

API frame specifications

The firmware supports two API operating modes: without escaped characters and with escaped characters. Use the **AP** command to enable either mode. To configure a device to one of these modes, set the following **AP** parameter values:

AP command setting	Description
AP = 0	Transparent operating mode, UART serial line replacement with API modes disabled. This is the default option.
AP = 1	API operation.
AP = 2	API operation with escaped characters (only possible on UART).

The API data frame structure differs depending on what mode you choose.

The firmware silently discards any data it receives prior to the start delimiter. If the device does not receive the frame correctly or if the checksum fails, the device discards the frame.

API operation (AP parameter = 1)

We recommend this API mode for most applications. The following table shows the data frame structure when you enable this mode:

Frame fields	Byte	Description
Start delimiter	1	0x7E
Length	2 - 3	Most Significant Byte, Least Significant Byte
Frame data	4 - n	API-specific structure
Checksum	n + 1	1 byte

API operation-with escaped characters (AP parameter = 2)

Set API to 2 to allow escaped control characters in the API frame. Due to its increased complexity, we only recommend this API mode in specific circumstances. API 2 may help improve reliability if the serial interface to the device is unstable or malformed frames are frequently being generated.

When operating in API 2, if an unescaped 0x7E byte is observed, it is treated as the start of a new API frame and all data received prior to this delimiter is silently discarded. For more information on using this API mode, refer to the following knowledge base article:

http://knowledge.digi.com/articles/Knowledge_Base_Article/Escaped-Characters-and-API-Mode-2

The following table shows the structure of an API frame with escaped characters:

Frame fields	Byte	Description	
Start delimiter	1	0x7E	
Length	2 - 3	Most Significant Byte, Least Significant Byte	Characters escaped if needed
Frame data	4 - n	API-specific structure	
Checksum	n + 1	1 byte	

Escape characters

When sending or receiving a UART data frame, you must escape (flag) specific data values 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. If not escaped, 0x11 and 0x13 are sent as is.

Data bytes that need to be escaped:

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

Example - Raw UART 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 previous 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.

Start delimiter

This field indicates the beginning of a frame. It is always 0x7E. This allows the device to easily detect a new incoming frame.

Length

The length field specifies the total number of bytes included in the frame's data field. Its two-byte value excludes the start delimiter, the length, and the checksum.

Frame data

This field contains the information that a device receives or will transmit. The structure of frame data depends on the purpose of the API frame:

			Frame data								
Start delimiter	Length		Frame type	Data							Checksum
1	2	3	4	5	6	7	8	9	...	n	n+1
0x7E	MSB	LSB	API frame type	Data							Single byte

- **Frame type** is the API frame type identifier. It determines the type of API frame and indicates how the Data field organizes the information.
- **Data** contains the data itself. This information and its order depend on the what type of frame that the Frame type field defines.

Checksum is the last byte of the frame and helps test data integrity. It is calculated by taking the hash sum of all the API frame bytes that came before it, except the first three bytes (start delimiter and length).

The device does not process frames sent through the serial interface with incorrect checksums, and ignores their data.

Calculate and verify checksums

To calculate the checksum of an API frame:

1. Add all bytes of the packet, except the start delimiter 0x7E and the length (the second and third bytes).
2. Keep only the lowest 8 bits from the result.
3. Subtract this quantity from 0xFF.

To verify the checksum of an API frame:

1. Add all bytes including the checksum; do not include the delimiter and length.
2. If the checksum is correct, the last two digits on the far right of the sum equal 0xFF.

Example

Consider the following sample data packet: **7E 00 0A 01 01 50 01 00 48 65 6C 6C 6F B8**+

Byte(s)	Description
7E	Start delimiter
00 0A	Length bytes
01	API identifier
01	API frame ID
50 01	Destination address low
00	Option byte
48 65 6C 6C 6F	Data packet
B8	Checksum

To calculate the check sum you add all bytes of the packet, excluding the frame delimiter **7E** and the length (the second and third bytes):

7E 00 0A 01 01 50 01 00 48 65 6C 6C 6F B8

Add these hex bytes:

01 + 01 + 50 + 01 + 00 + 48 + 65 + 6C + 6C + 6F = 247

Now take the result of 0x247 and keep only the lowest 8 bits which in this example is 0xC4 (the two far right digits). Subtract 0x47 from 0xFF and you get 0x3B (0xFF - 0xC4 = 0x3B). 0x3B is the checksum for this data packet.

If an API data packet is composed with an incorrect checksum, the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module will consider the packet invalid and will ignore the data.

To verify the check sum of an API packet add all bytes including the checksum (do not include the delimiter and length) and if correct, the last two far right digits of the sum will equal FF.

01 + 01 + 50 + 01 + 00 + 48 + 65 + 6C + 6C + 6F + B8 = 2FF

Escaped characters in API frames

If operating in API mode with escaped characters (**AP** parameter = 2), when sending or receiving a serial 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'ed with 0x20).

The following data bytes need to be escaped:

- 0x7E: start delimiter
- 0x7D: escape character
- 0x11: XON
- 0x13: XOFF

To escape a character:

1. Insert 0x7D (escape character).
2. Append it with the byte you want to escape, XOR'ed with 0x20.

In API mode with escaped characters, the length field does not include any escape characters in the frame and the firmware calculates the checksum with non-escaped data.

API frame exchanges

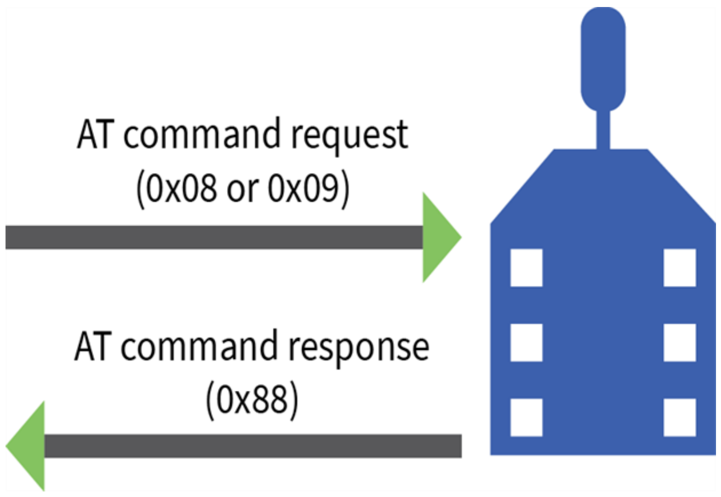
Every outgoing API frame has a corresponding response (or ACK) frame that indicates the success or failure of the outgoing API frame. This section details some of the common API exchanges that occur.

You can use the Frame ID field to correlate between the outgoing frames and associated responses.

Note Using a Frame ID of 0 disables responses, which can reduce network congestion for non-critical transmissions.

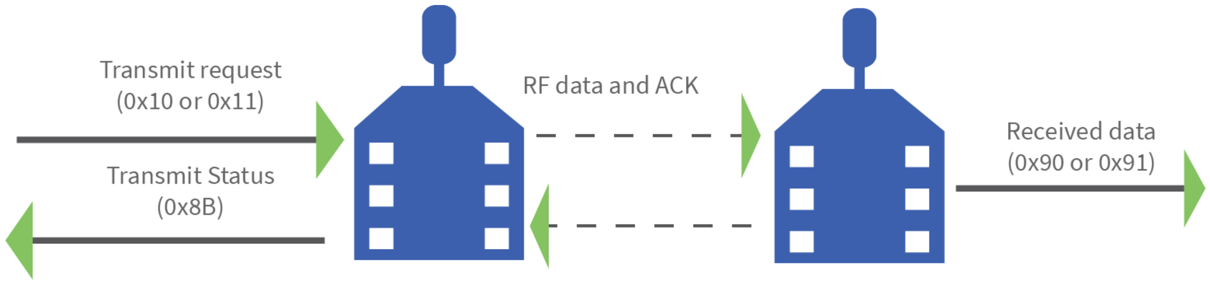
AT commands

The following image shows the API frame exchange that takes place at the UART when you send a 0x08 AT Command Request or 0x09 AT Command-Queue Request to read or set a device parameter. To disable the 0x88 AT Command Response, set the frame ID to 0 in the request.



Transmit and Receive RF data

The following image shows the API exchanges that take place on the serial interface when a device sends a 0x10, or 0x11 Transmit Request to another device.

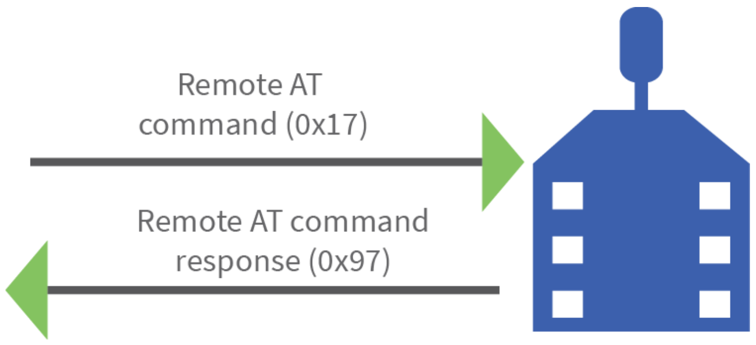


The device sends the 0x8B Transmit Status frame at the end of a data transmission unless you set the frame ID to 0 in the transmit request. If the packet cannot be delivered to the destination, the 0x8B Transmit Status frame indicates the cause of failure.

Use the **AP** command to choose the type of data frame you want to receive, either a (0x90) Receive Packet or a (0x91) Explicit Rx Indicator frame.

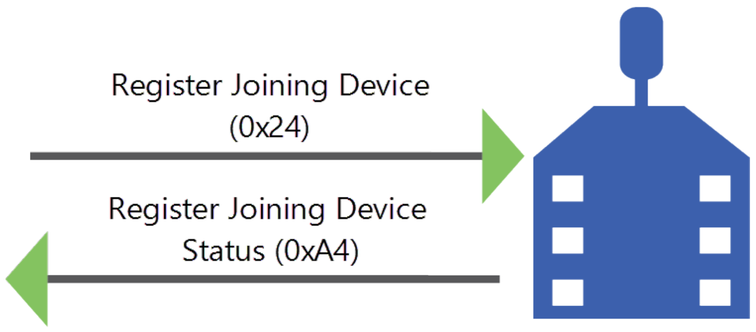
Remote AT commands

The following image shows the API frame exchanges that take place on the serial interface when you send a 0x17 Remote AT Command frame The 0x97 Remote AT Command Response is always generated and you can use it to identify if the remote device successfully received and applied the command.



Device Registration

The following image shows the API frame exchanges that take place at the serial interface when registering a joining device to a trust center.


Register Joining Device (0x24)
Register Joining Device Status (0xA4)

Frame descriptions

The following sections describe the API frames.

AT Command frame - 0x08

Description

Use this frame to query or set device parameters on the local device. This API command applies changes after running the command. You can query parameter values by sending the 0x08 AT Command frame with no parameter value field (the two-byte AT command is immediately followed by the frame checksum).

A 0x8B response frame is populated with the parameter value that is currently set on the device.

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x08
AT command	5-6	Command name: two ASCII characters that identify the AT command.
Parameter value	7-n	If present, indicates the requested parameter value to set the given register. If no characters are present, it queries the register.

Example

The following example illustrates an AT Command frame when you query the device's **NH** parameter value.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x04
Frame type	3	0x08
Frame ID	4	0x52

Frame data fields	Offset	Example
AT command	5	0x4E (N)
	6	0x48 (H)
Parameter value (**NH**2 = two network hops)	7	0x02
Checksum	8	0x0D

AT Command - Queue Parameter Value frame - 0x09

Description

This frame allows you to query or set device parameters. In contrast to the AT Command (0x08) frame, this frame queues new parameter values and does not apply them until you issue either:

- The **AT** Command (0x08) frame (for API type)
- The **AC** command

When querying parameter values, the 0x09 frame behaves identically to the 0x08 frame. The device returns register queries immediately and not does not queue them. The response for this command is also an **AT** Command Response frame (0x88).

