XBee®/XBee-PRO® SE (Smart Energy) RF Modules

ZigBee SE (Smart Energy) RF Modules by Digi International

Firmware Versions:

- 31xx Coordinator - API Operation
- 33xx Router - API Operation
- 39xx End Device - API Operation
- 34xx Wall Router - API Operation

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

90033931_E
1/23/2012

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

Contents

Overview 6

Key Features 6

Worldwide Acceptance 7

Specifications 8

Mechanical Drawings 9

SIF Header Interface 10

Mounting Considerations 10

Pin Signals 11

EM250 Pin Mappings 13

Design Notes 13

Power Supply Design 13

Recommended Pin Connections 14

Board Layout 14

Electrical Characteristics 16

Smart Energy: What It Means to You 16

Module Operation 17

Serial Communications 17

UART Data Flow 17

Serial Buffers 17

Serial Flow Control 18

Serial Interface Protocols 18

Modes of Operation 19

Idle Mode 19

Transmit Mode 19

Receive Mode 21

Sleep Mode 21

XBee ZigBee Networks 22

Introduction to ZigBee 22

ZigBee Stack Layers 22

Networking Concepts 22

Device Types 22

PAN ID 23

Operating Channel 24

ZigBee Application Layers: In Depth 24

Application Support Sublayer (APS) 24

Application Profiles 24

Smart Energy Application Profile 25

Smart Energy Device Types 25

Smart Energy Clusters 27

Coordinator Operation 28

Forming a Network 28

Channel Selection 28

PAN ID Selection 28

Security Policy 28

Persistent Data 28

XBee Smart Energy Coordinator Startup 29

Permit Joining 29

XBee Device Registration 29

Resetting the Coordinator 31

Leaving a Network 31

Example: Starting a Coordinator 31

Router Operation 32

Discovering ZigBee Networks 32

Joining a Network 32

Authentication 32

Persistent Data 33

XBee Smart Energy Router Joining 33

Key Establishment 35

Permit Joining 37

Router Network Connectivity 37

Leaving a Network 37

Resetting the Router 37

Example: Joining a Network 37

Smart Energy Range Extender 38

End Device Operation 39

Discovering ZigBee Networks 39

Joining a Network 39

Parent Child Relationship 39

End Device Capacity 39

Authentication 40

Device Registration 40

Key Establishment 40

Persistent Data 40

Orphan Scans 40

XBee: Smart Energy End Device Joining 41

Parent Connectivity 42

Resetting the End Device 42

Leaving a Network 42

Example: Joining a Network 42

Channel Scanning 43

ZigBee and Smart Energy: Creating a Network 43

Discovery 44

Data, Addressing and Routing 47

Addressing 47

64-bit Device Addresses 47

16-bit Device Addresses 47

Contents

Application Layer Addressing 47

Data Transmission 47

Broadcast Transmissions 47

Unicast Transmissions 48

Data Transmission Examples 50

RF Packet Routing 50

Link Status Transmission 51

AODV Mesh Routing 52

Many-to-One Routing 54

Source Routing 54

Encrypted Transmissions 58

Maximum RF Payload Size 58

Throughput 58

ZDO Transmissions 59

ZigBee Device Objects (ZDO) 59

Sending a ZDO Command 60

Receiving ZDO Commands and Responses 60

Transmission Timeouts 61

Unicast Timeout 62

Extended Timeout 62

Transmission Examples 63

Security 65

Security Modes 65

ZigBee Security Model 65

Network Layer Security 65

Frame Counter 66

Message Integrity Code 66

Network Layer Encryption and Decryption 66

Network Key Updates 67

APS Layer Security 67

Message integrity Code 67

APS Link Keys 67

APS Layer Encryption and Decryption 68

Network and APS Layer Encryption 68

Trust Center 68

Forming and Joining a Secure Network 68

Implementing Security on the XBee 69

Enabling Security 69

Setting the Network Security Key 69

Setting the APS Trust Center Link Key 69

Enabling APS Encryption 69

Using a Trust Center 70

XBee Security Examples 70

Example 1: Forming a network with security (pre-configured link keys) 70

Example 2: Forming a network with security (obtaining keys during joining) 71

Managing End Devices 72

End Device Operation 72

Parent Operation 72

End Device Poll Timeouts 73

Packet Buffer Usage 73

Non-Parent Device Operation 74

XBee End Device Configuration 74

Pin Sleep 74

Cyclic Sleep 76

Transmitting RF Data 79

Receiving RF Data 79

Waking End Devices with the Commissioning Pushbutton 80

Parent Verification 80

Rejoining 80

XBee Router/Coordinator Configuration 81

RF Packet Buffering Timeout 81

Child Poll Timeout 81

Transmission Timeout 81

Putting it all Together 82

Short Sleep Periods 82

Extended Sleep Periods 82

Sleep Examples 82

Network Commissioning and Diagnostics 84

Device Discovery 84

ZDO Discovery 84

Joining Announce 84

Commissioning Pushbutton and Associate LED 84

Commissioning Pushbutton 85

Associate LED 85

API Operation 87

API Frame Specifications 87

API Examples 89

API UART Exchanges 89

AT Commands 89

Transmitting and Receiving RF Data 89

Source Routing 90

Supporting the API 90

API Frames 91

AT Command 91

Contents

AT Command - Queue Parameter Value 92
Explicit Addressing ZigBee Command Frame 92
ZigBee Create Source Route 95
ZigBee Register Joining Device 96
AT Command Response 97
Modem Status 97
ZigBee Transmit Status 98
ZigBee Explicit Rx Indicator 99
Route Record Indicator 100
ZigBee Device Authenticated Indicator 101
ZigBee Many-to-One Route Request Indicator 102
ZigBee Register Joining Device Status 103
ZigBee Join Notification Status 104

Sending ZigBee Device Objects (ZDO) Commands with the API 105

Sending ZigBee Cluster Library (ZCL) Commands with the API 107

Sending Public Profile Commands with the API 109

XBee Command Reference Tables 112

RF Module Support 120

X-CTU Configuration Tool 120

XBee Bootloader 120

Programming XBee Modules 120

Serial Firmware Updates 120
Invoke XBee Bootloader 121
Send Firmware Image 121
SIF Firmware Updates 122

Writing Custom Firmware 122

Regulatory Compliance 122
Enabling GPIO 1 and 2 122
Detecting XBee vs. XBee-PRO 123
Ensuring Optimal Output Power 123

Improving Low Power Current Consumption 124

XBee (non-PRO) Initialization: 124
When sleeping (end devices): 124
When waking from sleep (end devices): 124

Appendix A:Definitions 125

Appendix B: Agency Certifications 127

Appendix C:Migrating from XBee ZB to XBee SE 135

Appendix D:Appendix D: Smart Energy Certificates 136

Example 136

Appendix E:Additional Information 138

1. Overview

The XBee/XBee-PRO Smart Energy RF Modules are designed to support the implementation of Smart Energy Devices which operate within the ZigBee Smart Energy Application Profile. The modules require minimal power and provide reliable delivery of data between remote devices.

The XBee/XBee-PRO SE firmware release can be installed on XBee series 2 modules. The SE firmware is a firmware upgrade to XBee ZB or ZNet modules.

The XBee SE firmware is based on the EmberZNet 3.x ZigBee PRO Feature Set mesh networking stack and includes support for ECC encryption and key establishment as required for the Smart Energy profile. XBee SE modules must have an installed certificate from a certificate authority in order to join a network which is running with Authentication enabled, or to use APS encryption for peer-to-peer communication with a unique pair of link keys.

Key Features

High Performance, Low Cost

XBee

- Indoor/Urban: up to 133’ (40 m)
- Outdoor line-of-sight: up to 400’ (120 m)
- Transmit Power: 2 mW (3 dBm)
- Receiver Sensitivity: -96 dBm

XBee-PRO (S2)

- Indoor/Urban: up to 300’ (90 m), 200' (60 m) for International variant
- Outdoor line-of-sight: Up to 2 miles (3200 m), 5000' (1500 m) for International variant
- Transmit Power: 50mW (17dBm), 10mW (10dBm) for International variant
- Receiver Sensitivity: -102 dBm

XBee-PRO (S2B)

- Indoor/Urban: up to 300’ (90 m), 200’ (60 m) for International variant
- Outdoor line-of-sight: Up to 2 miles (3200 m), 5000’ (1500 m) for International variant
- Transmit Power: 63mW (18dBm), 10mW (10dBm) for International variant
- Receiver Sensitivity: -102 dBm

Advanced Networking & Security

- Retries and Acknowledgements
- DSSS (Direct Sequence Spread Spectrum)
- Each direct sequence channel has over 65,000 unique network addresses available
- Point-to-point, point-to-multipoint and peer-to-peer topologies supported
- Self-routing, self-healing and fault-tolerant mesh networking

Low Power

XBee

- TX Peak Current: 40 mA (@3.3 V)
- RX Current: 40 mA (@3.3 V)
- Power-down Current: < 1 uA

XBee-PRO (S2)

- TX Peak Current: 295mA (170mA for international variant)
- RX Current: 45 mA (@3.3 V)
- Power-down Current: 3.5µA typical @ 25 degrees C

XBee-PRO (S2B)

- TX Peak Current: 205mA (117mA for international version)
- RX Current: 47mA (@3.3 V)
- Power-down Current: 3.5µA typical 25 degrees C

Easy-to-Use

- No configuration necessary for out-of box RF communications
- API Command Mode for configuring module parameters
- Small form factor
- Extensive command set
- Free X-CTU software (Testing and configuration software)
- Free and unlimited technical support

Worldwide Acceptance

FCC Approval (USA) Refer to Appendix A for FCC Requirements.
Systems that contain XBee®/XBee-PRO® SE RF Modules inherit Digi Certifications.

ISM (Industrial, Scientific & Medical) 2.4 GHz frequency band

Manufactured under ISO 9001:2000 registered standards

XBee®/XBee-PRO® Smart Energy RF Modules are optimized for use in US, Canada, Europe, Australia, and Japan (contact Digi for complete list of agency approvals).

Specifications

Specifications of the XBee®/XBee-PRO® SE RF Module

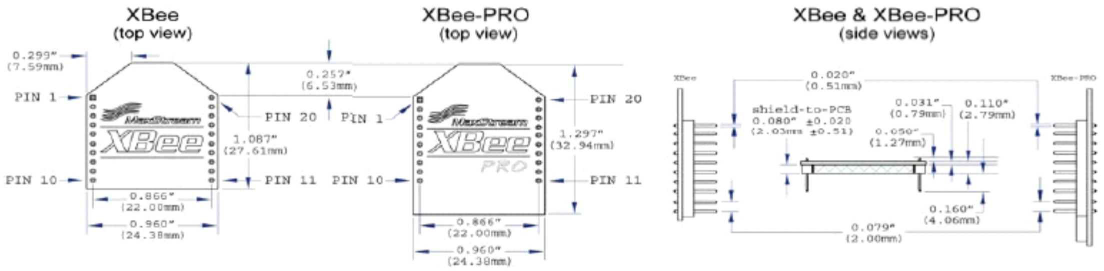
Specification	XBee	XBee-PRO (S2)	XBee-PRO (S2B)
Performance			
Indoor/Urban Range	up to 133 ft. (40 m)	Up to 300 ft. (90 m), up to 200 ft (60 m) international variant	Up to 300 ft. (90 m), up to 200 ft (60 m) international variant
Outdoor RF line-of-sight Range	up to 400 ft. (120 m)	Up to 2 miles (3200 m), up to 5000 ft (1500 m) international variant	Up to 2 miles (3200 m), up to 5000 ft (1500 m) international variant
Transmit Power Output	2mW (+3dBm), boost mode enabled 1.25mW (+1dBm), boost mode disabled	50mW (+17dBm) 10mW (+10dBm) international variant	63mW (+18dBm) 10mW (+10dBm) international variant
RF Data Rate	250,000 bps	250,000 bps	250,000 bps
Data Throughput	up to 35000 bps (see chapter 4)	up to 35000 bps (see chapter 4)	up to 35000 bps (see chapter 4)
Serial Interface Data Rate (software selectable)	1200 bps - 1 Mbps (non-standard baud rates also supported)	1200 bps - 1 Mbps (non-standard baud rates also supported)	1200 bps - 1 Mbps (non-standard baud rates also supported)
Receiver Sensitivity	-96dBm, boost mode enabled -95dBm, boost mode disabled	-102dBm	-102dBm
Power Requirements			
Supply Voltage	2.1 - 3.6 V	3.0 - 3.4 V	2.7 - 3.6 V
Operating Current (Transmit, max output power)	40mA (@ 3.3V, boost mode enabled) 35mA (@ 3.3V, boost mode disabled)	295mA (@3.3V), 170mA (@3.3V) international variant	205mA, up to 220mA with programmable variant (@3.3V) 217mA, up to 232mA with programmable variant (@3.3V)
Operating Current (Receive)	40mA (@ 3.3V, boost mode enabled) 38mA (@ 3.3V, boost mode disabled)	45mA (@3.3 V)	47mA, up to 62mA with programmable variant (@3.3V)
Idle Current (Receiver off)	15mA	15mA	15mA
Power-down Current	< 1μA @ 25^{o}C	3.5μA typical @ 25^{o}C	3.5μA typical @ 25^{o}C
General			
Operating Frequency Band	ISM 2.4 GHz	ISM 2.4 GHz	ISM 2.4 GHz
Dimensions	0.960” x 1.087” (2.438 cm x 2.761cm)	0.960 x 1.297 (2.438 cm x 3.294cm)	0.960 x 1.297 (2.438 cm x 3.294cm)
Operating Temperature	-40 to 85º C (industrial)	-40 to 85º C (industrial)	-40 to 85º C (industrial)
Antenna Options	Integrated Whip Antenna, Embedded PCB Antenna, RPSMA, or U.FL Connector	Integrated Whip Antenna, Embedded PCB Antenna, RPSMA, or U.FL Connector	Integrated Whip Antenna, Embedded PCB Antenna, RPSMA, or U.FL Connector
Networking & Security			
Supported Network Topologies	Point-to-point, Point-to-multipoint, Peer-to-peer, and Mesh	Point-to-point, Point-to-multipoint, Peer-to-peer, and Mesh	Point-to-point, Point-to-multipoint, Peer-to-peer, and Mesh
Number of Channels	16 Direct Sequence Channels	14 Direct Sequence Channels	15 Direct Sequence Channels
Addressing Options	PAN ID and Addresses, Cluster IDs and Endpoints (optional)	PAN ID and Addresses, Cluster IDs and Endpoints (optional)	PAN ID and Addresses, Cluster IDs and Endpoints (optional)
Agency Approvals			
United States (FCC Part 15.247)	FCC ID: OUR-XBEE2	FCC ID: MCQ-XBEEPRO2	FCC ID: MCQ-PROS2B
Industry Canada (IC)	IC: 4214A-XBEE2	IC: 1846A-XBEEPRO2	IC: 1846A-PROS2B
Europe (CE)	ETSI	ETSI	ETSI
Australia	C-Tick	C-Tick	C-Tick