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x09
Frame ID	4	Identifies the data frame for the host to correlate with a subsequent ACK. If set to **0**, the device does not send a response.
AT command	5-6	Command name: two ASCII characters that identify the AT command.
Parameter value	7-n	If present, indicates the requested parameter value to set the given register. If no characters are present, queries the register.

Example

The following example sends a command to change the baud rate (**BD**) to 115200 baud, but does not apply the changes immediately. The device continues to operate at the previous baud rate until you apply the changes.

Note In this example, you could send the parameter as a zero-padded 2-byte or 4-byte value.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x05
Frame type	3	0x09

Frame data fields	Offset	Example
Frame ID	4	0x01
AT command	5	0x42 (B)
	6	0x44 (D)
Parameter value (**BD**7 = 115200 baud)	7	0x07
Checksum	8	0x68

Transmit Request frame - 0x10

Description

This frame causes the device to send payload data as an RF packet to a specific destination.

- For broadcast transmissions, set the 64-bit destination address to **0x000000000000FFFF** .
- For unicast transmissions, set the 64 bit address field to the address of the desired destination node.
- Set the reserved field to **0xFFFE**.
- Query the **NP** command to read the maximum number of payload bytes.

You can set the broadcast radius from **0** up to **NH**. If set to **0**, the value of **NH** specifies the broadcast radius (recommended). This parameter is only used for broadcast transmissions.

You can read the maximum number of payload bytes with the **NP** command.

Format

The following table provides the contents of the frame. For details on the frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x10
Frame ID	4	Identifies the data frame for the host to correlate with a subsequent ACK. If set to **0**, the device does not send a response.
64-bit destination address	5-12	MSB first, LSB last. Set to the 64-bit address of the destination device. Broadcast = 0x000000000000FFFF
Reserved	13-14	Set to 0xFFFE.
Broadcast radius	15	Sets the maximum number of hops a broadcast transmission can occur. If set to 0, the broadcast radius is set to the maximum hops value.
Transmit options	16	See the Transmit Options table below. Set all other bits to 0.
RF data	17-n	Up to **NP** bytes per packet. Sent to the destination device.

Transmit Options bit field

Bit field:

Bit	Meaning	Description
0	Disable ACK	Disable acknowledgments on all unicasts
1	Disable RD	Disable Route Discovery on all DigiMesh unicasts
2	NACK	Enable unicast NACK messages on all DigiMesh API packets
3	Trace route	Enable a unicast Trace Route on all DigiMesh API packets
4	Reserved	<set this bit to 0>
5	Reserved	<set this bit to 0>
6,7	Delivery method	b’00 = <invalid option> b’01 - Point-multipoint (0x40) b’10 = Directed Broadcast (0x80) b’11 = DigiMesh (0xC0)

Example

The example shows how to send a transmission to a device if you disable escaping (**AP** = 1), with destination address 0x0013A200 400A0127, and payload “TxData0A”.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x16
Frame type	3	0x10
Frame ID	4	0x01
64-bit destination address	MSB 5	0x00
	6	0x13
	7	0xA2
	8	0x00
	9	0x40
	10	0x0A
	11	0x01
	LSB 12	0x27
16-bit destination network address	MSB 13	0xFF
	LSB 14	0xFE
Broadcast radius	15	0x00

Frame data fields	Offset	Example
Options	16	0x40
RF data	17	0x54
	18	0x78
	19	0x44
	20	0x61
	21	0x74
	22	0x61
	23	0x30
	24	0x41
Checksum	25	0x13

If you enable escaping (**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 device calculates the checksum (on all non-escaped bytes) as [0xFF - (sum of all bytes from API frame type through data payload)].

Explicit Addressing Command frame - 0x11

Description

This frame is similar to Transmit Request (0x10), but it also requires you to specify the application-layer addressing fields: endpoints, cluster ID, and profile ID.

This frame causes the device to send payload data as an RF packet to a specific destination, using specific source and destination endpoints, cluster ID, and profile ID.These fields ignore the ones specified by **DE**,**SE** and **CI**.

- For broadcast transmissions, set the 64-bit destination address to **0x000000000000FFFF** .
- For unicast transmissions, set the 64 bit address field to the address of the desired destination node.
- Set the reserved field to **0xFFFE**.

Query the **NP** command to read the maximum number of payload bytes. For more information, see Firmware version/information commands.

You can read the maximum number of payload bytes with the **NP** command.

Format

The following table provides the contents of the frame. For details on the frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x11
Frame ID	4	Identifies the data frame for the host to correlate with a subsequent ACK. If set to **0**, the device does not send a response.
64-bit destination Address	5-12	MSB first, LSB last. Set to the 64-bit address of the destination device. Broadcast = 0x000000000000FFFF
Reserved	13-14	Set to **0xFFFE**.
Source Endpoint	15	Source Endpoint for the transmission.
Destination Endpoint	16	Destination Endpoint for the transmission.
Cluster ID	17-18	The Cluster ID that the host uses in the transmission.
Profile ID	19-20	The Profile ID that the host uses in the transmission.

Frame data fields	Offset	Description
Broadcast Radius	21	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius set to the network maximum hops value. If the broadcast radius exceeds the value of **NH** then the devices use the value of **NH** as the radius. Only broadcast transmissions use this parameter.
Transmission Options	22	See the Transmit Options table below. Set all other bits to 0.
Data Payload	23-n	Data that is sent to the destination device.

Transmit Options bit field

Bit field:

Bit	Meaning	Description
0	Disable ACK	Disable acknowledgments on all unicasts
1	Disable RD	Disable Route Discovery on all DigiMesh unicasts
2	NACK	Enable unicast NACK messages on all DigiMesh API packets
3	Trace Route	Enable a unicast Trace Route on all DigiMesh API packets
4	Reserved	<set this bit to 0>
5	Reserved	<set this bit to 0>
6,7	Delivery method	b’00 = <invalid option> b’01 - Point-multipoint (0x40) b’10 = Directed Broadcast (0x80) b’11 = DigiMesh (0xC0)

Set all other bits to 0.

Example

The following example sends a data transmission to a device with:

- 64-bit address: 0x0013A200 01238400
- Source endpoint: 0xE8
- Destination endpoint: 0xE8
- Cluster ID: 0x11
- Profile ID: 0xC105
- Payload: TxData

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x1A
Frame type	3	0x11
Frame ID	4	0x01
64-bit destination address	MSB 5	0x00
	6	0x13
	7	0xA2
	8	0x00
	9	0x01
	10	0x23
	11	0x84
	LSB12	0x00
Reserved	13	0xFF
	14	0xFE
Source endpoint	15	0xE8
Destination endpoint	16	0xE8
Cluster ID	17	0x00
	18	0x11
Profile ID	19	0xC1
	20	0x05
Broadcast radius	21	0x00
Transmit options	22	0x00
Data payload	23	0x54
	24	0x78
	25	0x44
	26	0x61
	27	0x74
	28	0x61
Checksum	29	0xA6

Remote AT Command Request frame - 0x17

Description

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

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x17
Frame ID	4	Identifies the data frame for the host to correlate with a subsequent ACK. If set to **0**, the device does not send a response.
64-bit destination address	5-12	MSB first, LSB last. Set to the 64-bit address of the destination device.
Reserved	13-14	Set to 0xFFFE.
Remote command options	15	0x02 = Apply changes on remote. If you do not set this, you must send the **AC** command for changes to take effect. Set all other bits to 0.
AT command	16-17	Command name: two ASCII characters that identify the command.
Command parameter	18-n	If present, indicates the parameter value you request for a given register. If no characters are present, it queries the register. Numeric parameter values are given in binary format.

Example

The following example sends a remote command:

- Change the broadcast hops register on a remote device to 1 (broadcasts go to 1-hop neighbors only).
- Apply changes so the new configuration value takes effect immediately.

In this example, the 64-bit address of the remote device is 0x0013A200 40401122.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x10
Frame type	3	0x17
Frame ID	4	0x01
64-bit destination address	MSB 5	0x00
	6	0x13
	7	0xA2
	8	0x00
	9	0x40
	10	0x40
	11	0x11
	LSB 12	0x22
Reserved	13	0xFF
	14	0xFE
Remote command options	15	0x02 (apply changes)
AT command	16	0x42 (B)
	17	0x48 (H)
Command parameter	18	0x01
Checksum	19	0xF5

AT Command Response frame - 0x88

Description

A device sends this frame in response to an AT Command (0x08 or 0x09) frame. Some commands send back multiple frames; for example, the **ND** command.

Format

Frame data fields	Offset	Description
Frame type	3	0x88
Frame ID	4	Identifies the data frame for the host to correlate with a subsequent ACK. If set to **0**, the device does not send a response.
AT command	5-6	Command name: two ASCII characters that identify the command.
Command status	7	0 = OK 1 = ERROR 2 = Invalid command 3 = Invalid parameter 4 = Tx failure
Command data	8-n	The register data in binary format. If the host sets the register, the device does not return this field.

Example

If you change the **BD** parameter on a local device with a frame ID of 0x01, and the parameter is valid, the user receives the following response.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x05
Frame type	3	0x88
Frame ID	4	0x01
AT command	5	0x42 (B)
	6	0x44 (D)

Frame data fields	Offset	Example
Command status	7	0x00
Command data		(No command data implies the parameter was set rather than queried)
Checksum	8	0xF0

Modem Status frame - 0x8A

Description

Devices send the status messages in this frame in response to specific conditions.

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x8A
Status	4	0x00 = Hardware reset 0x01 = Watchdog timer reset 0x0B = Network woke up 0x0C = Network went to sleep

Example

When a device powers up, it returns the following API frame.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
LSB 2	LSB 2	0x02
Frame type	3	0x8A
Status	4	0x00
Checksum	5	0x75

Transmit Status frame - 0x8B

Description

When a Transmit Request (0x10, 0x11) completes, the device sends a Transmit Status message out of the serial interface. This message indicates if the Transmit Request was successful or if it failed.

Note Broadcast transmissions are not acknowledged and always return a status of 0x00, even if the delivery failed.

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x8B
Frame ID	4	Identifies the serial interface data frame being reported. If Frame ID = 0 in the associated request frame, no response frame is delivered.
16-bit destination address	5	The 16-bit Network Address where the packet was delivered (if successful). If not successful, this address is 0xFFFD (destination address unknown).
	6	
Transmit retry count	7	The number of application transmission retries that occur.
Delivery status	8	0x00 = Success 0x01 = MAC ACK failure 0x02 = Collision avoidance failure 0x21 = Network ACK failure 0x25 = Route not found 0x31 = Internal resource error 0x32 = Internal error 0x74 = Payload too large 0x75 = Indirect message requested
Discovery status	9	0x00 = No discovery overhead 0x02 = Route discovery

Example

In the following example, the destination device reports a successful unicast data transmission successful and a route discovery occurred. The outgoing Transmit Request that this response frame uses Frame ID of 0x47.

Frame Fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x07
Frame type	3	0x8B
Frame ID	4	0x47
Reserved	5	0xFF
	6	0xFE
Transmit retry count	7	0x00
Delivery status	8	0x00
Discovery status	9	0x02
Checksum	10	0x2E

Route Information Packet frame - 0x8D

Description

If you enable NACK or the Trace Route option on a DigiMesh unicast transmission, a device can output this frame for the transmission.

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x8D
Source event	4	0x11 = NACK 0x12 = Trace route
Length	5	The number of bytes that follow, excluding the checksum. If the length increases, new items have been added to the end of the list for future revisions.
Timestamp	6-9	System timer value on the node generating the Route Information Packet. The timestamp is in microseconds. Only use this value for relative time measurements because the time stamp count restarts approximately every hour.
ACK timeout count	10	The number of MAC ACK timeouts that occur.
TX blocked count	11	The number of times the transmission was blocked due to reception in progress.
Reserved	12	Reserved, set to 0s.
Destination address	13-20	The address of the final destination node of this network-level transmission.
Source address	21-28	Address of the source node of this network-level transmission.
Responder address	29-36	Address of the node that generates this Route Information packet after it sends (or attempts to send) the packet to the next hop (the Receiver node).
Receiver address	37-44	Address of the node that the device sends (or attempts to send) the data packet.