Specifications of the XBee®/XBee-PRO® SE RF Module

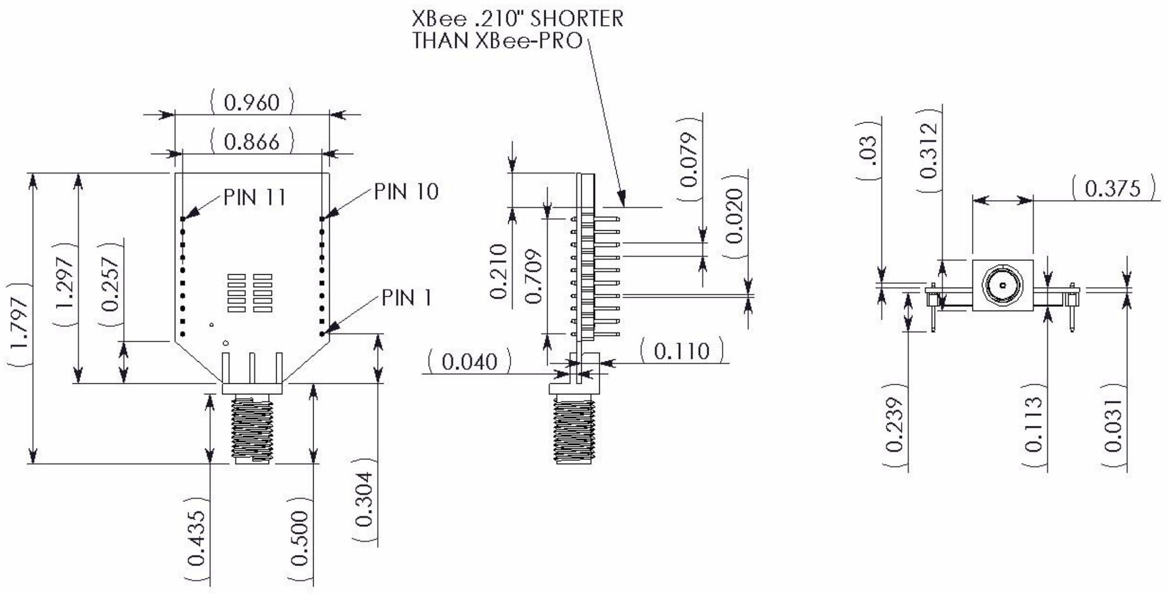
Specification	XBee	XBee-PRO (S2)	XBee-PRO (S2B)
Japan	R201WW07215214 Wire, chip, RPSMA, and U.FL versions are certified for Japan.The PCB antenna version is not.	R201WW08215142 (international variant) Wire, chip, RPSMA, and U.FL versions are certified for Japan. The PCB antenna version is not.	R201WW10215062
RoHS	Compliant	Compliant	Compliant

Mechanical Drawings

Mechanical drawings of the XBee®/XBee-PRO® SE RF Modules (antenna options not shown)

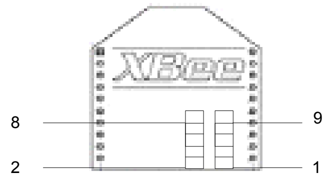


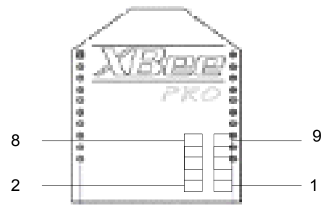
Mechanical Drawings for the RPSMA Variant



SIF Header Interface

The XBee/XBee-PRO Smart Energy modules include a SIF programming header that can be used with Ember's programming tools to upload custom firmware images onto the XBee module. The SIF header orientation and pinout are shown below.





This figure shows the orientation of the insight port header .

Pin Number	Pin Name
1	VBRD
2	SIF-MISO
3	Ground
4	SIF-MOSI
5	Ground
6	SIF-CLOCK
7	SIF-LOAD
8	RESET
9	PTI-EN
10	PTI-DATA

A male header can be populated on the XBee that mates with Ember's 2x5 ribbon cable. The male header and ribbon cables are available from Samtec:

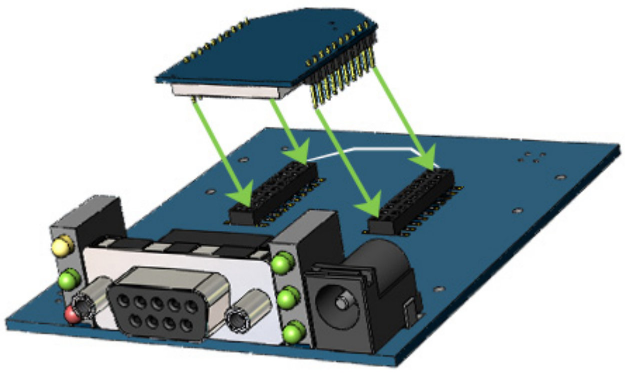
2x5 Male Header - FTSH-105-01-F-DV-K

2x5 Ribbon Cable - FFSD-05-D-12.00-01-N

Mounting Considerations

The XBee modules were designed to mount into a receptacle (socket) and therefore does not require any soldering when mounting it to a board. The XBee-PRO Development Kits contain RS-232 and USB interface boards which use two 20-pin receptacles to receive modules.

XBee-PRO Module Mounting to an RS-232 Interface Board.



The receptacles used on Digi development boards are manufactured by Century Interconnect. Several other manufacturers provide comparable mounting solutions; however, Digi currently uses the following receptacles:

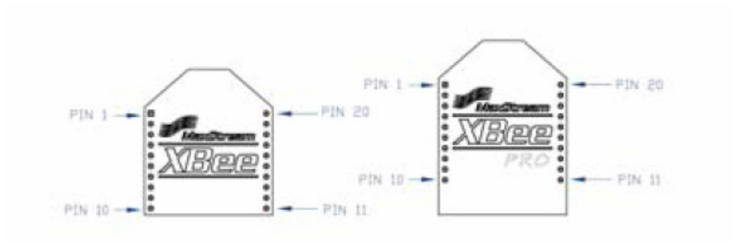
- Through-hole single-row receptacles - Samtec P/N: MMS-110-01-L-SV (or equivalent)
- Through-hole single-row receptacles - Mill-Max P/N: 831-43-0101-10-001000

- Surface-mount double-row receptacles -
  Century Interconnect P/N: CPRMSL20-D-0-1 (or equivalent)
- Surface-mount single-row receptacles -
  Samtec P/N: SMM-110-02-SM-S

Digi also recommends printing an outline of the module on the board to indicate the orientation the module should be mounted.

Pin Signals

XBee®/XBee-PRO® SE RF Module Pin Number
(top sides shown - shields on bottom)



Pin Assignments for the XBee-PRO Modules
(Low-asserted signals are distinguished with a horizontal line above signal name.)

Pin #	Name	Direction	Default State	Description
1	VCC	-	-	Power supply
2	DOUT	Output	Output	UART Data Out
3	DIN / $\overline{\textbf{CONFIG}}$	Input	Input	UART Data In
4	DIO12	Either	Disabled	Digital I/O 12
5	$\overline{\textbf{RESET}}$	Input	Open-Collector with pull-up	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI / DIO10	Either	Output	PWM Output 0 / RX Signal Strength Indicator / Digital IO
7	DIO11	Either	Input	Digital I/O 11
8	[reserved]	-	Disabled	Do not connect
9	$\overline{\text{DTR}}$ / SLEEP_RQ/ DIO8	Either	Input	Pin Sleep Control Line or Digital IO 8
10	GND	-	-	Ground
11	DIO4	Either	Disabled	Digital I/O 4
12	$\overline{\textbf{CTS}}$ / DIO7	Either	Output	Clear-to-Send Flow Control or Digital I/O 7. CTS, if enabled, is an output.
13	ON / $\overline{\textbf{SLEEP}}$	Output	Output	Module Status Indicator or Digital I/O 9
14	VREF	Input	-	Not used on this module. For compatibility with other XBee modules, we recommend connecting this pin to a voltage reference if Analog sampling is desired. Otherwise, connect to GND.
15	Associate / DIO5	Either	Output	Associated Indicator, Digital I/O 5
16	$\overline{\textbf{RTS}}$ / DIO6	Either	Input	Request-to-Send Flow Control, Digital I/O 6. RTS, if enabled, is an input.
17	AD3 / DIO3	Either	Disabled	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Disabled	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Disabled	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0 / Commissioning Button	Either	Disabled	Analog Input 0, Digital IO 0, or Commissioning Button

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

EM250 Pin Mappings

The following table shows how the EM250 pins are used on the XBee.

EM250 Pin Number	XBee Pin Number	Other Usage
13 (Reset)	5*	Connected to pin 8 on 2x5 SIF header.
19 (GPIO 11)	16*	
20 (GPIO 12)	12*	
21 (GPIO 0)	15	
22 (GPIO 1)		**XBee** Tied to ground (module identification) **XBee-PRO (S2)** Low-asserting shutdown line for output power compensation circuitry. **XBee-PRO (S2B)** Used to communicate with Temp Sensor and control Shutdown for low-power mode
24 (GPIO 2)		**XBee** Not connected. Configured as output low. **XBee-PRO (S2)** Powers the output power compensation circuitry. **XBee-PRO (S2B)** Used to communicate with Temp Sensor and control Shutdown for low-power mode
25 (GPIO 3)	13	
26 (GPIO 4 / ADC 0)	20	Connected to pin 9 on 2x5 SIF header.
27 (GPIO 5 / ADC 1)	19	Connected to pin 10 on 2x5 SIF header.
29 (GPIO 6 /ADC 2)	18	
30 (GPIO 7 / ADC 3	17	
31 (GPIO 8)	4	
32 (GPIO 9)	2*	
33 (GPIO 10)	3*	
34 (SIF_CLK)		Connected to pin 6 on 2x5 SIF header.
35 (SIF_MISO)		Connected to pin 2 on 2x5 SIF header.
36 (SIF_MOSI)		Connected to pin 4 on 2x5 SIF header.
37 (SIF_LOAD)		Connected to pin 7 on 2x5 SIF header.
40 (GPIO 16)	7	
41 (GPIO 15)	6	
42 (GPIO 14)	9	
43 (GPIO 13)	11	

*NOTE: These lines may not go to the external XBEE pins of the module if the programmable secondary processor is populated.