Example

The following example represents a possible Route Information Packet. A device receives the packet when it performs a trace route on a transmission from one device (serial number 0x0013A200 4052AAAA) to another (serial number 0x0013A200 4052DDDD).

This particular frame indicates that the network successfully forwards the transmission from one device (serial number 0x0013A200 4052BBBB) to another device (serial number 0x0013A200 4052CCCC).

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x2A
Frame type	3	0x8D
Source event	4	0x12
Length	5	0x27
Timestamp	MSB 6	0x9C
	7	0x93
	8	0x81
	LSB 9	0x7F
ACK timeout count	10	0x00
TX blocked count	11	0x00
Reserved	12	0x00
Destination address	MSB 13	0x00
	14	0x13
	15	0xA2
	16	0x00
	17	0x40
	18	0x52
	19	0xAA
	LSB 20	0xAA

Frame data fields	Offset	Example
Source address	MSB 21	0x00
	22	0x13
	23	0xA2
	24	0x00
	25	0x40
	26	0x52
	27	0xDD
	LSB 28	0xDD
Responder address	MSB 29	0x00
	30	0x13
	31	0xA2
	32	0x00
	33	0x40
	34	0x52
	35	0xBB
	LSB 36	0xBB
Receiver address	MSB 37	0x00
	38	0x13
	39	0xA2
	40	0x00
	41	0x40
	42	0x52
	43	0xCC
	LSB 44	0xCC
Checksum	45	0xD2

Aggregate Addressing Update frame - 0x8E

Description

The device sends out an Aggregate Addressing Update frame on the serial interface of 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.

For more information, refer to Establish and maintain network links.

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x8E
Format ID	4	Byte reserved to indicate the format of additional packet information which may be added in future firmware revisions. In the current firmware revision, this field returns 0x00.
New address	5-12	Address to which **DH** and **DL** are being set.
Old address	13-20	Address to which **DH** and **DL** were previously set.

Example

In the following example, a device with destination address (**DH**/**DL**) of 0x0013A200 4052AAAA updates its destination address to 0x0013A200 4052BBBB.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x12
Frame type	3	0x8E
Format ID	4	0x00

Frame data fields	Offset	Example
New address	MSB 5	0x00
	6	0x13
	7	0xA2
	8	0x00
	9	0x40
	10	0x52
	11	0xBB
	LSB 12	0xBB
Old address	13	0x00
	14	0x13
	15	0xA2
	16	0x00
	17	0x40
	18	0x52
	19	0xAA
	20	0xAA
Checksum	21	0x19

Receive Packet frame - 0x90

Description

When a device configured with a standard API Rx Indicator (**AO** = **0**) receives an RF data packet, it sends it out the serial interface using this message type.

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x90
64-bit source address	4-11	The sender's 64-bit address. MSB first, LSB last.
Reserved	12-13	Reserved.
Receive options	14	Bit field: 0x00 = Packet acknowledged 0x01 = Packet was a broadcast packet 0x06, 0x07: b’01 = Point-Multipoint b’10 = Repeater mode (directed broadcast) b’11 = DigiMesh Ignore all other bits.
Received data	15-n	The RF data the device receives.

Example

In the following 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 sends the following frame out its serial interface.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x12
Frame type	3	0x90

Frame data fields	Offset	Example
64-bit source address	MSB 4	0x00
	5	0x13
	6	0xA2
	7	0x00
	8	0x40
	9	0x52
	10	0x2B
	LSB 11	0xAA
Reserved	12	0xFF
	13	0xFE
Receive options	14	0x01
Received data	15	0x52
	16	0x78
	17	0x44
	18	0x61
	19	0x74
	20	0x61
Checksum	21	0x11

Explicit Rx Indicator frame - 0x91

Description

When a device configured with explicit API Rx Indicator (**AO** = **1**) receives an RF packet, it sends it out the serial interface using this message type.

Note If a Transmit Request frame - 0x10 is sent to a device with **AO** = **1**, the receiving device receives a 0x91 frame with the Source endpoint (SE), Destination endpoint (DE), and Cluster ID (CI) that were set on the transmitting device in Transparent mode, and not the default values.

The Cluster ID and endpoints must be used to identify the type of transaction that occurred.

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x91
64-bit source address	4-11	MSB first, LSB last. The sender's 64-bit address.
Reserved	12-13	Reserved.
Source endpoint	14	Endpoint of the source that initiates transmission.
Destination endpoint	15	Endpoint of the destination where the message is addressed.
Cluster ID	16-17	The Cluster ID where the frame is addressed.
Profile ID	18-19	The Profile ID where the fame is addressed.
Receive options	20	Bit field: 0x00 = Packet acknowledged 0x01 = Packet was a broadcast packet 0x06, 0x07: b'01 = Point-Multipoint b'10 = Repeater mode (directed broadcast) b'11 = DigiMesh Ignore all other bits.
Received data	21-n	Received RF data.

Example

In the following example, a device with a 64-bit address of 0x0013A200 40522BAA sends a broadcast data transmission to a remote device with payload RxData.

If a device sends the transmission:

- With source and destination endpoints of 0xE0
- Cluster ID = 0x2211
- Profile ID = 0xC105

If **AO** = **1** on the receiving device, it sends the following frame out its serial interface.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x18
Frame type	3	0x91
64-bit source address	MSB 4	0x00
	5	0x13
	6	0xA2
	7	0x00
	8	0x40
	9	0x52
	10	0x2B
	LSB 11	0xAA
Reserved	12	0xFF
	13	0xFE
Source endpoint	14	0xE0
Destination endpoint	15	0xE0
Cluster ID	16	0x22
	17	0x11
Profile ID	18	0xC1
	19	0x05
Receive options	20	0x02

Frame data fields	Offset	Example
Received data	21	0x52
	22	0x78
	23	0x44
	24	0x61
	25	0x74
	26	0x61
Checksum	27	0x68

I/O Data Sample Rx Indicator frame - 0x92

Description

When you enable periodic I/O sampling or digital I/O change detection on a remote device, the UART of the device that receives the sample data sends this frame out.

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x92
64-bit source address	4-11	The sender's 64-bit address.
Reserved	12-13	Reserved.
Receive options	14	Bit field: 0x01 = Packet acknowledged 0x02 = Packet is a broadcast packet Ignore all other bits
Number of samples	15	The number of sample sets included in the payload. Always set to 1.
Digital channel mask	16-17	Bitmask field that indicates which digital I/O lines on the remote have sampling enabled, if any.
Analog channel mask	18	Bitmask field that indicates which analog I/O lines on the remote have sampling enabled, if any.
Digital samples (if included)	19-20	If the sample set includes any digital I/O lines (Digital channel mask > 0), these two bytes contain samples for all enabled digital I/O lines. DIO lines that do not have sampling enabled return 0. Bits in these two bytes map the same as they do in the Digital channel mask field.
Analog sample	21-n	If the sample set includes any analog I/O lines (Analog channel mask > 0), each enabled analog input returns a 2-byte value indicating the A/D measurement of that input. Analog samples are ordered sequentially from ADO/DIO0 to AD3/DIO3.

Example

In the following example, the device receives an I/O sample from a device with a 64-bit serial number of 0x0013A20040522BAA.

The configuration of the transmitting device takes a digital sample of a number of digital I/O lines and an analog sample of AD1. It reads the digital lines to be 0x0014 and the analog sample value is 0x0225.

The complete example frame is:

7E00 1492 0013 A200 4052 2BAA FFFE 0101 001C 0200 1402 25F9

Frame fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x14
64-bit source address	MSB 4	0x00
	5	0x13
	6	0xA2
	7	0x00
	8	0x40
	9	0x52
	10	0x2B
	LSB 11	0xAA
Reserved	MSB 12	0xFF
	LSB 13	0xFE
Receive options	14	0x01
Number of samples	15	0x01
Digital channel mask	16	0x00
	17	0x1C
Analog channel mask	18	0x02
Digital samples (if included)	19	0x00
	20	0x14
Analog sample	21	0x02
	22	0x25
Checksum	23	0xF5

Node Identification Indicator frame - 0x95

Description

A device receives this frame when:

- it transmits a node identification message to identify itself
- **AO** = **0**

The data portion of this frame is similar to a network discovery response. For more information, see ND (Network Discover).

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x95
64-bit source address	4-11	MSB first, LSB last. The sender's 64-bit address.
Reserved	12-13	Reserved.
Receive options	14	Bit field: 0x00 = Packet acknowledged 0x01 = Packet was a broadcast packet 0x06, 0x07: b’01 = Point-Multipoint b’10 = Repeater mode (directed broadcast) b’11 = DigiMesh Ignore all other bits
Reserved	15-16	Reserved.
64-bit remote address	17-24	Indicates the 64-bit address of the remote device that transmitted the Node Identification Indicator frame.
NI string	25-26	Node identifier string on the remote device. The NI string is terminated with a NULL byte (0x00).
Reserved	27-28	Reserved.

Frame data fields	Offset	Description
Device type	29	0 = Coordinator 1 = Normal Mode 2 = End Device For more options, see NO (Network Discovery Options).
Source event	30	1 = Frame sent by node identification pushbutton event—See D0 (DIO0/AD0).
Digi Profile ID	31-32	Set to the Digi application profile ID.
Digi Manufacturer ID	33-34	Set to the Digi Manufacturer ID.
Digi DD value (optional)	35-38	Reports the **DD** value of the responding device. Use the **NO** command to enable this field.
RSSI (optional)	39	Received signal strength indicator. Use the **NO** command to enable this field.

Example

If you press the commissioning pushbutton on a remote device with 64-bit address 0x0013A200407402AC and a default **NI** string sends a Node Identification, all devices on the network receive the following node identification indicator:

A remote device with 64-bit address 0x0013A200407402AC and a default **NI** string sends a Node Identification, all devices on the network receive the following node identification indicator:

```
0x7e 0025 9500 13a2 0040 7402 acff fec2 fffe 0013 a200 4074 02ac 2000 fffe 0101
c105 101e
```

If you press the commissioning button on a remote router device with 64-bit address 0x0013A200 40522BAA, 16-bit address 0x7D84, and default **NI** string, devices on the network receive the node identification indicator.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x25
Frame type	3	0x95

Frame data fields	Offset	Example
64-bit source address	MSB 4	0x00
	5	0x13
	6	0xA2
	7	0x00
	8	0x40
	9	0x74
	10	0x02
	LSB 11	0xAC
Reserved	12	0xFF
	13	0xFE
Receive options	14	0xC2
Reserved	15	0xFF
	16	0xFE
64-bit remote address	MSB 17	0x00
	18	0x13
	19	0xA2
	20	0x00
	21	0x40
	22	0x74
	23	0x02
	LSB 24	0xAC
NI string	25	0x20
	26	0x00
Reserved	27	0xFF
	28	0xFE
Device type	29	0x01
Source event	30	0x01
Digi Profile ID	31	0xC1
	32	0x05

Frame data fields	Offset	Example
Digi Manufacturer ID	33	0x10
	34	0x1E
Digi DD value (optional)	35	0x00
	36	0x0C
	37	0x00
	38	0x00
RSSI (optional)	39	0x2E
Checksum	40	0x33

Remote Command Response frame - 0x97

Description

If a device receives this frame in response to a Remote Command Request (0x17) frame, the device sends an AT Command Response (0x97) frame out the serial interface.

Some commands, such as the **ND** command, may send back multiple frames.

Format

The following table provides the contents of the frame. For details on frame structure, see API frame specifications.

Frame data fields	Offset	Description
Frame type	3	0x97
Frame ID	4	This is the same value that is passed in to the request.
64-bit source (remote) address	5-12	The address of the remote device returning this response.
Reserved	13-14	Reserved.
AT commands	15-16	The name of the command.
Command status	17	0 = OK 1 = ERROR 2 = Invalid Command 3 = Invalid Parameter 4 = No response
Command data	18-n	The value of the requested register.