Design Notes

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

Power Supply Design

Poor power supply can lead to poor radio performance especially if the supply voltage is not kept within tolerance or is excessively noisy. To help reduce noise a 1uF and 8.2pF capacitor are recommended to be placed as near to pin 1 on the PCB as possible. If using a switching regulator for your power supply, switching frequencies above 500kHz are preferred. Power supply ripple should be limited to a maximum 250mV peak to peak.

Note: An additional 10uF decoupling cap is recommended near pin 1 of the module for designs using the programmable modules. The nearest proximity to pin 1 of the 3 caps should be in the following order: 8.2pF, 1uF followed by 10uF.

Recommended Pin Connections

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

All unused pins should be left disconnected. No specific treatment is needed for unused outputs.

Other pins may be connected to external circuitry for convenience of operation including the Associate LED pin (pin 15) and the Commissioning pin (pin 20). The Associate LED pin will flash differently depending on the state of the module to the network, and a pushbutton attached to pin 20 can enable various join functions without having to send UART commands. Please see the commissioning pushbutton and associate LED section in chapter 7 for more details. The source and sink capabilities are limited to 4mA for all pins on the module.

The VREF pin (pin 14) is not used on this module. For compatibility with other XBee modules, we recommend connecting this pin to a voltage reference if analog sampling is desired. Otherwise, connect to GND.

Board Layout

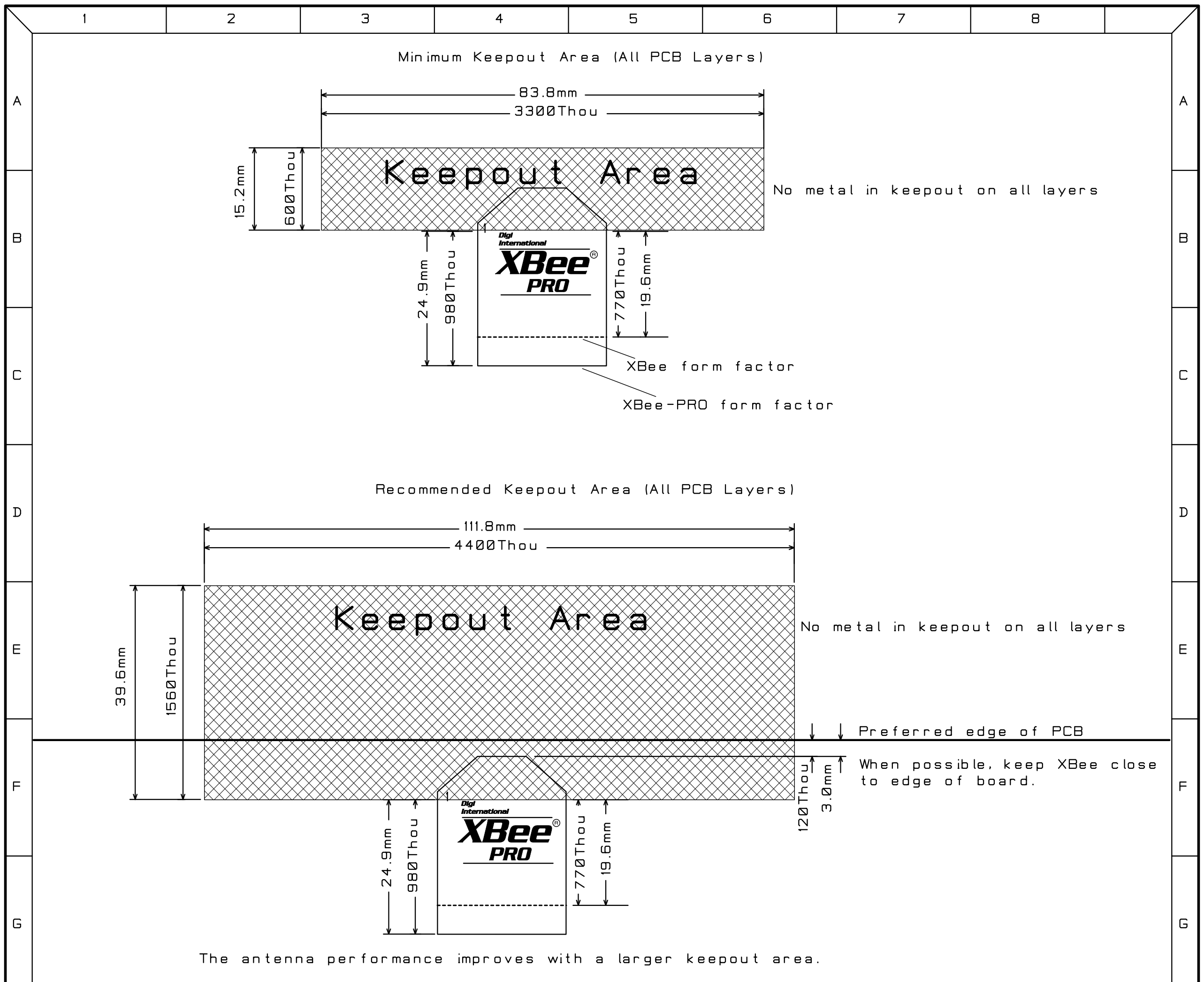
XBee modules do not have any specific sensitivity to nearby processors, crystals, or other PCB components. Other than mechanical considerations, no special PCB placement is required for integrating XBee radios except for those with integral antennas. In general, Power and GND traces should be thicker than signal traces and be able to comfortably support the maximum currents.

The radios are also designed to be self- sufficient and work with the integrated and external antennas without the need for additional ground planes on the host PCB. However, considerations should be taken on the choice of antenna and antenna location. Metal objects that are near an antenna cause reflections and may reduce the ability for an antenna to efficiently radiate. Using an integral antenna (like a wire whip antenna) in an enclosed metal box will greatly reduce the range of a radio. For this type of application, an external antenna would be a better choice.

External antennas should be positioned away from metal objects as much as possible. Metal objects next to the antenna or between transmitting and receiving antennas can often block or reduce the transmission distance. Some objects that are often overlooked are metal poles, metal studs or beams in structures, concrete (it is usually reinforced with metal rods), metal enclosures, vehicles, elevators, ventilation ducts, refrigerators, and microwave ovens.

The Wire Whip Antenna should be straight and perpendicular to the ground plane and/or chassis. It should reside above or away from any metal objects like batteries, tall electrolytic capacitors or metal enclosures. If the antenna is bent to fit into a tight space, it should be bent so that as much of the antenna as possible is away from metal. Caution should be used when bending the antenna, since this will weaken the solder joint where the antenna connects to the module. Antenna elements radiate perpendicular to the direction they point. Thus a vertical antenna emits across the horizon.

Embedded PCB antennas should not have any ground planes or metal objects above or below the module at the antenna location. For best results the module should be in a plastic enclosure, instead of metal one. It should be placed at the edge of the PCB to which it is mounted. The ground, power and signal planes should be vacant immediately below the antenna section (See drawing for recommended keepout area).


Minimum Keepout Area (All PCB Layers)
83.8mm
3300Thou
Keepout Area
No metal in keepout on all layers
15.2mm
600Thou
24.9mm
980Thou
770Thou
19.6mm
Digi International XBee PRO
XBee form factor
XBee-PRO form factor
Recommended Keepout Area (All PCB Layers)
111.8mm
4400Thou
Keepout Area
No metal in keepout on all layers
39.6mm
1560Thou
Preferred edge of PCB
When possible, keep XBee close to edge of board.
120Thou
3.0mm
24.9mm
980Thou
770Thou
19.6mm
The antenna performance improves with a larger keepout area.

Notes:
1. Non-metal enclosures are recommended. For metal enclosures, an external antenna should be used.
2. Metal chassis or mounting structures in the keepout area should be at least 1 inch(2.54 cm) from antenna.
3. Maximize distance between antenna and metal objects that might be mounted in 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.

REV	ECO	DESCRIPTION OF CHANGE	BY	CKD	APPR	DATE

APPROVALS:	DATE:	TITLE Keepout Area for Embedded PCB Antenna
DESIGNED:		
DRAWN:		
CHECKED:		
ENGINEER:		PART NO. / REV.
© Digi International Inc. All rights reserved		DO NOT SCALE DRAWING / SHEET 1 of 1

Electrical Characteristics

DC Characteristics of the XBee-PRO (VCC = 3.0 - 3.4 VDC).

Symbol	Parameter	Condition	Min	Typical	Max	Units
V_{IL}	Input Low Voltage	All Digital Inputs	-	-	0.2 * VCC	V
V_{IH}	Input High Voltage	All Digital Inputs	0.8 * VCC	-	-	V
V_{OL}	Output Low Voltage	I_{OL} = 2mA, VCC >= 2.7V	-	-	0.18 * VCC	V
V_{OH}	Output High Voltage	I_{OH} = -2mA, VCC >= 2.7V	0.82 * VCC	-	-	V
I_{IN}	Input Leakage Current	V_{IN} = VCC or GND, all inputs, per pin	-	-	0.5µA	µA
I_{OHS}	Output Source Current (standard)	All digital outputs except RSSI/PWM, DIO10, DIO4	-	-	4	mA
I_{OHH}	Output Source Current (high current)	RSSI/PWM, DIO10, DIO4 digital outputs	-	-	8	mA
I_{OLS}	Output Sink Current (standard)	All digital inputs except RSSI/PWM, DIO10, DIO4	-	-	4	mA
I_{OLH}	Output Sink Current (high current)	RSSI/PWM, DIO10, DIO4 digital inputs	-	-	8	mA
$I_{OH} + I_{OL}$	Total Output Current for all I/O pins	All digital outputs	-	-	40	mA
V_{REFI}	V_{REF} Internal	EM250 has an internal reference that is fixed	1.19	1.2	1.21	V
V_{IADC}	ADC Input Voltage Range		0	-	V_{REFI}	V
R_{IS}	Input Impedance	When taking a sample	1	-	-	MΩ
R_I	Input Impedance	When not taking a sample	10	-	-	MΩ

Smart Energy: What It Means to You

The concept of Smart Energy is to provide utility companies and consumers with the means to manage consumption with the power and flexibility of wireless networking.

A workgroup of the ZigBee Alliance, known as the Advanced Metering Initiative (AMI) workgroup, has developed the ZigBee Smart Energy Profile Specification.

A Smart Energy Network consists of a number of devices communicating over a ZigBee mesh network. Meters, thermostats, switches, appliances, and displays are equipped with wireless RF transceivers to exchange control and data messages with each other.

A special device manages the network which supports a non-ZigBee gateway interface with the utility company. Metering data is polled from the Smart Energy network by the utility company for billing purposes.

Pricing information, alert messages, and load requests are sent by the utility company to the network.

The consumer interacts with the network by sending control commands and reading messages from the devices and the utility company.

Security is an important aspect of Smart Energy. Special methods are used to protect the network from interference or tampering. The Smart Energy RF Module provides each module with means to communicate with the network's Trust Center. The Trust Center requires other nodes to authenticate themselves before allowing them to join the network.

See chapter 3 for details.

2. Module Operation

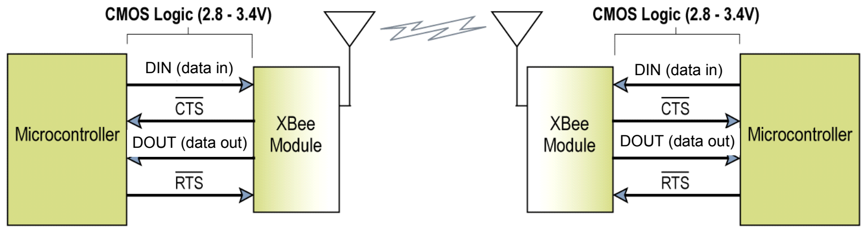
Serial Communications

The XBee modules interface to a host device through a logic-level asynchronous serial port. Through its serial port, the module can communicate with any logic and voltage compatible UART; or through a level translator to any serial device (for example: through a proprietary RS-232 or USB interface board).

UART Data Flow

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

System Data Flow Diagram in a UART-interfaced environment
(Low-asserted signals distinguished with horizontal line over signal name.)

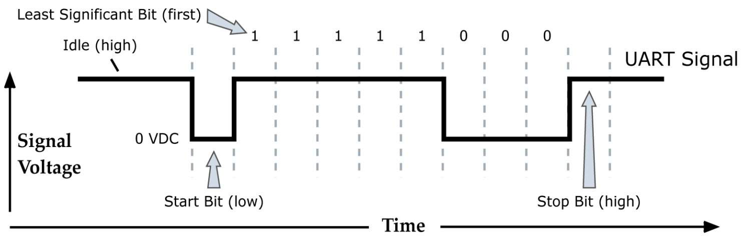


Serial Data

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

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

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



The module UART performs tasks, such as timing and parity checking, that are needed for data communications. Serial communications depend on the two UARTs to be configured with compatible settings (baud rate, parity, start bits, stop bits, data bits).