Example

If a device sends a remote command to a remote device with 64-bit address 0x0013A200 40522BAA to query the **SL** command, and if the frame ID = 0x55, the response would look like the following example.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x13
Frame type	3	0x97
Frame ID	4	0x55

Frame data fields	Offset	Example
64-bit source (remote) address	MSB 5	0x00
	6	0x13
	7	0xA2
	8	0x00
	9	0x40
	10	0x52
	11	0x2B
	LSB 12	0xAA
Reserved	13	0xFF
	14	0xFE
16-bit source (remote) address	MSB 13	0x7D
	LSB 14	0x84
AT commands	15	0x53 (S)
	16	0x4C (L)
Command status	17	0x00
Command data	18	0x40
	19	0x52
	20	0x2B
	21	0xAA
Checksum	22	0xF4

Over-the-Air Firmware Update Status - 0xA0

Description

The Over-the-Air Firmware Update Status frame provides an indication of the status of a firmware update transmission attempt. A query command (0x01 0x51) sent to a target with a 64-bit address of 0x0013A200 40522BAA through an updater with 64-bit address 0x0013A200403E0750 and 16-bit address 0x0000, generates the following expected response.

Format

Frame data fields	Offset	Description
Frame type	3	0xA0
64-bit Source (remote) address	4-11	MSB first, LSB last. The address of the remote radio returning this response.
16-bit destination address	12-13	The 16-bit address of the updater device.
Receive options	14	0x01 - Packet Acknowledged. 0x02 - Packet was a broadcast.
Bootloader message type	15	0x06 - ACK 0x15 - NACK 0x40 - No Mac ACK 0x51 - Query (received if the bootloader is not active on the target) 0x52 - Query Response
Block number	16	Block number used in the update request. Set to 0 if not applicable.
64-bit target address	17-n	The 64-bit Address of remote device that is being updated (target)

Example

If a query request returns a 0x15 (NACK) status, the target is likely waiting for a firmware update image. If no messages are sent to it for about 75 seconds, the target will timeout and accept new query messages.

If a query returns a 0x51 (QUERY) status, then the target's bootloader is not active and will not respond to query messages.

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00

Frame data fields	Offset	Example
	LSB 2	0x16
Frame type	3	0xA0
64-bit source (remote) address	MSB 4	0x00
	5	0x13
	6	0xA2
	7	0x00
	8	0x40
	9	0x3E
	10	0x07
	11	0x50
16-bit destination address	12	0x00
	13	0x00
Receive options	14	0x01
Bootloader message type	15	0x52
Block number	16	0x00
64-bit target address	17	0x00
	18	0x13
	19	0xA2
	20	0x00
	21	0x40
	22	0x52
	23	0x2B
	24	0xAA
Checksum	25	0x66

Regulatory information

United States (FCC)

XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules comply with Part 15 of the FCC rules and regulations. Compliance with the labeling requirements, FCC notices and antenna usage guidelines is required.

To fulfill FCC Certification, the OEM must comply with the following regulations:

1. The system integrator must ensure that the text on the external label provided with this device is placed on the outside of the final product.
2. RF Modules may only be used with antennas that have been tested and approved for use with the modules.

OEM labeling requirements

WARNING! As an Original Equipment Manufacturer (OEM) you must ensure that FCC labeling requirements are met. You must include a clearly visible label on the outside of the final product enclosure that displays the following content:

Required FCC Label for OEM products containing the XBee-PRO S2C SMT RF Module

Contains FCC ID: MCQ-PS2CSM

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1.) this device may not cause harmful interference and (2.) this device must accept any interference received, including interference that may cause undesired operation.

Required FCC Label for OEM products containing the XBee S2C TH RF Module

Contains FCC ID: MCQ-S2CTH

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1.) this device may not cause harmful interference and (2.) this device must accept any interference received, including interference that may cause undesired operation.

Required FCC Label for OEM products containing the XBee-PRO S2C TH RF Module

Contains FCC ID: MCQ-PS2CTH

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1.) this device may not cause harmful interference and (2.) this device must accept any interference received, including interference that may cause undesired operation.

FCC notices

IMPORTANT: XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules have 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.

FCC-approved antennas (2.4 GHz)

The XBee and XBee-PRO RF Modules can be installed using antennas and cables constructed with non-standard connectors (RPSMA, RPTNC, etc.) An adapter cable may be necessary to attach the XBee connector to the antenna connector.

The modules are FCC approved for fixed base station and mobile applications for the channels indicated in the tables below. If the antenna is mounted at least 25 cm (10 in) from nearby persons, the application is considered a mobile application. Antennas not listed in the table must be tested to comply with FCC Section 15.203 (Unique Antenna Connectors) and Section 15.247 (Emissions).

The antennas in the tables below have been approved for use with this module. Cable loss is required when using gain antennas as shown in the tables. Digi does not carry all of these antenna variants. Contact Digi Sales for available antennas.

All antenna part numbers followed by an asterisk (*) are not available from Digi. Consult with an antenna manufacturer for an equivalent option.

XBee S2C SMT RF module

The following table shows the antennas approved for use with the XBee S2C SMT RF module.

Part number	Type (description)	Gain (dBi)	Application*	Min. separation	Required antenna cable loss (dB)		
					Channels 11-24	Channel 25	Channel 26
Integral antennas							
29000313	Integral PCB antenna	0.0	Fixed/Mobile	25 cm	N/A	N/A	N/A
A24-QI	Monopole (Integrated whip)	1.5	Fixed/Mobile	25 cm	N/A	N/A	N/A
Dipole antennas							
A24-HASM-450	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1	Fixed	25 cm	N/A	N/A	N/A
A24-HABSM*	Dipole (Articulated RPSMA)	2.1	Fixed	25 cm	N/A	N/A	N/A
29000095	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1	Fixed/Mobile	25 cm	N/A	N/A	N/A
A24-HABUF-P5I	Dipole (Half-wave articulated bulkhead mount U.FL. w/ 5” pigtail)	2.1	Fixed/Mobile	25 cm	N/A	N/A	N/A

Part number	Type (description)	Gain (dBi)	Application*	Min. separation	Required antenna cable loss (dB)		
					Channels 11-24	Channel 25	Channel 26
A24-HASM-525	Dipole (Half-wave articulated RPSMA - 5.25")	2.1	Fixed	25 cm	N/A	N/A	N/A
Omni-directional antennas							
A24-F2NF	Omni-directional (Fiberglass base station)	2.1	Fixed/Mobile	25 cm	N/A	N/A	N/A
A24-F3NF	Omni-directional (Fiberglass base station)	3.0	Fixed/Mobile	25 cm	N/A	N/A	N/A
A24-F5NF	Omni-directional (Fiberglass base station)	5.0	Fixed	25 cm	N/A	N/A	N/A
A24-F8NF	Omni-directional (Fiberglass base station)	8.0	Fixed	2 m	N/A	N/A	0.1
A24-F9NF	Omni-directional (Fiberglass base station)	9.5	Fixed	2 m	N/A	N/A	1.6
A24-F10NF	Omni-directional (Fiberglass base station)	10.0	Fixed	2 m	N/A	N/A	2.1
A24-F12NF	Omni-directional (Fiberglass base station)	12.0	Fixed	2 m	N/A	N/A	4.1
A24-W7NF	Omni-directional (Fiberglass base station)	7.2	Fixed	2 m	N/A	N/A	N/A
A24-M7NF	Omni-directional (Mag-mount base station)	7.2	Fixed	2 m	N/A	N/A	N/A
A24-F15NF	Omni-directional (Fiberglass base station)	15.0	Fixed	2 m	1.1	1.1	7.1
Panel antennas							
A24-P8SF	Flat Panel	8.5	Fixed	2 m	N/A	N/A	6.1
A24-P8NF	Flat Panel	8.5	Fixed	2 m	N/A	N/A	6.1
A24-P13NF	Flat Panel	13.0	Fixed	2 m	N/A	3.1	10.6
A24-P14NF	Flat Panel	14.0	Fixed	2 m	N/A	4.1	11.6
A24-P15NF	Flat Panel	15.0	Fixed	2 m	N/A	5.1	12.6
A24-P16NF	Flat Panel	16.0	Fixed	2 m	N/A	6.1	13.6

					Required antenna cable loss (dB)		
Part number	**Type (description)**	**Gain (dBi)**	**Application***	**Min. separation**	**Channels 11-24**	**Channel 25**	**Channel 26**
A24-P19NF	Flat Panel	19.0	Fixed	2 m	1.1	9.1	16.6
Yagi antennas							
A24-Y6NF	Yagi (6-element)	8.8	Fixed	2 m	N/A	N/A	3.9
A24-Y7NF	Yagi (7-element)	9.0	Fixed	2 m	N/A	N/A	4.1
A24-Y9NF	Yagi (9-element)	10.0	Fixed	2 m	N/A	N/A	5.1
A24-Y10NF	Yagi (10-element)	11.0	Fixed	2 m	N/A	0.6	6.1
A24-Y12NF	Yagi (12-element)	12.0	Fixed	2 m	N/A	1.6	7.1
A24-Y13NF	Yagi (13-element)	12.0	Fixed	2 m	N/A	1.6	7.1
A24-Y15NF	Yagi (15-element)	12.5	Fixed	2 m	N/A	2.1	7.6
A24-Y16NF	Yagi (16-element)	13.5	Fixed	2 m	N/A	3.1	8.6
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5 dBi	Fixed	2 m	N/A	3.1	8.6
A24-Y18NF	Yagi (18-element)	15.0	Fixed	2 m	1.1	4.6	10.1

XBee S2C TH RF Module

The following table shows the antennas approved for use with the XBee S2C TH RF Module.

Part number	Type (description)	Gain (dBi)	Application*	Min. separation	Required antenna cable loss (dB) Channels 11-24	Channel 25	Channel 26
Integral antennas							
29000294	Integral PCB antenna	-0.5	Fixed/Mobile	25 cm	N/A	N/A	N/A
A24-QI	Monopole (Integrated whip)	1.5	Fixed/Mobile	25 cm	N/A	N/A	N/A
Dipole antennas							
A24-HASM-450	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1	Fixed	25 cm	N/A	N/A	N/A
A24-HABSM	Dipole (Articulated RPSMA)	2.1	Fixed	25 cm	N/A	N/A	N/A
29000095	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1	Fixed/Mobile	25 cm	N/A	N/A	N/A
A24-HABUF-P5I	Dipole (Half-wave articulated bulkhead mount U.FL. w/ 5” pigtail)	2.1	Fixed/Mobile	25 cm	N/A	N/A	N/A
A24-HASM-525	Dipole (Half-wave articulated RPSMA - 5.25")	2.1	Fixed	25 cm	N/A	N/A	N/A
Omni-directional antennas							
A24-F2NF	Omni-directional (Fiberglass base station)	2.1	Fixed/Mobile	25 cm	N/A	N/A	N/A
A24-F3NF	Omni-directional (Fiberglass base station)	3.0	Fixed/Mobile	25 cm	N/A	N/A	N/A
A24-F5NF	Omni-directional (Fiberglass base station)	5.0	Fixed	25 cm	N/A	N/A	N/A
A24-F8NF	Omni-directional (Fiberglass base station)	8.0	Fixed	2 m	N/A	N/A	N/A

Part number	Type (description)	Gain (dBi)	Application*	Min. separation	Required antenna cable loss (dB)		
					Channels 11-24	Channel 25	Channel 26
A24-F9NF	Omni-directional (Fiberglass base station)	9.5	Fixed	2 m	N/A	N/A	0.9
A24-F10NF	Omni-directional (Fiberglass base station)	10.0	Fixed	2 m	N/A	N/A	1.4
A24-F12NF	Omni-directional (Fiberglass base station)	12.0	Fixed	2 m	N/A	N/A	3.4
A24-W7NF	Omni-directional (Fiberglass base station)	7.2	Fixed	2 m	N/A	N/A	N/A
A24-M7NF	Omni-directional (Mag-mount base station)	7.2	Fixed	2 m	N/A	N/A	N/A
A24-F15NF	Omni-directional (Fiberglass base station)	15.0	Fixed	2 m	0.4	0.4	6.4
Panel antennas							
A24-P8SF	Flat Panel	8.5	Fixed	2 m	N/A	N/A	4.9
A24-P8NF	Flat Panel	8.5	Fixed	2 m	N/A	N/A	4.9
A24-P13NF	Flat Panel	13.0	Fixed	2 m	N/A	3.4	9.4
A24-P14NF	Flat Panel	14.0	Fixed	2 m	N/A	4.4	10.4
A24-P15NF	Flat Panel	15.0	Fixed	2 m	N/A	5.4	11.4
A24-P16NF	Flat Panel	16.0	Fixed	2 m	N/A	6.4	12.4
A24-P19NF	Flat Panel	19.0	Fixed	2 m	0.4	9.4	15.4
Yagi antennas							
A24-Y6NF	Yagi (6-element)	8.8	Fixed	2 m	N/A	N/A	4.7
A24-Y7NF	Yagi (7-element)	9.0	Fixed	2 m	N/A	N/A	4.9
A24-Y9NF	Yagi (9-element)	10.0	Fixed	2 m	N/A	0.4	5.9
A24-Y10NF	Yagi (10-element)	11.0	Fixed	2 m	N/A	1.4	6.9

Part number	Type (description)	Gain (dBi)	Application*	Min. separation	Required antenna cable loss (dB)		
					Channels 11-24	Channel 25	Channel 26
A24-Y12NF	Yagi (12-element)	12.0	Fixed	2 m	N/A	2.4	7.9
A24-Y13NF	Yagi (13-element)	12.0	Fixed	2 m	N/A	2.4	7.9
A24-Y15NF	Yagi (15-element)	12.5	Fixed	2 m	N/A	2.9	8.4
A24-Y16NF	Yagi (16-element)	13.5	Fixed	2 m	N/A	3.9	9.4
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5	Fixed	2 m	N/A	3.9	9.4
A24-Y18NF	Yagi (18-element)	15.0	Fixed	2 m	0.4	5.4	10.9

XBee-PRO S2C SMT RF Module

The following table shows the antennas approved for use with the XBee-PRO S2C SMT RF Module.

Part Number	Type (Description)	Gain (dBi)	Application*	Min Separation	Required antenna cable loss (dB)	
					Channels 11-23†	Channel 24†
Internal antennas						
29000313	Integral PCB antenna	0.0	Fixed/Mobile	25 cm	N/A	N/A
A24-QI	Monopole (Integrated whip)	1.5	Fixed/Mobile	25 cm	N/A	N/A
Dipole antennas						
A24-HASM-450	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1	Fixed	25 cm	N/A	N/A
A24-HABSM	Dipole (Articulated RPSMA)	2.1	Fixed	25 cm	N/A	N/A
29000095	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1	Fixed/Mobile	25 cm	N/A	N/A
A24-HABUF-P5I	Dipole (Half-wave articulated bulkhead mount U.FL. w/ 5” pigtail)	2.1	Fixed/Mobile	25 cm	N/A	N/A
A24-HASM-525	Dipole (Half-wave articulated RPSMA - 5.25")	2.1	Fixed	25 cm	N/A	N/A
Omni-directional antennas						
A24-F2NF	Omni-directional (Fiberglass base station)	2.1	Fixed/Mobile	25 cm	N/A	N/A
A24-F3NF	Omni-directional (Fiberglass base station)	3.0	Fixed/Mobile	25 cm	N/A	N/A
A24-F5NF	Omni-directional (Fiberglass base station)	5.0	Fixed	25 cm	N/A	N/A
A24-F8NF	Omni-directional (Fiberglass base station)	8.0	Fixed	2 m	N/A	N/A

Part Number	Type (Description)	Gain (dBi)	Application*	Min Separation	Required antenna cable loss (dB)	
					Channels 11-23†	Channel 24†
A24-F9NF	Omni-directional (Fiberglass base station)	9.5	Fixed	2 m	N/A	N/A
A24-F10NF	Omni-directional (Fiberglass base station)	10	Fixed	2 m	N/A	N/A
A24-F12NF	Omni-directional (Fiberglass base station)	12	Fixed	2 m	N/A	1.6
A24-W7NF	Omni-directional (Fiberglass base station)	7.2	Fixed	2 m	N/A	N/A
A24-M7NF	Omni-directional (Mag-mount base station)	7.2	Fixed	2 m	N/A	N/A
A24-F15NF	Omni-directional (Fiberglass base station)	15.0	Fixed	2 m	1.1	4.6
Panel antennas						
A24-P8SF	Flat Panel	8.5	Fixed	2 m	N/A	2.1
A24-P8NF	Flat Panel	8.5	Fixed	2 m	N/A	2.1
A24-P13NF	Flat Panel	13.0	Fixed	2 m	2.7	6.6
A24-P14NF	Flat Panel	14.0	Fixed	2 m	3.7	7.6
A24-P15NF	Flat Panel	15.0	Fixed	2 m	4.7	8.6
A24-P16NF	Flat Panel	16.0	Fixed	2 m	5.7	9.6
A24-P19NF	Flat Panel	19.0	Fixed	2 m	8.7	12.6
Yagi antennas						
A24-Y6NF	Yagi (6-element)	8.8	Fixed	2 m	N/A	1.9
A24-Y7NF	Yagi (7-element)	9.0	Fixed	2 m	N/A	2.1
A24-Y9NF	Yagi (9-element)	10.0	Fixed	2 m	N/A	3.1
A24-Y10NF	Yagi (10-element)	11.0	Fixed	2 m	0.6	4.1

Part Number	Type (Description)	Gain (dBi)	Application*	Min Separation	Required antenna cable loss (dB)	
					Channels 11-23†	Channel 24†
A24-Y12NF	Yagi (12-element)	12.0	Fixed	2 m	1.6	5.1
A24-Y13NF	Yagi (13-element)	12.0	Fixed	2 m	1.6	5.1
A24-Y15NF	Yagi (15-element)	12.5	Fixed	2 m	2.1	5.6
A24-Y16NF	Yagi (16-element)	13.5	Fixed	2 m	3.1	6.6
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5	Fixed	2 m	3.1	6.6
A24-Y18NF	Yagi (18-element)	15.0	Fixed	2 m	4.6	8.1

XBee-PRO S2C TH RF Module

The following table shows the antennas approved for use with the XBee-PRO S2C TH RF Module.

Part number	Type (description)	Gain (dBi)	Application*	Min. separation	Required antenna cable loss (dB)	
					Channels 11-23†	Channel 24†
Integral antennas						
29000294	Integral PCB antenna	-0.5	Fixed/Mobile	25 cm	N/A	N/A
A24-QI	Monopole (Integrated whip)	1.5	Fixed/Mobile	25 cm	N/A	N/A
Dipole antennas						
A24-HASM-450	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1	Fixed/Mobile	25 cm	N/A	N/A
A24-HABSM	Dipole (Articulated RPSMA)	2.1	Fixed	25 cm	N/A	N/A
29000095	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1	Fixed/Mobile	25 cm	N/A	N/A
A24-HABUF-P5I	Dipole (Half-wave articulated bulkhead mount U.FL. w/ 5” pigtail)	2.1	Fixed	25 cm	N/A	N/A
A24-HASM-525	Dipole (Half-wave articulated RPSMA - 5.25")	2.1	Fixed/ Mobile	25 cm	N/A	N/A
Omni-directional antennas						
A24-F2NF	Omni-directional (Fiberglass base station)	2.1	Fixed/Mobile	25 cm	N/A	N/A
A24-F3NF	Omni-directional (Fiberglass base station)	3.0	Fixed/Mobile	25 cm	N/A	N/A
A24-F5NF	Omni-directional (Fiberglass base station)	5.0	Fixed	25 cm	N/A	N/A
A24-F8NF	Omni-directional (Fiberglass base station)	8.0	Fixed	2 m	N/A	N/A

Part number	Type (description)	Gain (dBi)	Application*	Min. separation	Required antenna cable loss (dB)	
					Channels 11-23†	Channel 24†
A24-F9NF	Omni-directional (Fiberglass base station)	9.5	Fixed	2 m	N/A	N/A
A24-F10NF	Omni-directional (Fiberglass base station)	10.0	Fixed	2 m	N/A	N/A
A24-F12NF	Omni-directional (Fiberglass base station)	12.0	Fixed	2 m	N/A	1.4
A24-W7NF	Omni-directional (base station)	7.2	Fixed	2 m	N/A	N/A
A24-M7NF	Omni-directional (Mag-mount base station)	7.2	Fixed	2 m	N/A	N/A
A24-F15NF	Omni-directional (Fiberglass base station)	15.0	Fixed	2 m	0.4	4.4
Panel antennas						
A24-P8SF	Flat Panel	8.5	Fixed	2 m	N/A	0.4
A24-P8NF	Flat Panel	8.5	Fixed	2 m	N/A	0.4
A24-P13NF	Flat Panel	13	Fixed	2 m	2.4	4.9
A24-P14NF	Flat Panel	14	Fixed	2 m	3.4	5.9
A24-P15NF	Flat Panel	15.0	Fixed	2 m	4.4	6.9
A24-P16NF	Flat Panel	16.0	Fixed	2 m	5.4	7.9
A24-19NF	Flat Panel	19.0	Fixed	2 m	8.4	10.9
Yagi antennas						
A24-Y6NF	Yagi (6-element)	8.8	Fixed	2 m	N/A	1.2
A24-Y7NF	Yagi (7-element)	9.0	Fixed	2 m	N/A	1.4
A24-Y9NF	Yagi (9-element)	10.0	Fixed	2 m	N/A	2.4
A24-Y10NF	Yagi (10-element)	11.0 dBi	Fixed	2 m	0.4	3.4

Part number	Type (description)	Gain (dBi)	Application*	Min. separation	Required antenna cable loss (dB)	
					Channels 11-23†	Channel 24†
A24-Y12NF	Yagi (12-element)	12.0	Fixed	2 m	1.4	4.4
A24-Y13NF	Yagi (13-element)	12.0	Fixed	2 m	1.4	4.4
A24-Y15NF	Yagi (15-element)	12.5	Fixed	2 m	1.9	4.9
A24-Y16NF	Yagi (16-element)	13.5	Fixed	2 m	2.9	5.9
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5	Fixed	2 m	2.9	5.9
A24-Y18NF	Yagi (18-element)	15.0	Fixed	2 m	4.4	7.4

* If using the RF module in a portable application (for example - if the module is used in a handheld device and the antenna is less than 25 cm from the human body when the device is in operation): The integrator is responsible for passing additional SAR (Specific Absorption Rate) testing based on FCC rules 2.1091 and FCC Guidelines for Human Exposure to Radio Frequency Electromagnetic Fields, OET Bulletin and Supplement C. The testing results will be submitted to the FCC for approval prior to selling the integrated unit. The required SAR testing measures emissions from the module and how they affect the person.

† Although certified to operate on channels 11-24, currently this product only supports channels 12-23.

RF exposure

If you are an integrating the XBee into another product, you must include the following Caution statement in OEM product manuals to alert users of FCC RF exposure compliance:

CAUTION! To satisfy FCC RF exposure requirements for mobile transmitting devices, a separation distance of 20 cm or more should be maintained between the antenna of this device and persons during device operation. To ensure compliance, operations at closer than this distance are not recommended. The antenna used for this transmitter must not be co-located in conjunction with any other antenna or transmitter.

FCC publication 996369 related information

In publication 996369 section D03, the FCC requires information concerning a module to be presented by OEM manufacturers. This section assists in answering or fulfilling these requirements.

2.1 General

No requirements are associated with this section.

2.2 List of applicable FCC rules

This module conforms to FCC Part 15.247.

2.3 Summarize the specific operational use conditions

Certain approved antennas require attenuation for operation. For the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module, see FCC-approved antennas (2.4 GHz).