Serial Buffers

The XBee modules maintain small buffers to collect received serial and RF data, which is illustrated in the figure below. The serial receive buffer collects incoming serial characters and holds them until they can be processed. The serial transmit buffer collects data that is received via the RF link that will be transmitted out the UART.

The UART baud rate and parity on the XBee module can be configured with the BD and NB commands respectively. See the command table for details.

Serial Receive Buffer

When serial data enters the RF module through the DIN Pin (pin 3), the data is stored in the serial receive buffer until it can be processed. Under certain conditions, the module may not be able to process data in the serial receive buffer immediately. If large amounts of serial data are sent to the module, $\overline{\text{CTS}}$ flow control may be required to avoid overflowing the serial receive buffer.

Cases in which the serial receive buffer may become full and possibly overflow:

1. If the module is receiving a continuous stream of RF data, the data in the serial receive buffer will not be transmitted until the module is no longer receiving RF data.

2. If the module is transmitting an RF data packet, the module may need to discover the destination address or establish a route to the destination. After transmitting the data, the module may need to retransmit the data if an acknowledgment is not received, or if the transmission is a broadcast. These issues could delay the processing of data in the serial receive buffer.

Serial Transmit Buffer

When RF data is received, the data is moved into the serial transmit buffer and sent out the UART. If the serial transmit buffer becomes full enough such that all data in a received RF packet won't fit in the serial transmit buffer, the entire RF data packet is dropped.

Cases in which the serial transmit buffer may become full resulting in dropped RF packets

1. If the RF data rate is set higher than the interface data rate of the module, the module could receive data faster than it can send the data to the host.

2. If the host does not allow the module to transmit data out from the serial transmit buffer because of being held off by hardware flow control.

Serial Flow Control

The $\overline{\text{CTS}}$ module pin is used to provide $\overline{\text{CTS}}$ flow control. $\overline{\text{CTS}}$ flow control provides an indication to the host to stop sending serial data to the module. RTS flow control is not available. $\overline{\text{CTS}}$ flow control is always enabled.

When the serial receive buffer is 17 bytes away from being full, the module de-asserts $\overline{\text{CTS}}$ (sets it high) to signal to the host device to stop sending serial data. $\overline{\text{CTS}}$ is re-asserted after the serial receive buffer has 34 bytes of space.

The host device is expected to be able to receive data as fast as the Smart Energy RF Module supplies it. Otherwise potentially critical data packets may be discarded.

Serial Interface Protocols

The Smart Energy RF Module only supports the API (Application Programming Interface) serial interface.

API Operation

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

Transmit Data Frames (received through the DIN pin (pin 3)) include:

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

Receive Data Frames (sent out the DOUT pin (pin 2)) include:

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

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

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

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

RF modules that contain the following firmware versions will support API operation: 31xx (API Coordinator), 33xx (API Router), and 39xx (API end device).

Modes of Operation

Idle Mode

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

- Transmit Mode (Serial data in the serial receive buffer is ready to be packetized)
- Receive Mode (Valid RF data is received through the antenna)
- Sleep Mode (End Devices only)

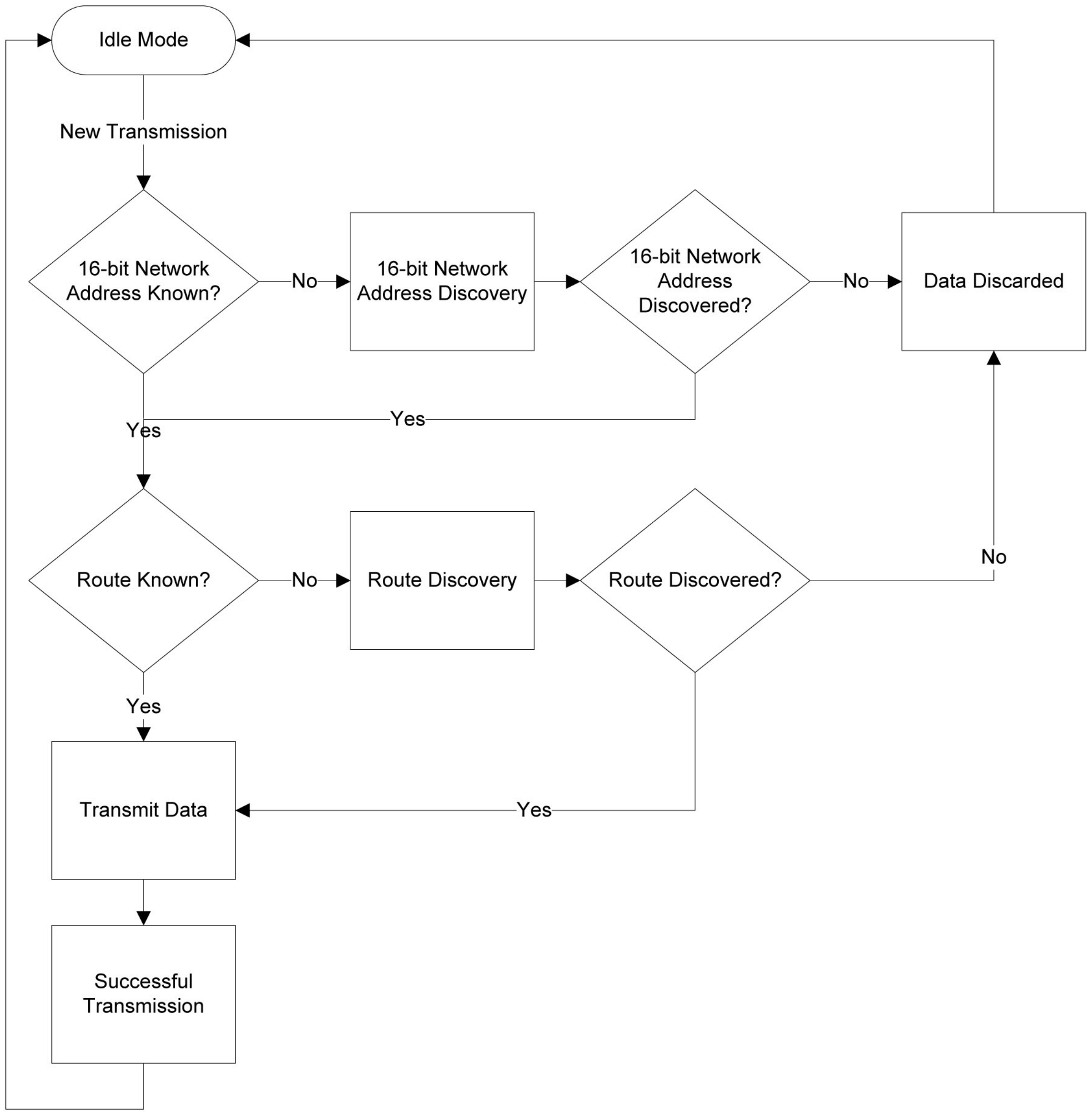
Transmit Mode

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

Prior to transmitting the data, the module ensures that a 16-bit network address and route to the destination node have been established.

If the destination 16-bit network address is not known, network address discovery will take place. If a route is not known, route discovery will take place for the purpose of establishing a route to the destination node. If a module with a matching network address is not discovered, the packet is discarded. The data will be transmitted once a route is established. If route discovery fails to establish a route, the packet will be discarded.

Transmit Mode Sequence



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

It is possible in rare circumstances for the destination to receive a data packet, but for the source to not receive the network acknowledgment. In this case, the source will retransmit the data, which could cause the destination to receive the same data packet multiple times. The XBee modules do not filter out duplicate packets. The application should include provisions to address this potential issue.

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

Receive Mode

If a valid RF packet is received, the data is transferred to the serial transmit buffer.

Sleep Mode

Sleep modes allow the end device to enter states of low power consumption when not in use. The end device supports support both pin sleep (sleep mode entered on pin transition) and cyclic sleep (module sleeps for a fixed time). XBee sleep modes are discussed in detail in chapter 6.

3. XBee ZigBee Networks

Introduction to ZigBee

ZigBee is an open global standard built on the IEEE 802.15.4 MAC/PHY. ZigBee defines a network layer above the 802.15.4 layers to support advanced mesh routing capabilities. The ZigBee specification is developed by a growing consortium of companies that make up the ZigBee Alliance. The Alliance is made up of over 300 members, including semiconductor, module, stack, and software developers.

ZigBee Stack Layers

The ZigBee stack consists of several layers including the PHY, MAC, Network, Application Support Sublayer (APS), and ZigBee Device Objects (ZDO) layers. Technically, an Application Framework (AF) layer also exists, but will be grouped with the APS layer in remaining discussions. The ZigBee layers are shown in the figure below.

A description of each layer appears in the following table:

ZigBee Layer	Description
PHY	Defines the physical operation of the ZigBee device including receive sensitivity, channel rejection, output power, number of channels, chip modulation, and transmission rate specifications. Most ZigBee applications operate on the 2.4 GHz ISM band at a 250kbps data rate. See the IEEE 802.15.4 specification for details.
MAC	Manages RF data transactions between neighboring devices (point to point). The MAC includes services such as transmission retry and acknowledgment management, and collision avoidance techniques (CSMA-CA).
Network	Adds routing capabilities that allows RF data packets to traverse multiple devices (multiple "hops") to route data from source to destination (peer to peer).
APS (AF)	Application layer that defines various addressing objects including profiles, clusters, and endpoints.
ZDO	Application layer that provides device and service discovery features and advanced network management capabilities.

Networking Concepts

Device Types

ZigBee defines three different device types: coordinator, router, and end device.

A **coordinator** has the following characteristics: it

- Selects a channel and PAN ID (both 64-bit and 16-bit) to start the network
- Can allow routers and end devices to join the network
- Can assist in routing data
- Cannot sleep--should be mains powered.

A **router** has the following characteristics: it

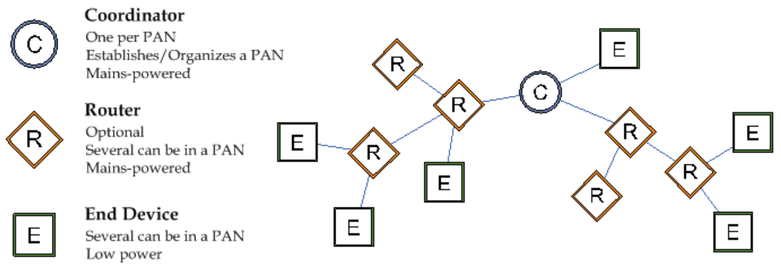
- Must join a ZigBee PAN before it can transmit, receive, or route data
- After joining, can allow routers and end devices to join the network

- After joining, can assist in routing data
- Cannot sleep--should be mains powered.

An **end device** has the following characteristics: it

- Must join a ZigBee PAN before it can transmit or receive data
- Cannot allow devices to join the network
- Must always transmit and receive RF data through its parent. Cannot route data.
- Can enter low power modes to conserve power and can be battery-powered.

An example of such a network is shown below:



In ZigBee networks, the coordinator must select a PAN ID (64-bit and 16-bit) and channel to start a network. After that, it behaves essentially like a router. The coordinator and routers can allow other devices to join the network and can route data.

After an end device joins a router or coordinator, it must be able to transmit or receive RF data through that router or coordinator. The router or coordinator that allowed an end device to join becomes the "parent" of the end device. Since the end device can sleep, the parent must be able to buffer or retain incoming data packets destined for the end device until the end device is able to wake and receive the data.

PAN ID

ZigBee networks are called personal area networks or PANs. Each network is defined with a unique PAN identifier (PAN ID). This identifier is common among all devices of the same network. ZigBee devices are either preconfigured with a PAN ID to join, or they can discover nearby networks and select a PAN ID to join.

ZigBee supports both a 64-bit and a 16-bit PAN ID. Both PAN IDs are used to uniquely identify a network. Devices on the same ZigBee network must share the same 64-bit and 16-bit PAN IDs. If multiple ZigBee networks are operating within range of each other, each should have unique PAN IDs.

The 16-bit PAN ID is used as a MAC layer addressing field in all RF data transmissions between devices in a network. However, due to the limited addressing space of the 16-bit PAN ID (65,535 possibilities), there is a possibility that multiple ZigBee networks (within range of each other) could use the same 16-bit PAN ID. To resolve potential 16-bit PAN ID conflicts, the ZigBee Alliance created a 64-bit PAN ID.

The 64-bit PAN ID (also called the extended PAN ID), is intended to be a unique, non-duplicated value. When a coordinator starts a network, it can either start a network on a preconfigured 64-bit PAN ID, or it can select a random 64-bit PAN ID. The 64-bit PAN ID is used during joining; if a device has a preconfigured 64-bit PAN ID, it will only join a network with the same 64-bit PAN ID. Otherwise, a device could join any detected PAN and inherit the PAN ID from the network when it joins. The 64-bit PAN ID is included in all ZigBee beacons and is used in 16-bit PAN ID conflict resolution.