Host product user guides should include the antenna table if end customers are permitted to select antennas.

2.4 Limited module procedures

Not applicable.

2.5 Trace antenna designs

While it is possible to build a trace antenna into the host PCB, this requires at least a Class II permissive change to the FCC grant which includes significant extra testing and cost. If an embedded trace antenna is desired, simply select the XBee module variant with the preferred antenna.

2.6 RF exposure considerations

For RF exposure considerations see RF exposure and FCC-approved antennas (2.4 GHz).

Host product manufacturers need to provide end-users a copy of the “RF Exposure” section of the manual: RF exposure.

2.7 Antennas

A list of approved antennas is provided for the XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules. See FCC-approved antennas (2.4 GHz).

2.8 Label and compliance information

Host product manufacturers need to follow the sticker guidelines outlined in OEM labeling requirements.

2.9 Information on test modes and additional testing requirements

Contact a Digi sales representative for information on how to configure test modes for the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module.

2.10 Additional testing, Part 15 Subpart B disclaimer

All final host products must be tested to be compliant to FCC Part 15 Subpart B standards. While the XBee/XBee-PRO S2C DigiMesh 2.4 module was tested to be complaint to FCC unintentional radiator standards, FCC Part 15 Subpart B compliance testing is still required for the final host product. This testing is required for all end products. XBee/XBee-PRO S2C DigiMesh 2.4 RF Module Part 15 Subpart B compliance does not affirm the end product's compliance.

See FCC notices for more details.

Europe (CE)

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules (non-PRO variants) have been tested for use in several European countries. For a complete list, refer to www.digi.com/resources/certifications.

If XBee/XBee-PRO S2C DigiMesh 2.4 RF Modules are incorporated into a product, the manufacturer must ensure compliance of the final product with articles 3.1a and 3.1b of the Radio Equipment Directive. A Declaration of Conformity must be issued for each of these standards and kept on file as described in the Radio Equipment Directive.

Furthermore, the manufacturer must maintain a copy of the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module user guide documentation and ensure the final product does not exceed the specified power ratings, antenna specifications, and/or installation requirements as specified in the user guide.

Maximum power and frequency specifications

For the through-hole device:

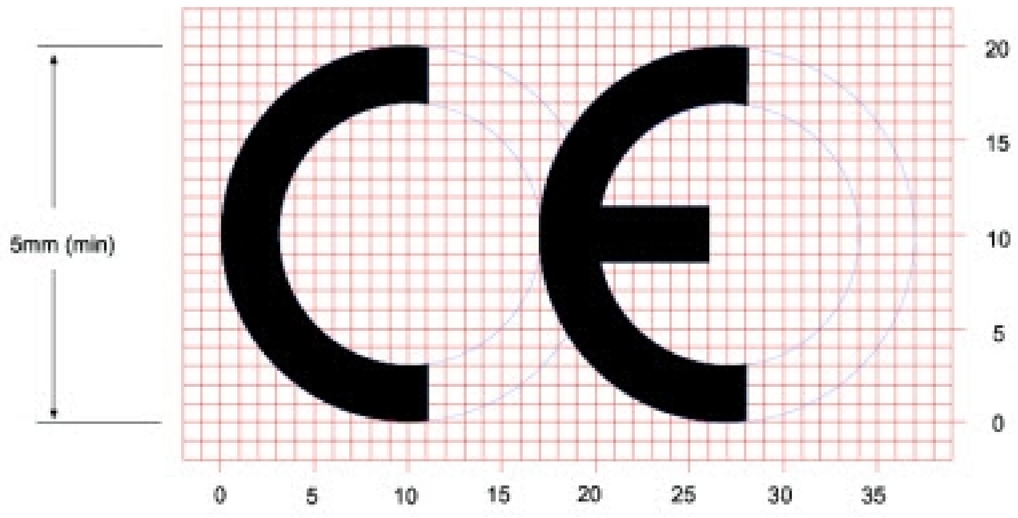
- Maximum power: 9.82 mW (9.92 dBm) Equivalent Isotropically Radiated Power (EIRP) at normal condition.
- Frequencies: 5 MHz channel spacing, beginning at 2405 MHz and ending at 2480 MHz.

For the surface-mount device:

- Maximum power: 12.65 mW (11.02 dBm) EIRP.
- Frequencies: 5 MHz channel spacing, beginning at 2405 MHz and ending at 2480 MHz.

OEM labeling requirements

The "CE" marking must be affixed to a visible location on the OEM product. The following figure shows CE labeling requirements.



The CE mark shall consist of the initials “CE” taking the following form:

- If the CE marking is reduced or enlarged, the proportions given in the above graduated drawing must be respected.
- The CE marking must have a height of at least 5 mm except where this is not possible on account of the nature of the apparatus.
- The CE marking must be affixed visibly, legibly, and indelibly.

Important note

Digi customers assume full responsibility for learning and meeting the required guidelines for each country in their distribution market. Refer to the radio regulatory agency in the desired countries of operation for more information.

Listen Before Talk requirement

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module must be configured to comply with the Listen Before Talk (LBT) requirements in the EN 300 328 standard. This can be accomplished by one of the following options:

1. Set the **PL** command to 3 (6 dBm) or lower, which ensures that the maximum transmitter power is under the limit at which LBT is required.
   or
2. Set the **CA** command as described in CA (CCA Threshold) to enable LBT at the required noise threshold level.

Declarations of conformity

Digi has issued Declarations of Conformity for the XBee RF Modules concerning emissions, EMC, and safety. For more information, see www.digi.com/resources/certifications.

Antennas

The following antennas have been tested and approved for use with the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module:

All antenna part numbers followed by an asterisk (*) are not available from Digi. Consult with an antenna manufacturer for an equivalent option.

- Dipole (2.1 dBi, Omni-directional, Articulated RPSMA, Digi part number A24-HABSM)
- PCB Antenna (0.0 dBi)
- Monopole Whip (1.5 dBi)

ISED (Innovation, Science and Economic Development Canada)

This device complies with Industry Canada license-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 the following text:

For XBee S2C surface-mount

Contains Model XBee S2C Radio, IC: 1846A-XBS2C

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.

For XBee-PRO S2C surface-mount

Contains Model PS2CSM Radio, IC: 1846A-PS2CSM

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.

For XBee S2C through-hole

Contains Model S2CTH Radio, IC: 1846A-S2CTH

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.

For XBee-PRO S2C through-hole

Contains Model PS2CTH Radio, IC: 1846A-PS2CTH

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.

Transmitters for detachable antennas

This radio transmitter has been approved by Industry Canada to operate with the antenna types listed in the tables in 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. The required antenna impedance is 50 ohms.

Le présent émetteur radio a été approuvé par Industrie Canada pour fonctionner avec les types d'antenne énumérés 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 (EIRP) 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 àl'établissement d'une communication satisfaisante.

Australia (RCM)

XBee DigiMesh 2.4 and XBee-PRO DigiMesh 2.4 modules comply with requirements to be used in end products in Australia and New Zealand. All products with EMC and radio communications must have registered RCM and R-NZ marks. Registration to use the compliance mark will only be accepted from Australia or New Zealand manufacturers or importers, or their agents.

In order to have a RCM or R-NZ mark on an end product, a company must comply with **a** or **b** below.

a. Have a company presence in Australia or New Zealand.
b. Have a company/distributor/agent in Australia or New Zealand that will sponsor the importing of the end product.

Contact Digi for questions related to locating a contact in Australia and New Zealand.

South Korea

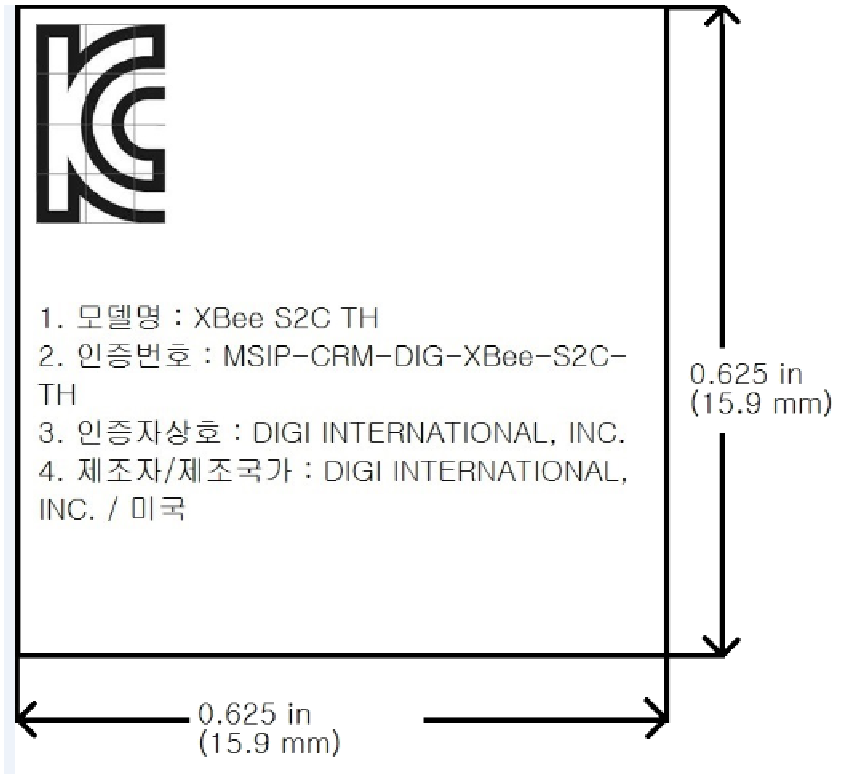
The low-power XBee S2C TH and XBee S2C devices have received South Korean approvals. To show conformity to the certificate, you must add a label with the South Korean product information to the XBee S2C DigiMesh RF Module.

For the through-hole device, you can place the label on the reverse side.

Recommended label material: Abraham Technical (700342) MFG P/N TAAE-014250.

The label size is: 15.9 mm x 15.9 mm (0.625 in x 0.625 in)

The complete label information is as follows:

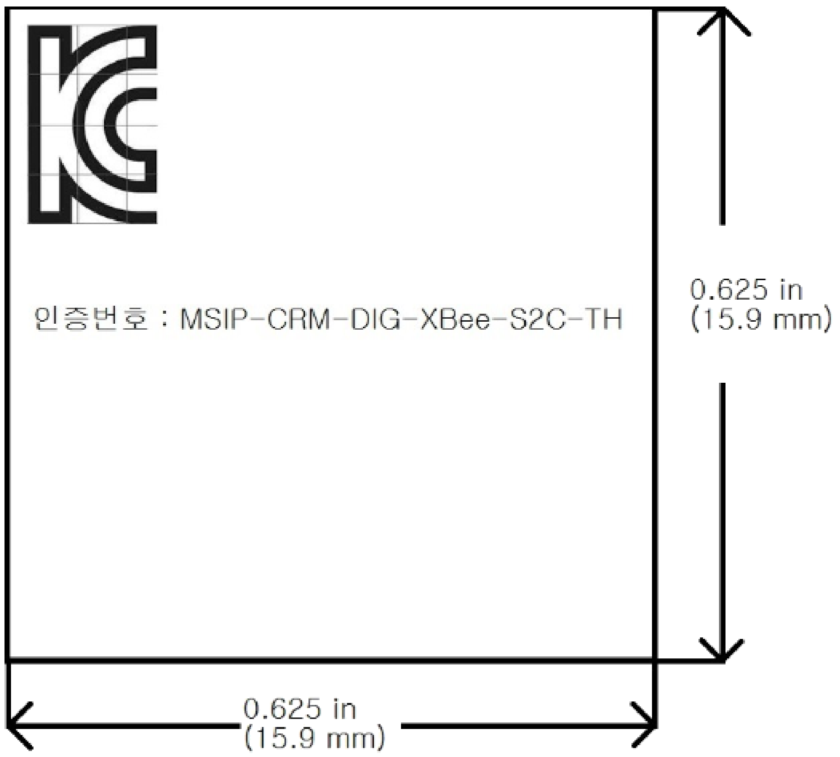

1. 모델명 : XBee S2C TH
2. 인증번호 : MSIP-CRM-DIG-XBee-S2C-TH
3. 인증자상호 : DIGI INTERNATIONAL, INC.
4. 제조자/제조국가 : DIGI INTERNATIONAL, INC. / 미국
0.625 in (15.9 mm)
0.625 in (15.9 mm)

The KCC logo must be at least 5 mm tall.