Routers and end devices are typically configured to join a network with any 16-bit PAN ID as long as the 64-bit PAN ID is valid. Coordinators typically select a random 16-bit PAN ID for their network.

Since the 16-bit PAN ID only allows up to 65,535 unique values, and since the 16-bit PAN ID is randomly selected, provisions exist in ZigBee to detect if two networks (with different 64-bit PAN

IDs) are operating on the same 16-bit PAN ID. If such a conflict is detected, the ZigBee stack can perform PAN ID conflict resolution to change the 16-bit PAN ID of the network in order to resolve the conflict. See the ZigBee specification for details.

To summarize, ZigBee routers and end devices should be configured with the 64-bit PAN ID of the network they want to join. They typically acquire the 16-bit PAN ID when they join a network.

Operating Channel

ZigBee utilizes direct-sequence spread spectrum modulation and operates on a fixed channel. The 802.15.4 PHY defines 16 operating channels in the 2.4 GHz frequency band. XBee modules support all 16 channels (11-26), XBee-PRO (S2) modules support 14 of the 16 channels (11-24), and XBee-PRO (S2B) modules support 15 of the 16 channels (11-25).

ZigBee Application Layers: In Depth

This section provides a more in-depth look at the ZigBee application stack layers (APS, ZDO) including a discussion on ZigBee endpoints, clusters, and profiles.

Application Support Sublayer (APS)

The APS layer in ZigBee adds support for application profiles, cluster IDs, and endpoints.

Application Profiles

Application profiles specify various device descriptions including required functionality for various devices. The collection of device descriptions forms an application profile. Application profiles can be defined as "Public" or "Private" profiles. Private profiles are defined by a manufacturer whereas public profiles are defined, developed, and maintained by the ZigBee Alliance. Each application profile has a unique profile identifier assigned by the ZigBee Alliance.

Examples of public profiles include:

- Home Automation
- Smart Energy
- Commercial Building Automation

The Smart Energy profile, for example, defines various device types including an energy service portal, load controller, thermostat, in-home display, etc. The Smart Energy profile defines required functionality for each device type. For example, a load controller must respond to a defined command to turn a load on or off. By defining standard communication protocols and device functionality, public profiles allow interoperable ZigBee solutions to be developed by independent manufacturers.

Clusters

A cluster is an application message type defined within a profile. Clusters are used to specify a unique function, service, or action. For example, the following are some clusters defined in the home automation profile:

- On/Off - Used to switch devices on or off (lights, thermostats, etc.)
- Level Control - Used to control devices that can be set to a level between on and off
- Color Control - Controls the color of color capable devices.

Each cluster has an associated 2-byte cluster identifier (cluster ID). The cluster ID is included in all application transmissions. Clusters have associated attributes and commands that together define functionality. For example, a Smart Energy gateway (service portal) might send a Load Control Event command to a load controller in order to schedule turning on or off an appliance. Upon executing the event, the load controller would send a Report Event Status command back to the gateway.

Devices that operate in an application profile (private or public) must respond correctly to all required clusters. For example, a light switch that will operate in the home automation public profile must correctly implement the On/Off and other required clusters in order to interoperate with other home automation devices. The ZigBee Alliance has defined a ZigBee Cluster Library (ZCL) that contains definitions or various general use clusters that could be implemented in any profile.

XBee modules implement various clusters in the Digi private profile. In addition, the API can be used to send or receive messages on any cluster ID (and profile ID or endpoint). See the Explicit Addressing ZigBee Command API frame in chapter 9 for details.

Endpoints

The APS layer includes supports for endpoints. An endpoint can be thought of as a running application, similar to a TCP/IP port. A single device can support one or more endpoints. Each application endpoint is identified by a 1-byte value, ranging from 1 to 240. Each defined endpoint on a device is tied to an application profile. A device could, for example, implement one endpoint that supports a Smart Energy load controller, and another endpoint that supports other functionality on a private profile.

ZigBee Device Profile

Profile ID 0x0000 is reserved for the ZigBee Device Profile. This profile is implemented on all ZigBee devices. Device Profile defines many device and service discovery features and network management capabilities. Endpoint 0 is a reserved endpoint that supports the ZigBee Device Profile. This endpoint is called the ZigBee Device Objects (ZDO) endpoint.

ZigBee Device Objects (ZDO)

The ZDO (endpoint 0) supports the discovery and management capabilities of the ZigBee Device Profile. A complete listing of all ZDP services is included in the ZigBee specification. Each service has an associated cluster ID.

The XBee Smart Energy firmware allows applications to easily send ZDO messages to devices in the network using the API. See the ZDO Transmissions section in chapter 4 for details.

Smart Energy Application Profile

A working group of the ZigBee Alliance, known as the Advanced Metering Initiative (AMI) working group, has developed the ZigBee Smart Energy profile specification. A Smart Energy network consists of a number of devices communicating over a ZigBee mesh network. Meters, thermostats, switches, appliances, and displays can be equipped with wireless RF transceivers to exchange control and data messages with each other.

A special device manages the network which can support a non-ZigBee gateway interface with the utility company. Metering data can be polled from the Smart Energy network by the utility company for billing purposes. Pricing information, alert messages, and load requests can also be sent by the utility company to the network. The consumer can interact with the network by sending control commands and reading messages from the devices and the utility company.

The Smart Energy profile includes advanced security requirements to ensure that only authorized devices join the network. It also includes provisions to support data encryption, integrity, and authentication.

Smart Energy Device Types

- The Smart Energy profile defines the following device types:
- Energy service portal
- Metering device
- In-Premise display device
- Programmable communicating thermostat (PCT) device
- Load control device
- Range extender
- Smart appliance
- Prepayment terminal.

Details on each are provided below. (See the Smart Energy specification for implementation details.)

Energy Service Portal

The Energy Service Portal (ESP) acts as a ZigBee Coordinator and Trust Center to form and manage a Smart Energy (SE) network. The ESP acts as a router to convey wireless messages among the other devices in the network. The ESP may physically reside within another SE device (like a metering device) or exist as a standalone device. The ESP provides a backhaul or gateway connection to the utility company. The backhaul connection may be implemented by non-ZigBee protocols and communicate via RF, Ethernet, or some other means.

Metering Device

The Metering device measures the use of a consumable provided by a utility company or other provider (electricity, gas, water, heat). The device responds to polling requests for a reading. It can also be requested to provide periodic readings to a requesting device. A utility provider sends a request to an ESP, which relays the request to the Metering device. The Metering device responds with a reading, which the ESP relays to the utility. Consumers may use In-Premise Display devices to monitor the Metering devices in their respective networks.

In-Premise Display Device

The In-Premise Display device relays meter readings to the consumer with a graphical or text display. These readings may include: reading level, use over selectable periods of time, pricing information (tier rates for night, day, peak load times), text messages (there will be a demand response and load control event occurring between 2 and 4 pm this afternoon). It may be interactive - there is provision for an acknowledgement button for special messages.

Programmable Communicating Thermostat (PCT) Device

The PCT is a wireless thermostat which may be used by the consumer to opt in or opt out of demand response and load control (DR+LC) events. It will receive DR+LC requests from the utility. If the user has "opted in", then the thermostat will temporarily adjust its setpoint by moving its threshold up a few degrees to ease the power demand from air conditioners.

Load Control Device

The Load Control Device is a kind of smart plug. It responds to DR+LC events by reducing duty cycles or switching off non-essential equipment during peak load times as defined by the utility provider. Again, the consumer may choose to opt-in or opt-out of those event requests.

Range Extender Device

A Range Extender Device is an extension of a ZigBee Router. It relays messages within the SE network among the devices.

Smart Appliance Device

A Smart Appliance Device responds to price messages from the utility, and may generate messages for consumer display. A washer might switch to colder water if gas or electricity costs are running high. A dishwasher might report its cycle status periodically. Freezers might report over temperature alarms.

Prepayment Terminal Device

The Prepayment Terminal Device definition is TBD at this time. The intent is to provide consumers with means to prepay their utility bills in increments rather than through a billing agreement. This is more common in Europe and developing countries than in the United States. It accepts payment by a card swipe or digit entry, displays the balance, generates message alerts when the balance runs low, and displays network messages.

Smart Energy Clusters

This section describes the clusters which are referenced by the Smart Energy Application Profile. References to their detailed description appear in their respective subsections. The customer must implement these clusters (except for the Key Establishment cluster) in the external coprocessor, communicating with the Smart Energy Generic Module across the serial UART port.

Implementation details on each cluster can be found in either the ZigBee Cluster Library (ZCL) or ZigBee Smart Energy profile specification.

The following sections elaborate briefly on Smart Energy clusters, and their cluster identifier values.

Basic - 0x0000

This cluster is used for obtaining device information, enabling a device, and resetting it remotely to factory defaults.

Time - 0x000A

This cluster provides an interface to a real-time clock. Smart Energy devices are expected to synchronize their real time clocks with the ESP at a rate of no more than once per 24 hours, and maintain agreement to within a minute.

Key Establishment - 0x0800

This cluster is responsible for managing secure communications between Smart Energy devices. The Smart Energy profile imposes an authentication process after joining has occurred. Once a device authenticates itself with the Trust Center, it can communicate with other nodes in the network.

Price - 0x0700

This cluster is used to communicate pricing data for Gas, Energy, or Water. The data is distributed to the ESP from the utility company. The ESP then publishes the data to the local network, so the consumer (and suitably configured smart devices) may take advantage of schedules and tiered pricing to optimize the costs of consumption.

Demand Response and Load Control - 0x0701

On the server side, commands are defined for creating and cancelling load control events. Load control events are used to schedule requested changes to duty cycles, temperature setpoints, or load shedding for specified classes of devices. These commands may originate with the consumer or the utility.

On the client side, responses are made which indicate if a device will "opt-in" or "opt-out" to participate in a load control event. For example, a consumer may configure a medical device to opt-out of participating in a load shedding event, but allow an air conditioner to "opt-in".

Simple Metering - 0x0702

This cluster is used by the consumer and the utility to poll metering devices for consumption data. Metering may be extended to several types of meters: electric, gas, water, heat, cooling, etc.

Message - 0x0703

This cluster is used to pass utility text messages from to the ESP to devices on the network, or to make them available to devices which may poll the ESP at a later time.

Complex Metering - 0x0704, and Pre-payment - 0x0705

Currently these clusters are TBD in the Smart Energy Profile specification.

Coordinator Operation

Forming a Network

The coordinator is responsible for selecting the channel, PAN ID (16-bit and 64-bit), security policy, and stack profile for a network. Since a coordinator is the only device type that can start a network, each ZigBee network must have one coordinator. After the coordinator has started a network, it can allow new devices to join the network. It can also route data packets and communicate with other devices on the network. In a Smart Energy network, the coordinator is typically the trust center.

To ensure the coordinator starts on a good channel and unused PAN ID, the coordinator performs a series of scans to discover any RF activity on different channels (energy scan) and to discover any nearby operating PANs (PAN scan). The process for selecting the channel and PAN ID are described in the following sections.

Channel Selection

When starting a network, the coordinator must select a "good" channel for the network to operate on. To do this, it performs an energy scan on multiple channels (frequencies) to detect energy levels on each channel. Channels with excessive energy levels are removed from its list of potential channels to start on.

PAN ID Selection

After completing the energy scan, the coordinator scans its list of potential channels (remaining channels after the energy scan) to obtain a list of neighboring PANs. To do this, the coordinator sends a beacon request (broadcast) transmission on each potential channel. All nearby coordinators and routers (that have already joined a ZigBee network) will respond to the beacon request by sending a beacon back to the coordinator. The beacon contains information about the PAN the device is on, including the PAN identifiers (16-bit and 64-bit). This scan (collecting beacons on the potential channels) is typically called an active scan or PAN scan.

After the coordinator completes the channel and PAN scan, it selects a random channel and unused 16-bit PAN ID to start on.

Security Policy

The security policy determines which devices are allowed to join the network, and which device(s) can authenticate joining devices. See chapter 5 for a detailed discussion of various security policies.

Persistent Data

Once a coordinator has started a network, it retains the following information through power cycle or reset events:

- PAN ID
- Operating channel
- Security policy and frame counter values
- Child table (end device children that are joined to the coordinator).

The coordinator will retain this information indefinitely until it leaves the network. When the coordinator leaves a network and starts a new network, the previous PAN ID, operating channel, and child table data are lost.

XBee Smart Energy Coordinator Startup

The following commands control the coordinator network formation process.

Network formation commands used by the coordinator to form a network.

Command	Description
ID	Used to determine the 64-bit PAN ID. If set to 0 (default), a random 64-bit PAN ID will be selected.
SC	Determines the scan channels bitmask (up to 16 channels) used by the coordinator when forming a network. The coordinator will perform an energy scan on all enabled SC channels. It will then perform a PAN ID scan and then form the network on one of the SC channels.
SD	Set the scan duration period. This value determines how long the coordinator performs an energy scan or PAN ID scan on a given channel.
NK	Set the network security key for the network. If set to 0 (default), a random network security key will be used.
KY	Set the trust center link key for the network. If set to 0 (default), a random link key will be used.

Once the coordinator starts a network, the network configuration settings and child table data persist through power cycles as mentioned in the "Persistent Data" section.

When the coordinator has successfully started a network, it

- Allows other devices to join the network for a time (see NJ command.)
- Sets AI=0
- Starts blinking the Associate LED
- Sends an API modem status frame ("coordinator started") out the UART (API firmware only).

These behaviors are configurable using the following commands:

Command	Description
NJ	Sets the permit-join time on the coordinator, measured in seconds.

If any of the command values in the network formation commands table changes, the coordinator will leave its current network and start a new network, possibly on a different channel. Note that command changes must be applied (AC command) before taking effect.

Permit Joining

The permit joining attribute on the coordinator is configurable with the NJ command. Joining cannot be permanently enabled in a Smart Energy network.

Joining Enabled

If the value of NJ is nonzero, then joining will be enabled for that many seconds. If NJ is set to 0xFF, then joining is permanently enabled. This is not a recommended security practice as it weakens security. If NJ is zero, then joining is disabled unless the commissioning button is pressed twice, in which case joining is enabled for one minute. The timer is started once the XBee joins a network. Joining will not be re-enabled if the module is power cycled or reset. The following mechanisms can restart the permit-joining timer:

- Changing NJ to a different value (and applying changes with the AC or CN commands)
- Pressing the commissioning button twice
- Issuing the CB command with a parameter of 2 in which case software emulation of a 2 button press will occur.

XBee Device Registration

Up to 10 devices can be registered with the XBee coordinator (trust center) using the Register Joining Device API frame (0x24). Registering a device informs the trust center of the 64-bit address and initial link key of a device that is authorized to join the network. (In some cases, the device's initial link key may be derived from an installation code.) Devices that are removed from the network should likewise be de-registered or removed from the trust center using the same API frame.

Example 1 - Registering a Device

A router with a 64-bit address of 0x0013A200404C15A6 wants to join the network with a link key of 0x1. The following API frame can be sent to the XBee coordinator (trust center) to register the device.

Raw:

7E 00 0E 24 01 0013A200 404C15A6 FFFE 00 01 E0

Decoded:

0x7E Start delimiter

0x000E Length (all bytes after length, excluding checksum)

0x24 API frame type (Register Joining Device)

0x01 Frame ID (arbitrarily selected. Set >0 to get a status response.)

0x0013A200 404C15A6 64-bit address of joining device

0xFFFE 16-bit address (set to 0xFFFE)

0x00 Key Options (set to 0)

0x01 Key (up to 16 bytes, leading 0's can be omitted)

0xE0 Checksum

If successful, the XBee would respond with:

Raw:

7E 0003 A4 01 00 5A

Decoded:

0x7E Start delimiter

0x0003 Length (all bytes after length, excluding checksum)

0xA4 API frame type (Register Joining Device Status)

0x01 Frame ID (matches frame ID of the request)

0x00 Status (success)

0x5A Checksum

Example 2 - Removing a Device

Removing a device from the PAN is a 2 step process. The device must first be told to leave the network, and then it should be removed from the trust center's key table.

Suppose we want to remove a router with a 64-bit address of 0x0013A200404C15A5 from the network. The explicit transmit API frame (0x11) can be used to send a ZDO Leave Request ([4], 2.4.3.3.5) as shown below:

Raw:

7E 001D 11 01 0013A200404C15A5 FFFE 00 00 0034 0000 00 00 A5154C4000A21300 00 C6

Decoded:

0x7E Start delimiter

0x001D Length

0x11 API frame type (Explicit Addressing ZigBee Command Frame)

0x01 Frame ID (arbitrarily selected. Set > 0 to get a status response)

0x0013A200404C15A564-bit address of destination device

0xFFFE16-bit address of destination device (unknown)

0x00 Source Endpoint

0x00 Destination Endpoint

0x0034 Cluster Id

0x0000 Profile Id (ZDO)

0x00 Broadcast radius - use maximum hops

0x00 Options

0xA5154C4000A21300   Device Address in little endian

0x00 Do not remove child devices (if any)

0xC6 Checksum

Then, to remove the router from the trust center's key table, the following Register Joining Device API frame can be sent:

Raw:

7E 00 0D 24 01 0013A200 404C15A6 FFFE 00 E1

Decoded:

0x7E Start delimiter

0x000 Length (number of bytes after length, excluding checksum)

0x24 API frame type (Register Device)

0x01 Frame ID (arbitrarily selected. Set >0 to get a status response)

0x0013A200404C15A6 64-bit address of device to remove

0xFFFE 16-bit address (set to 0xFFFE)

0x00 Key Options (set to 0)

0xE2 Checksum

Resetting the Coordinator

When the coordinator is reset or power cycled, it checks its PAN ID, operating channel and stack profile against the network configuration settings (ID, CH). If the coordinator's PAN ID, operating channel, or security policy is not valid based on its network and security configuration settings, then the coordinator will leave the network and attempt to form a new network based on its network formation command values.

To prevent the coordinator from leaving an existing network, the WR command should be issued after all network formation commands have been configured in order to retain these settings through power cycle or reset events.

Leaving a Network

There are a couple of mechanisms that will cause the coordinator to leave its current PAN and start a new network based on its network formation parameter values. These include the following:

- Change the ID command such that the current 64-bit PAN ID is invalid.
- Change the SC command such that the current channel (CH) is not included in the channel mask.
- Change the KY command value.
- Issue the NR0 command to cause the coordinator to leave.
- Press the commissioning button 4 times or issue the CB command with a parameter of 4.

Note that changes to ID, SC, and security command values only take effect when changes are applied (AC or CN commands).

Example: Starting a Coordinator

1. Set SC and ID to the desired scan channels and PAN ID values. (The defaults should suffice.)

2. If SC or ID is changed from the default, issue the WR command to save the changes.

3. If SC or ID is changed from the default, apply changes (make SC and ID changes take effect) by sending the AC command.

4. The Associate LED will start blinking once the coordinator has selected a channel and PAN ID.

5. The API Modem Status frame ("Coordinator Started") is sent out the UART (API firmware only).

6. Reading the AI command (association status) will return a value of 0, indicating a successful startup.

7. Reading the MY command (16-bit address) will return a value of 0, the ZigBee-defined 16-bit address of the coordinator.

After startup, the coordinator will allow joining based on its NJ value.

Router Operation

Routers must discover and join a valid ZigBee network before they can participate in a ZigBee network. After a router has joined a network, it can allow new devices to join the network. It can also route data packets and communicate with other devices on the network.

Discovering ZigBee Networks

To discover nearby ZigBee networks, the router performs a PAN (or active) scan, just like the coordinator does when it starts a network. During the PAN scan, the router sends a beacon request (broadcast) transmission on the first channel in its scan channels list. All nearby coordinators and routers operating on that channel (that are already part of a ZigBee network) respond to the beacon request by sending a beacon back to the router. The beacon contains information about the PAN the nearby device is on, including the PAN identifier (PAN ID), and whether or not joining is allowed. The router evaluates each beacon received on the channel to determine if a valid PAN is found. A router considers a PAN to be valid if the PAN:

- Has a valid 64-bit PAN ID (PAN ID matches ID if ID > 0)
- Has the correct stack profile (ZS command)
- Is allowing joining.

If a valid PAN is not found, the router performs the PAN scan on the next channel in its scan channels list and continues scanning until a valid network is found, or until all channels have been scanned. If all channels have been scanned and a valid PAN was not discovered, all channels will be scanned again.

In the Smart Energy profile, devices cannot attempt joining repeatedly. To be compliant to Smart Energy requirements, the XBee performs up to three channel scans. If all three attempts fail, the application or user must retry joining. The following events cause the XBee to perform up to three join attempts:

- Single commissioning button press (see chapter 7)
- CB command with a parameter of 1
- Resetting the XBee (FR or hardware reset).

Joining a Network

Once the router discovers a valid network, it sends an association request to the device that sent a valid beacon requesting a join on the ZigBee network. The device allowing the join then sends an association response frame that either allows or denies the join.

When a router joins a network, it receives a 16-bit address from the device that allowed the join. The 16-bit address is randomly selected by the device that allowed the join.

After joining a network, the router sends a broadcast ZDO device announce message advertising its 64-bit and 16-bit addresses.

Authentication

In a Smart Energy network, the router must then go through an authentication process. See the Security chapter for a discussion on security and authentication.

After the router is joined (and authenticated, in a secure network), it can allow new devices to join the network.

Persistent Data

Once a router has joined a network, it retains the following information through power cycle or reset events:

- PAN ID
- Operating channel
- Security policy and frame counter values
- Child table (end device children that are joined to the coordinator).

The router will retain this information indefinitely until it leaves the network. When the router leaves a network, the previous PAN ID, operating channel, and child table data are lost.

XBee Smart Energy Router Joining

When the router is powered on, if it is not already joined to a valid ZigBee network, it immediately attempts to find and join a valid ZigBee network.

The following commands control the router joining process.

Command	Description
ID	Sets the 64-bit PAN ID to join. Setting ID=0 allows the router to join any 64-bit PAN ID.
SC	Set the scan channels bitmask that determines which channels a router will scan to find a valid network. SC on the router should be set to match SC on the coordinator. For example, setting SC to 0x281 enables scanning on channels 0x0B, 0x12, and 0x14, in that order.
EO	Configures whether or not the device should initiate key establishment after joining.
SD	Set the scan duration, or time that the router will listen for beacons on each channel.
KY	If the trust center link key is known, KY on the router can be set to match the trust center link key

Once the router joins a network, the network configuration settings and child table data persist through power cycles as mentioned in the "Persistent Data" section previously. If joining fails, the status of the last join attempt can be read in the AI command register.

If any of the above command values change, when command register changes are applied (AC or CN commands), the router will leave its current network and attempt to discover and join a new valid network.

When a Smart Energy router has successfully joined a network, it:

- Allows other devices to join the network for a time
- Sets AI=0
- Starts blinking the Associate LED
- Sends an API modem status frame ("associated") out the UART (API firmware only).

These behaviors are configurable using the following commands:

Command	Description
NJ	Sets the permit-join time on the router, or the time that it will allow new devices to join the network, measured in seconds.
LT	Sets the Associate LED blink time when joined. Default is 2 blinks per second (router).

Key Establishment

Key establishment is the process whereby a device can authenticate on a ZigBee network and obtain a new link key, known only to itself and the trust center.

The XBee SE module can automatically initiate key establishment with the trust center after joining a ZigBee network. (This behavior is disabled by default.)