The text shown in the label is:

1. 모델명 : XBee S2C TH
2. 인증번호 : MSIP-CRM-DIG-XBee-S2C-TH
3. 인증자상호 : DIGI INTERNATIONAL, INC.
4. 제조자/제조국가 : DIGI INTERNATIONAL, INC. / 미국

If the label size does not accommodate the required content, you can use abbreviated information, as follows:



The KCC logo must be at least 5 mm tall.

The text shown on the label is:

인증번호 : MSIP-CRM-DIG-XBee-S2C-TH

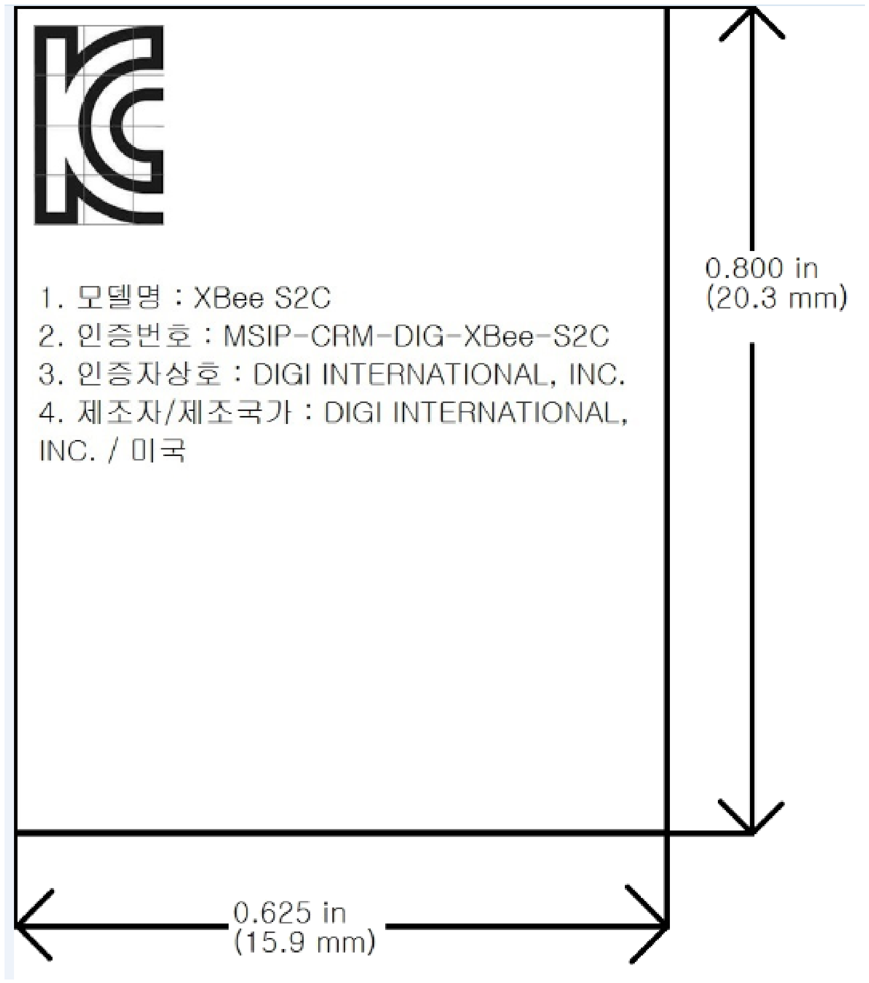
For the surface-mount version, the label will overlay the existing product label.

CAUTION! By placing a label over the existing label, the certifications for Europe (CE), Australia, New Zealand (RCM), and Japan will no longer apply.

Recommended label material: Abraham Technical TELT-000465.

The label size is: 15.9 mm x 20.3 mm (0.625 in x 0.8 in)

The complete label information is as follows:

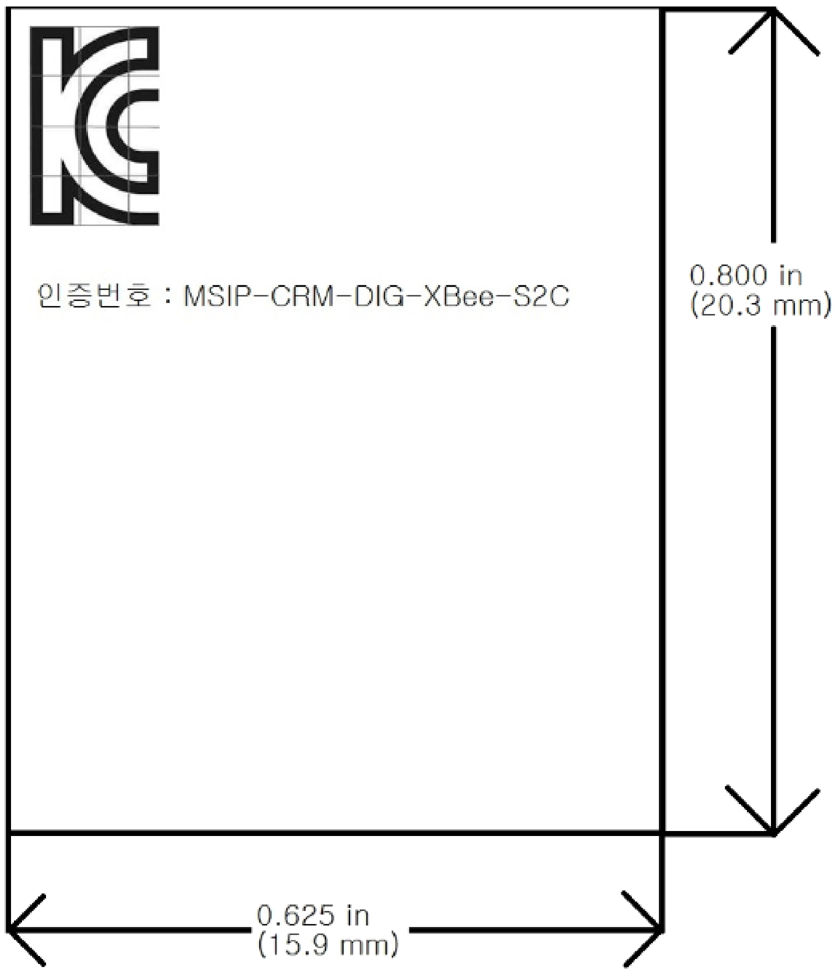

1. 모델명 : XBee S2C
2. 인증번호 : MSIP-CRM-DIG-XBee-S2C
3. 인증자상호 : DIGI INTERNATIONAL, INC.
4. 제조자/제조국가 : DIGI INTERNATIONAL, INC. / 미국
0.800 in (20.3 mm)
0.625 in (15.9 mm)

The KCC logo must be at least 5 mm tall.

The text shown in the label is:

1. 모델명 : XBee S2C
2. 인증번호 : MSIP-CRM-DIG-XBee-S2C
3. 인증자상호 : DIGI INTERNATIONAL, INC.
4. 제조자/제조국가 : DIGI INTERNATIONAL, INC. / 미국

If the label size does not accommodate the required content, you can use the abbreviated information, as follows:



The KCC logo must be at least 5 mm tall.

The text shown in the label is:

인증번호 : MSIP-CRM-DIG-XBee-S2C

Load DigiMesh 2.4 firmware on ZB devices

Background

Our XBee/XBee-PRO ZB RF modules are built on the same hardware as the XBee/XBee-PRO S2C DigiMesh 2.4 RF Module. It is possible to load DigiMesh 2.4 firmware on existing ZB modules. The table below shows which part numbers are compatible with DigiMesh 2.4 firmware.

Note Currently the DigiMesh 2.4 firmware is approved for use only in the United States, Canada, Europe, Australia and Japan. You can find region-specific regulatory information for the firmware in Regulatory information.

CAUTION! The antenna cable loss requirements for the DigiMesh 2.4 firmware are different than the ZB firmware for gain antennas exceeding 2.1 dBi. If you migrate a ZB device to DigiMesh 2.4 firmware, and are using gain antennas, you must adhere to the cable loss requirements found in Regulatory information.

XBee/XBee-PRO DigiMesh 2.4 S2C part numbers	Revision	Form factor	Hardware version (HV)
XB24CZ7PIS-004 XB24CZ7RIS-004 XB24CZ7UIS-004	All	XBee SMT	0x22
XB24CZ7PIT-004 XB24CZ7SIT-004 XB24CZ7UIT-004 XB24CZ7WIT-004	All	XBee TH	0x2E
XBP24CZ7PIS-004 XBP24CZ7RIS-004 XBP24CZ7UIS-004	Rev L (and later)	XBee SMT	0x30
XBP24CZ7PIT-004 XBP24CZ7SIT-004 XBP24CZ7UIT-004 XBP24CZ7WIT-004	All	XBee TH	0x2D

In addition to the differences between the DigiMesh 2.4 and Zigbee protocols, some of the operational features are different between the two firmware versions. For example, the XBee-PRO DigiMesh 2.4 supports fewer channels than the Zigbee firmware. It is important that you read and understand this user guide before developing with the DigiMesh 2.4 firmware.

Load firmware

To load DigiMesh 2.4 firmware on an existing Zigbee or 802.154 device, use the following instructions.

1. Verify that your device's part number (listed on the label) is included in the list shown in Background.
2. Install the device in a Digi development board and connect it to your PC.
3. The next steps involve loading firmware using XCTU. To download XCTU and read detailed instructions about it, go to:
   https://www.digi.com/products/xbee-rf-solutions/xctu-software/xctu

4. When you get to the **Update firmware** dialog box, in the **Function set** area, click the **DigiMesh 2.4** option, and the newest firmware version.
5. Click **Update** and follow the instructions.
6. When the updating process successfully completes, your device runs DigiMesh 2.4 firmware. You can change back to Zigbee or 802.15.4 firmware at any time by following the same process and selecting the function set, which specifies whether you want to use the Zigbee, 802.15.4, or DigiMesh 2.4 protocol.

Migrate from XBee through-hole to surface-mount devices

We design the XBee surface-mount and through-hole devices to be compatible with each other and offer the same basic feature set. The surface-mount form factor has more I/O pins. Because the XBee device was originally offered in only the through-hole form factor, we offer this section to help you migrate from the through-hole to the surface-mount form factor.

Pin mapping

The following table shows the pin mapping for the surface-mount (SMT) pins to the through-hole (TH) pins. The pin names are from the XBee S2C SMT device.

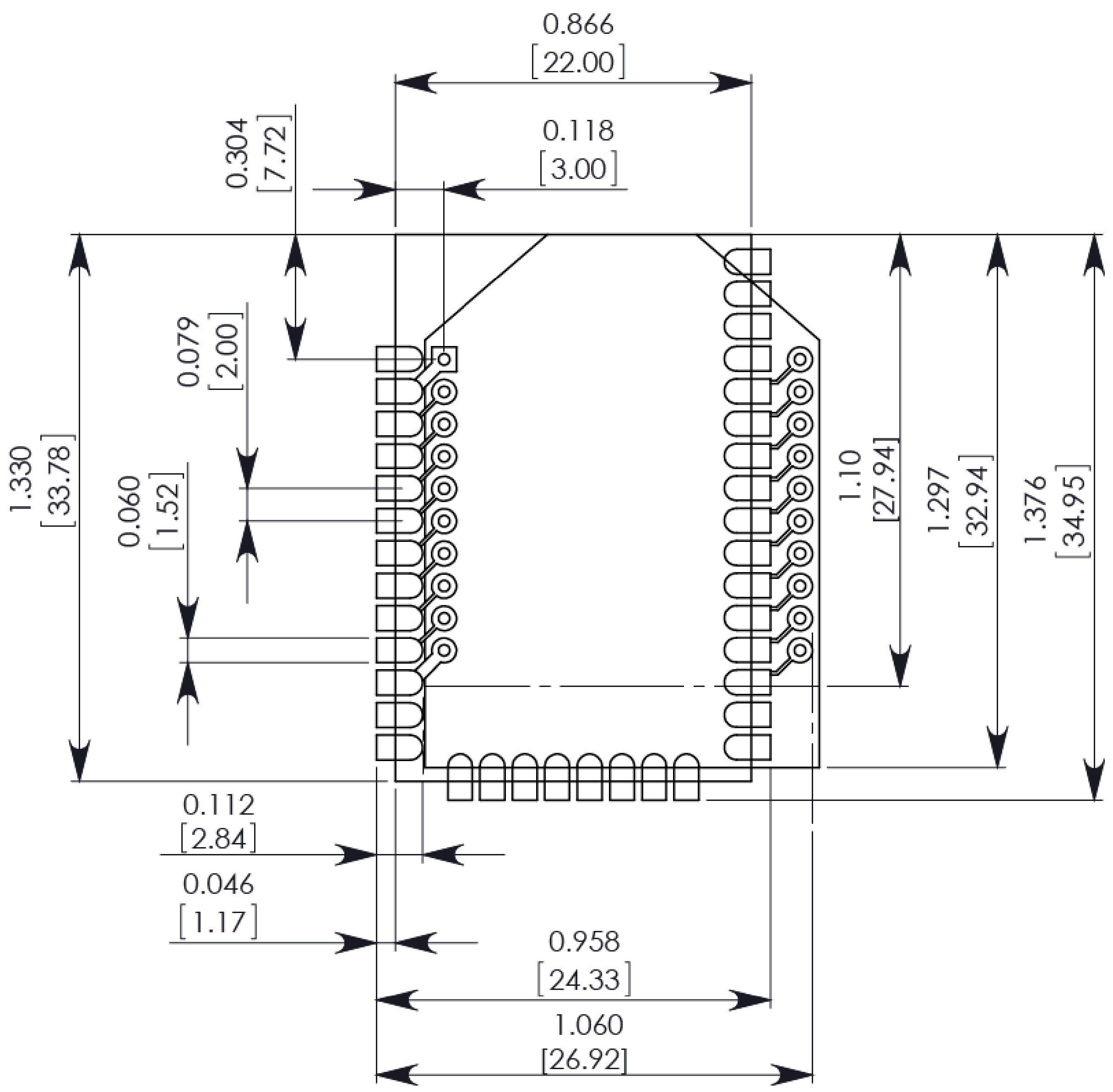
SMT Pin #	Name	TH Pin #
1	GND	
2	VCC	1
3	DOUT	2
4	DIN/$\overline{\text{CONFIG}}$	3
5	[Reserved]	4
6	$\overline{\text{RESET}}$	5
7	PWM0/RSSI PWM	6
8	PWM1	7
9	[Reserved]	8
10	DI8/SLEEP_RQ/$\overline{\text{DTR}}$	9
11	GND	10
12	SPI_$\overline{\text{ATTN}}$ /$\overline{\text{BOOTMODE}}$	
13	GND	
14	SPI_CLK	
15	SPI_$\overline{\text{SSEL}}$	
16	SPI_MOSI	
17	SPI_MISO	
18	[Reserved]	
19	[Reserved]	
20	[Reserved]	
21	[Reserved]	
22	GND	
23	[Reserved]	
24	DIO4	11
25	DIO7/$\overline{\text{CTS}}$	12
26	On/$\overline{\text{SLEEP}}$	13
27	V_{REF}	14

SMT Pin #	Name	TH Pin #
28	DIO5/ASSOC	15
29	DIO6/$\overline{\text{RTS}}$	16
30	DIO3/AD3	17
31	DIO2/AD2	18
32	DIO1/AD1	19
33	DIO0/AD0	20
34	[Reserved]	
35	GND	
36	RF	
37	[Reserved]	

Mount the devices

One important difference between the SMT and TH devices is the way they mount to a printed circuit board (PCB). Each footprint requires different mounting techniques.

We designed a footprint that allows you to attach either device to a PCB. The following drawing shows the layout.



The round holes in the diagram are for the TH design, and the semi-oval pads are for the SMT design. Pin 1 of the TH design is lined up with pad 1 of the SMT design, but the pins are actually offset by one pad; see Pin mapping. By using diagonal traces to connect the appropriate pins, the layout will work for both devices.

PCB design and manufacturing contains information on attaching the SMT device.

PCB design and manufacturing

The XBee/XBee-PRO S2C DigiMesh 2.4 RF Module is designed for surface-mount on the OEM PCB. It has castellated pads to allow for easy solder attach inspection. The pads are all located on the edge of the module, so there are no hidden solder joints on these modules.

Recommended solder reflow cycle

The following table provides the recommended solder reflow cycle. The table shows the temperature setting and the time to reach the temperature; it does not show the cooling cycle.

Time (seconds)	Temperature (degrees C)
30	65
60	100
90	135
120	160
150	195
180	240
210	260

The maximum temperature should not exceed 260 °C.

The device will reflow during this cycle, and therefore must not be reflowed upside down. Take care not to jar the device while the solder is molten, as this can remove components under the shield from their required locations.

Hand soldering is possible and should be performed in accordance with approved standards.

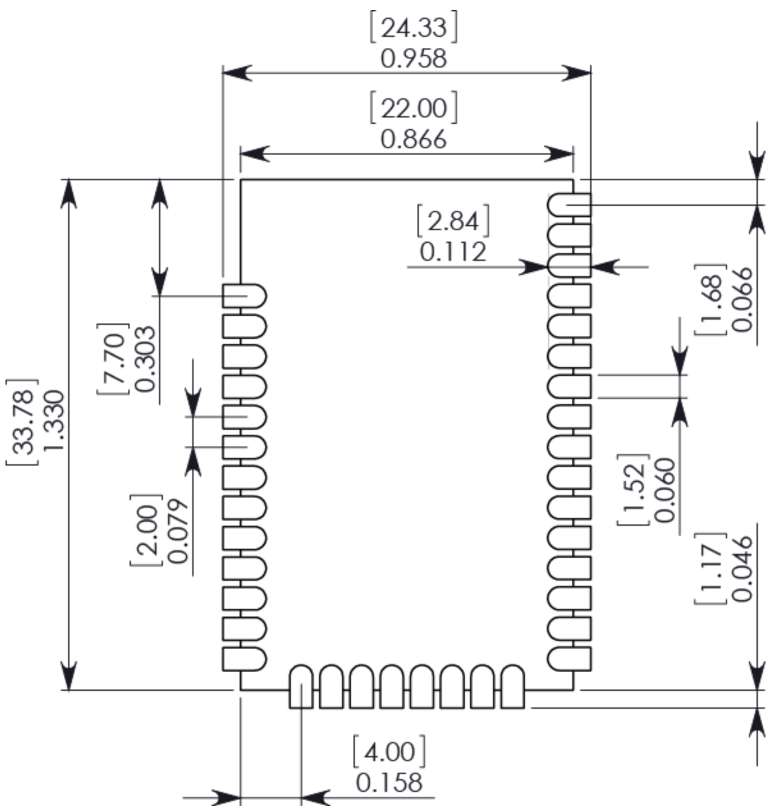
The device has a Moisture Sensitivity Level (MSL) of 3. When using this product, consider the relative requirements in accordance with standard IPC/JEDEC J-STD-020.

In addition, note the following conditions:

a. Calculated shelf life in sealed bag: 12 months at < 40 °C and < 90% relative humidity (RH).
b. Environmental condition during the production: 30 °C /60% RH according to IPC/JEDEC J-STD-033C, paragraphs 5 through 7.
c. The time between the opening of the sealed bag and the start of the reflow process cannot exceed 168 hours if condition b) is met.
d. Baking is required if conditions b) or c) are not met.
e. Baking is required if the humidity indicator inside the bag indicates a RH of 10% more.
f. If baking is required, bake modules in trays stacked no more than 10 high for 4-6 hours at 125 °C.

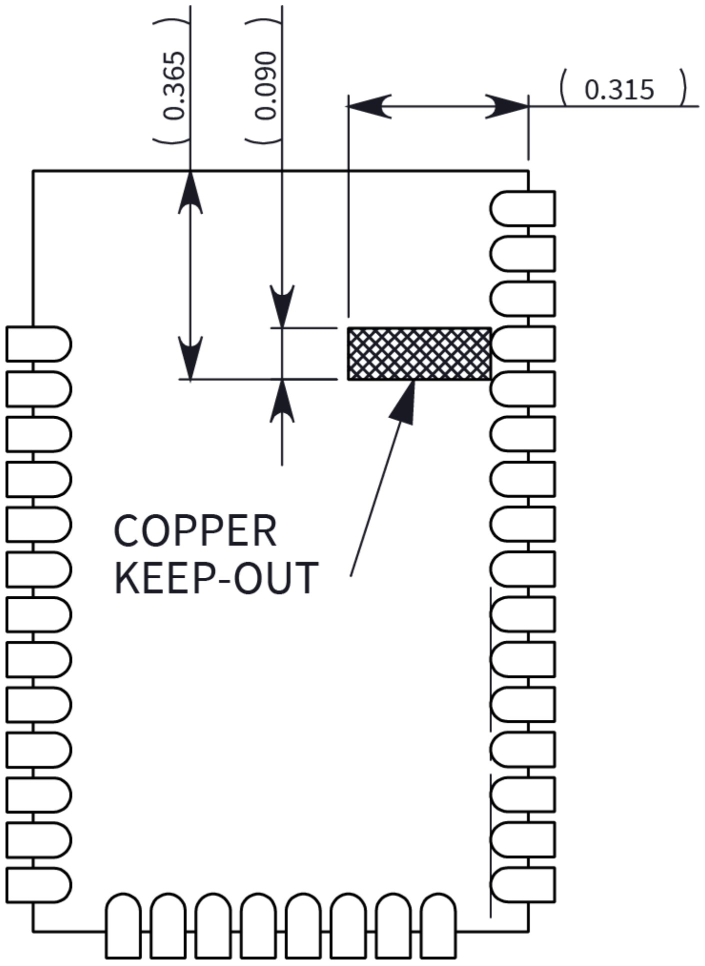
Recommended footprint and keepout

We recommend that you use the following PCB footprints for surface-mounting. The dimensions without brackets are in inches, and those in brackets are in millimeters.



Match the solder footprint to the copper pads, but you may need to adjust it depending on the specific needs of assembly and product standards. We recommend a stencil thickness of 0.15 mm (0.005 in). Place the component last and set the placement speed to the slowest setting.

While the underside of the module is mostly coated with solder resist, we recommend that the copper layer directly below the module be left open to avoid unintended contacts. Copper or vias must not interfere with the three exposed RF test points on the bottom of the module (see below). Furthermore, these modules have a ground plane in the middle on the back side for shielding purposes, which can be affected by copper traces directly below the module.



Flux and cleaning

We recommend that you use a “no clean” solder paste in assembling these devices. This eliminates the clean step and ensures that you do not leave unwanted residual flux under the device where it is difficult to remove. In addition:

- Cleaning with liquids can result in liquid remaining under the device or in the gap between the device and the host PCB. This can lead to unintended connections between pads.
- The residual moisture and flux residue under the device are not easily seen during an inspection process.

Rework

CAUTION! Any modification to the device voids the warranty coverage and certifications.

Rework should never be performed on the module itself. The module has been optimized to give the best possible performance, and reworking the module itself will void warranty coverage and certifications. We recognize that some customers will choose to rework and void the warranty; the following information is given as a guideline in such cases to increase the chances of success during rework, though the warranty is still voided.

The module may be removed from the OEM PCB by the use of a hot air rework station, or hot plate. Care should be taken not to overheat the module. During rework, the module temperature may rise above its internal solder melting point and care should be taken not to dislodge internal components from their intended positions.