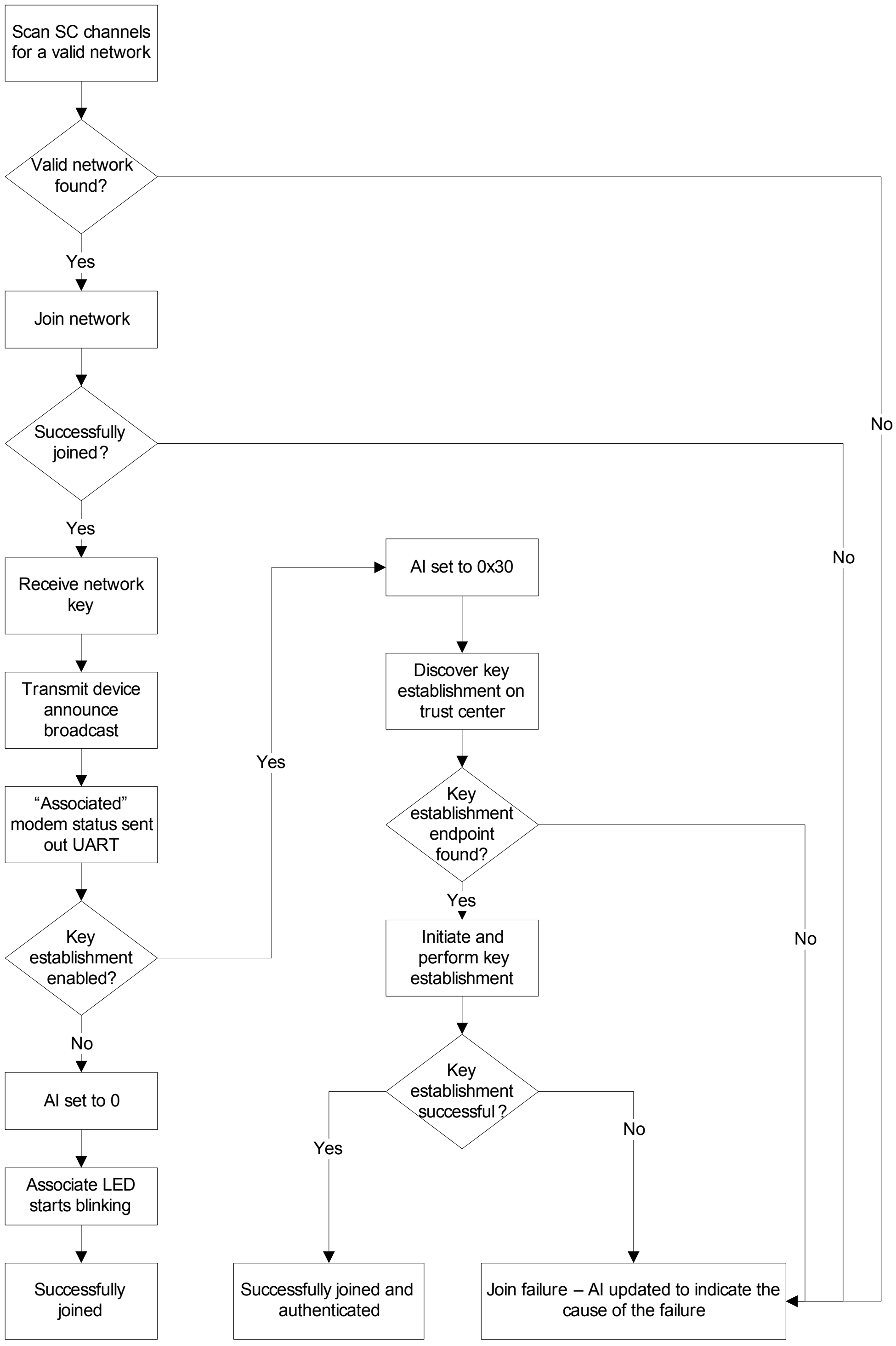
The following steps are necessary for a router to initiate key establishment:

- The router must have a Smart Energy certificate installed. (See appendix D.)
- The encryption options command (EO) must be set to enable key establishment.

If an XBee has certificate information installed, and if key establishment is enabled (EO command), the XBee will do the following after joining a ZigBee network:

- Send a ZDO match descriptor request to find the endpoint on the trust center that supports the key establishment cluster.
- Perform key establishment with the trust center to obtain a new link key. (This includes sending the initiate key establishment request, ephemeral data request, and confirm key commands.)

If the EO command is set to enable key establishment, the XBee will not blink its Associate LED or set AI to 0 until key establishment completes. The following image shows the join logic when key establishment is enabled or disabled.


Scan SC channels for a valid network
Valid network found?
Yes
No
Join network
Successfully joined?
Yes
No
Receive network key
Transmit device announce broadcast
"Associated" modem status sent out UART
Key establishment enabled?
Yes
No
AI set to 0
Associate LED starts blinking
Successfully joined
AI set to 0x30
Discover key establishment on trust center
Key establishment endpoint found?
Yes
No
Initiate and perform key establishment
Key establishment successful?
Yes
No
Successfully joined and authenticated
Join failure – AI updated to indicate the cause of the failure

Permit Joining

The permit joining attribute on the router is configurable with the NJ command.

Joining Temporarily Enabled

If NJ is 0x01 to 0xFE, joining will be enabled only for a number of seconds, based on the NJ parameter. If NJ=0x00, permit join is disabled. If NJ=0xFF, permit join is on permanently. If NJ is 0x00 or 0xFF and a CB2 command is issued either manually or by API frame, then a permit join of one minute will be broadcast to the network. The timer is started once the XBee joins a network. Joining will not be re-enabled if the module is power cycled or reset. The following mechanisms can restart the permit-joining timer:

- Changing NJ to a different value (and applying changes with the AC or CN commands)
- Pressing the commissioning button twice
- Issuing the CB command with a parameter of 2 (software emulation of a 2 button press)
- Causing the router to leave and rejoin the network.

Router Network Connectivity

Once a router joins a ZigBee network, it remains connected to the network on the same channel and PAN ID as long as it is not forced to leave. (See Leaving a Network section for details.) If the scan channels (SC), PAN ID (ID) and security settings (KY) do not change after a power cycle, the router will remain connected to the network after a power cycle.

If a router may physically move out of range of the network it initially joined, the application should include provisions to detect if the router can still communicate with the original network. If communication with the original network is lost, the application may choose to force the router to leave the network (see Leaving a Network section for details).

Leaving a Network

There are a couple of mechanisms that will cause the router to leave its current PAN and attempt to discover and join a new network based on its network joining parameter values.

These include the following:

- Change the ID command such that the current 64-bit PAN ID is invalid.
- Change the SC command such that the current channel (CH) is not included in the channel mask.
- Change the KY command value.
- Issue the NR0 command to cause the router to leave.
- Press the commissioning button 4 times or issue the CB command with a parameter of 4.
- Issue a network leave command.

Note that changes to ID, SC, and security command values only take effect when changes are applied (AC or CN commands).

Resetting the Router

When the router is reset or power cycled, it checks its PAN ID, operating channel and stack profile against the network configuration settings (ID, SC). If the router's PAN ID or operating channel is invalid, the router will leave the network and attempt to join a new network based on its network joining command values.

To prevent the router from leaving an existing network, the WR command should be issued after all network joining commands have been configured in order to retain these settings through power cycle or reset events.

Example: Joining a Network

After starting a coordinator (that is allowing joins), the following steps will cause a router to join the network:

1. Set ID to the desired 64-bit PAN ID, or to 0 to join any PAN.

2. Set SC to the list of channels to scan to find a valid network.

3. If SC or ID is changed from the default, apply changes (make SC and ID changes take effect) by issuing the AC or CN command.

4. The Associate LED will start blinking once the router has joined a PAN.

5. If the Associate LED is not blinking, the AI command can be read to determine the cause of join failure.

6. Once the router has joined, the OP and CH commands will indicate the operating 64-bit PAN ID and channel the router joined.

7. The MY command will reflect the 16-bit address the router received when it joined.

8. The API Modem Status frame ("Associated") is sent out the UART (API firmware only).

9. The joined router will allow other devices to join for a time based on its NJ setting.

Smart Energy Range Extender

Joining a Network

The following steps will allow you to join your Smart Energy Range Extender to an existing Smart Energy network:

1. Power the Smart Energy Range Extender by plugging it into a standard electric wall outlet. Check the Associate/Power LED is lit and steady.

2. Register the MAC address and link key (or install code) of the Smart Energy Range Extender with the Coordinator of the existing Smart Energy network you wish to join.

3. Permit joining on the network. Use the method preferred by the network Coordinator or joined routers of that network.

4. While joining is permitted on the network, start the Smart Energy Range Extender’s join process by pressing the commissioning button once. Joining will require about 20 seconds to complete. If joining is successful, the Associate/Power LED will blink.

The Smart Energy Range Extender will make three attempts to join a network before giving up. A steadily lit Associate/Power LED indicates the Smart Energy Range Extender has not joined a network.

Operation

To permit joining on the Smart Energy network with a joined Smart Energy Range Extender, press the commissioning button twice. A joined Range Extender may be removed from power, moved to another location and powered up again. The Associate/Power LED should continue to blink.

Note that a blinking Associate/Power LED is not an indicator of link quality, or that the Range Extender can currently communicate with the network. It is an indicator that the Range Extender was able to successfully join and associate with the network at some time in the past.

Leaving a Network

To make a joined Smart Energy Range Extender leave a network, press the commissioning button four times. Unlike Digi’s other Smart Energy or ZigBee devices, this action will not reset the device back to its default configuration settings.

The Smart Energy Range Extender will leave the network, attempt to re-join, then attempt to join a new network. If permit joining is enabled on a compatible neighboring Smart Energy network, and the configuration settings (ID, SC, etc) are compatible, and the Smart Energy Range Extender is in range of a joined router or the coordinator, the Smart Energy Range Extender should join with that network.

End Device Operation

Similar to routers, end devices must also discover and join a valid ZigBee network before they can participate in a network. After an end device has joined a network, it can communicate with other devices on the network. Since end devices are intended to be battery powered and therefore support low power (sleep) modes, end devices cannot allow other devices to join, nor can they route data packets.

Discovering ZigBee Networks

End devices go through the same process as routers to discover networks by issuing a PAN scan. After sending the broadcast beacon request transmission, the end device listens for a short time in order to receive beacons sent by nearby routers and coordinators on the same channel. The end device evaluates each beacon received on the channel to determine if a valid PAN is found. An end device considers a PAN to be valid if the PAN:

- Has a valid 64-bit PAN ID (PAN ID matches ID if ID > 0)
- Has the correct stack profile (ZS command)
- Is allowing joining
- Has capacity for additional end devices (see End Device Capacity section below).

If a valid PAN is not found, the end device performs the PAN scan on the next channel in its scan channels list and continues this process until a valid network is found, or until all channels have been scanned. If all channels have been scanned and a valid PAN was not discovered, the end device may enter a low power sleep state and scan again later.

If scanning all SC channels fails to discover a valid PAN, XBee Smart Energy modules will attempt to enter a low power state and will retry scanning all SC channels after the module wakes from sleeping. If the module cannot enter a low power state, it will retry scanning all channels, similar to the router. To meet Smart Energy requirements, the end device will attempt up to three scans. If all 3 attempts fail, the application or user must retry joining. The following events cause the XBee to perform up to three join attempts:

- Single commissioning button press (see chapter 7)
- CB command with a parameter of 1
- Resetting the XBee (FR or hardware reset).

Note: The XBee Smart Energy end device will not enter sleep until it has completed scanning all SC channels for a valid network.

Joining a Network

Once the end device discovers a valid network, it joins the network, similar to a router, by sending an association request (to the device that sent a valid beacon) to request a join on the ZigBee network. The device allowing the join then sends an association response frame that either allows or denies the join.

When an end device joins a network, it receives a 16-bit address from the device that allowed the join. The 16-bit address is randomly selected by the device that allowed the join.

Parent Child Relationship

Since an end device may enter low power sleep modes and not be immediately responsive, the end device relies on the device that allowed the join to receive and buffer incoming messages on its behalf until it is able to wake and receive those messages. The device that allowed an end device to join becomes the parent of the end device, and the end device becomes a child of the device that allowed the join.

End Device Capacity

Routers and coordinators maintain a table of all child devices that have joined called the child table. This table is a finite size and determines how many end devices can join. If a router or coordinator has at least one unused entry in its child table, the device is said to have end device capacity. In other words, it can allow one or more additional end devices to join. ZigBee networks should have sufficient routers to ensure adequate end device capacity.

In XBee Smart Energy firmware, the NC command (number of remaining end device children) can be used to determine how many additional end devices can join a router or coordinator. If NC returns 0, then the router or coordinator device has no more end device capacity. (Its child table is full.)

Also of note, since routers cannot sleep, there is no equivalent need for routers or coordinators to track joined routers. Therefore, there is no limit to the number of routers that can join a given router or coordinator device. (There is no "router capacity" metric.)

Authentication

In a network where security is enabled, the end device must then go through an authentication process. See chapter 5 for a discussion on security and authentication.

Device Registration

The trust center (coordinator) is responsible for deciding which devices can join the Smart Energy network. To prevent unwanted devices from joining the network, the coordinator sends the network key encrypted by the trust center link key. For a device to join a Smart Energy network and receive the network key, it must either:

- Have its 64-bit address and initial link key registered with the trust center, or
- Be pre-configured with the same trust center link key used by the trust center.

The Register Joining Device API frame (0x24) is used to provide the coordinator with the address and key information for each device that will join the network. See Chapter 9 for details.

Key Establishment

The trust center supports the key establishment cluster on endpoint 0x5E. If a device attempts to perform key establishment with the trust center, the trust center may spend up to 4 seconds performing computations. During this time, its Associate LED will cease blinking, and the coordinator will not be responsive to serial or RF traffic. CTS will de-assert during these periods, indicating when the application should avoid sending serial data.

Persistent Data

The end device can retain its PAN ID, operating channel, and security policy information through a power cycle. However, since end devices rely heavily on a parent, the end device does an orphan scan to try and contact its parent. If the end device does not receive an orphan scan response (called a coordinator realignment command), it will leave the network and try to discover and join a new network. When the end device leaves a network, the previous PAN ID and operating channel settings are lost.

Orphan Scans

When an end device comes up from a power cycle, it performs an orphan scan to verify it still has a valid parent. The orphan scan is sent as a broadcast transmission and contains the 64-bit address of the end device. Nearby routers and coordinator devices that receive the broadcast check their child tables for an entry that contains the end device's 64-bit address. If an entry is found with a matching 64-bit address, the device sends a coordinator realignment command to the end device that includes the end device's 16-bit address, 16-bit PAN ID, operating channel, and the parent's 64-bit and 16-bit addresses.

If the orphaned end device receives a coordinator realignment command, it is considered joined to the network. Otherwise, it will attempt to discover and join a valid network.

XBee: Smart Energy End Device Joining

When an end device is powered on, if it is not joined to a valid ZigBee network, or if the orphan scan fails to find a parent, it immediately attempts to find and join a valid ZigBee network.

Similar to a router, the following commands control the end device joining process.

Network joining commands used by an end device to join a network.

Command	Description
ID	Sets the 64-bit PAN ID to join. Setting ID=0 allows the router to join any 64-bit PAN ID.
SC	Set the scan channels bitmask that determines which channels an end device will scan to find a valid network. SC on the end device should be set to match SC on the coordinator and routers in the desired network. For example, setting SC to 0x281 enables scanning on channels 0x0B, 0x12, and 0x14, in that order.
SD	Set the scan duration, or time that the end device will listen for beacons on each channel.
KY	If the trust center link key is known, KY on the router can be set to match the trust center link key.

Once the end device joins a network, the network configuration settings can persist through power cycles as mentioned in the "Persistent Data" section previously. If joining fails, the status of the last join attempt can be read in the AI command register.

If any of these command values changes, when command register changes are applied, the end device will leave its current network and attempt to discover and join a new valid network.

When a Smart Energy end device has successfully joined a network, it

- Sets AI=0
- Starts blinking the Associate LED
- Sends an API modem status frame ("associated") out the UART (API firmware only)
- Attempts to enter low power modes.

These behaviors are configurable using the following commands:

Command	Description
LT	Sets the Associate LED blink time when joined. Default is 2 blinks per second (end devices).
SM, SP, ST, SN, SO	Parameters that configure the sleep mode characteristics. (See Managing End Devices chapter for details.)

Parent Connectivity

The XBee Smart Energy end device sends regular poll transmissions to its parent when it is awake. These poll transmissions query the parent for any new received data packets. The parent always sends a MAC layer acknowledgment back to the end device. The acknowledgment indicates whether the parent has data for the end device or not.

If the end device does not receive an acknowledgment for 3 consecutive poll requests, it considers itself disconnected from its parent and will attempt to discover and join a valid ZigBee network. See "Managing End Devices" chapter for details.

Resetting the End Device

When the end device is reset or power cycled, if the orphan scan successfully locates a parent, the end device then checks its PAN ID, operating channel against the network configuration settings (ID, SC). If the end device's PAN ID or operating channel is invalid, the end device will leave the network and attempt to join a new network based on its network joining command values.

To prevent the end device from leaving an existing network, the WR command should be issued after all network joining commands have been configured in order to retain these settings through power cycle or reset events.

Leaving a Network

There are a couple of mechanisms that will cause the end device to leave its current PAN and attempt to discover and join a new network based on its network joining parameter values. These include the following:

- The ID command changes such that the current 64-bit PAN ID is invalid.
- The SC command changes such that the current operating channel (CH) is not included in the channel mask.
- The NR0 command is issued to cause the end device to leave.
- The commissioning button is pressed 4 times or the CB command is issued with a parameter of 4.
- The end device's parent is powered down or the end device is moved out of range of the parent such that the end device fails to receive poll acknowledgment messages.

Note that changes to command values only take effect when changes are applied (AC or CN commands).

Example: Joining a Network

After starting a coordinator (that is allowing joins), the following steps will cause an XBee end device to join the network:

1. Set ID to the desired 64-bit PAN ID, or to 0 to join any PAN.

2. Set SC to the list of channels to scan to find a valid network.

3. If SC or ID is changed from the default, apply changes (make SC and ID changes take effect) by issuing the AC or CN command.

4. The Associate LED will start blinking once the end device has joined a PAN.

5. If the Associate LED is not blinking, the AI command can be read to determine the cause of join failure.

6. Once the end device has joined, the OP and CH commands will indicate the operating 64-bit PAN ID and channel the end device joined.

7. The MY command will reflect the 16-bit address the end device received when it joined.

8. The API Modem Status frame ("Associated") is sent out the UART (API firmware only).

9. The joined end device will attempt to enter low power sleep modes based on its sleep configuration commands (SM, SP, SN, ST, SO).

Channel Scanning

As mentioned previously, routers and end devices must scan one or more channels to discover a valid network to join. When a join attempt begins, the XBee sends a beacon request transmission on the lowest channel specified in the SC (scan channels) command bitmask. If a valid PAN is found on the channel, the XBee will attempt to join the PAN on that channel. Otherwise, if a valid PAN is not found on the channel, it will attempt scanning on the next higher channel in the SC command bitmask. The XBee will continue to scan each channel (from lowest to highest) in the SC bitmask until a valid PAN is found or all channels have been scanned. Once all channels have been scanned, the next join attempt will start scanning on the lowest channel specified in the SC command bitmask.

For example, if the SC command is set to 0x400F, the XBee would start scanning on channel 11 (0x0B) and scan until a valid beacon is found, or until channels 11, 12, 13, 14, and 25 have been scanned (in that order).

Once an XBee router or end device joins a network on a given channel, if the XBee is told to leave (see "Leaving a Network" section), it will leave the channel it joined on and continue scanning on the next higher channel in the SC bitmask.

For example, if the SC command is set to 0x400F, and the XBee joins a PAN on channel 12 (0x0C), if the XBee leaves the channel, it will start scanning on channel 13, followed by channels 14 and 25 if a valid network is not found. Once all channels have been scanned, the next join attempt will start scanning on the lowest channel specified in the SC command bitmask.

ZigBee and Smart Energy: Creating a Network

Network Formation

The ESP, acting as the Coordinator in a ZigBee network, selects a channel and PAN ID for the network. Configuring an ESP for network creation involves the following AT commands: ID, SC, SD, NK, and KY.

Joining the Network

Details of what follows appears in the ZigBee Smart Energy Profile Specification ([1], section 5.4, Annex C, and Annex F). A more proprietary description may be found in the ZB RF Module manual. Joining a network involves the following AT commands: ID, SC, SD, KY, and NJ.

Configuration

There are two ways to prepare for a new node to join a Smart Energy network. Both involve the use of a link key which acts as a recognizable signature for authenticating identity. Both methods are referred to as "Out-of-band" link key configuration. "Out-of-band" means the link key is not transmitted across a radio band, or otherwise ever displayed publicly. It is important that the link key be kept private and secure. Otherwise, the security of the network could be compromised.

Preferred Method

Register the 64-bit extended address (MAC address) and the 16 byte link key (or installation code) of the joining device with the Trust Center on the ESP. This is done with the ZigBee Register Joining Device (0x24) API frame which is sent through the UART port to the Smart Energy Generic Module. The response will be the ZigBee Register Joining Device Status (0xA4) API frame which will indicate success or failure.

This is the preferred method, because it limits access to the network to pre-approved MAC addresses.

Second Method

Configure the joining device with the Trust Center Link Key. This is done by using the KY command on the joining device to match the Trust Center Link Key. This is risky, as it discloses an address and link key which are unique to a particular Smart Energy network. Anyone else who gets access to that address and link key will be able to join that network, which would compromise the security of that particular network.

Enable Joining

Before the new device can join the network, the network must be commanded to temporarily drop its guard, and permit joining for a time.

There are three methods by which one may enable "permit joining" on the network. All three may only be done by a node which is already joined to the network. Typically this will be done by the utility--sending a command to the ESP through its non-ZigBee backhaul interface.

Commissioning button

Two presses of the commissioning button (pin 20) on any node which is already joined to the network will cause a broadcast of the Permit-Join message. The NJ register setting determines the permit join time interval in units of one second. A value of 0x00 or 0xFF indicates the time interval should be one minute.

CB2

An AT command "CB" with a parameter value of 2 is equivalent in effect to two presses of the Commissioning button. The AT Command (0x08) API frame can be used to do this. The permit join time is set similar to the commissioning button example.

Broadcast of a ZDO Permit-join

The Explicit Addressing ZigBee Command Frame (0x11) can be used to broadcast a ZDO Permit-Join message ([4], 2.4.3.3.7). To use the frame, set the field values as follows:

64 bit Destination Address = 0x0000 0000 0000 FFFF

16 bit Destination Address = 0xFFFE

Source Endpoint = 0

Destination Endpoint = 0

Cluster = 0x36

Profile Id = 0x0000

Data Payload =

- 1 byte sequence number,
- 1 byte time (seconds),
- 1 byte Trust Center significance
  - 0x00 = no effect on Trust Center
  - 0x01 = Trust Center authentication policy is affected, if addressed to the T.C.

Discovery

After a device has joined and been authenticated, typically service discovery follows. The following example describes the hierarchy and order of discovery among nodes, endpoints, and clusters.

- Node
  - Endpoint(s)
    - Profile Id
    - SE Device Id
    - Cluster(s)
      - Attributes(s)
      - Command(s)

A node has an extended 64-bit MAC address and a short 16-bit NWK address. A node may support one or more endpoints.

Each endpoint is described by a Profile Id (0x0109 for Smart Energy), a SE Device ID, and a set of clusters.

Each cluster may be a server or a client-side type of that cluster. Generally the server side holds the attributes and responds to commands issued by a client side cluster. The client side issues commands to get and set the attributes.

To discover a device or service in the network, a Match_Desc_req ([4],2.4.3.1.7) is broadcast to find a node (or nodes) which supports a desired Profile ID (0x0109 for Smart Energy) with a set of input and output cluster ids. The response contains the short NWK address of the node, and a list of the endpoints which match the descriptor.

The extended MAC address of the node may be obtained by sending an IEEE address request, using the short NWK address of the target node.

An Explicit Addressing ZigBee Command Frame (0x11), addressed by (node/endpoint/cluster), carrying a General Command Frame ([3], 2.4) as payload, is used to discover, read, and write the attributes of a cluster.

Discovery Examples

The following information provides abbreviated examples of discovery with reference links for obtaining information about nodes, endpoints, clusters, and attributes. The examples are given in hierarchical order.

Node Discovery Example

One can obtain information about a node by unicasting a Node_Desc_req command. It is addressed to cluster 0x0002, and carries a 16-bit NWK address as payload.

In response one would receive a Node_Desc_rsp response. It will be addressed as cluster 0x8002, and carry as payload a: Status (1), 16-bit NWK address (2), and a node descriptor.

The node descriptor describes the node type, whether complex or user descriptors are available, the frequency band it uses, capabilities of its MAC layer, its manufacturer's code, a server mask, and descriptor capabilities. The node type tells if it is a Coordinator, Router, or End Device type. The server mask tells if the node is hosting a Trust Center. The descriptor capabilities tell if it can serve up an active endpoint list and/or an extended simple descriptor list.

Active Endpoint Example

One can obtain a list of endpoints supported on a node by unicasting an Active_EP_req command . It is addressed to cluster 0x0005, and carries a 16-bit NWK address as payload.

In response one would receive an Active_EP_rsp response ([4], 2.4.4.1.6.1). It will be addressed as cluster 0x8005, and carry as payload a: Status(1), 16-bit NWK address(2), an active endpoint count(1), and an active endpoint list.

Knowing the active endpoint list of a node, one can then make simple descriptor requests on each endpoint (see next section).

Simple Descriptor Example

One can obtain the simple descriptor for an endpoint on a node by uncasting a Simple_Desc_req command. It is addressed to cluster 0x0004, and carries a 16-bit NWK address and endpoint value as payload.

In response one would receive a Simple_Desc_rsp response. It is addressed as cluster 0x8004, and carries as payload: Status (1), 16-bit NWK address (2), Length (1) of the simple descriptor which follows, and a Simple Descriptor.

The simple descriptor will tell you the endpoint value, Application profile Id, Application device Id, input cluster list, and output cluster list which are associated with the endpoint.

As a reminder, the Smart Energy Application Profile Id is 0x0109. The mapping of Smart Energy Device IDs to Device Types is listed in a subsequent section in this document. The input and output cluster lists will tell you whether this endpoint is acting as the server or client side of a particular cluster ID. Input clusters are associated with the server side, and output clusters with the client side.

Match Descriptor Example

Rather than interrogate each node in the network to determine its functions and capabilities, one can use the Match_Desc_req to find a node (or nodes) which host endpoints which support a particular profile id and cluster combination.

The Match_Desc_req command may be broadcast to the network as a whole, or directed by unicast to a specific node.

It is addressed to cluster 0x0006, and carries as payload: 16-bit NWK address(2), ProfileId(2), #InputClusters(1), Input Cluster list(2 bytes per clusterId), #OutputClusters(1), and Output Cluster List (2 bytes per clusterId).

In response one receives a Match_Desc_rsp response from each matching device on the node (if the request was unicast) or nodes (if the request was broadcast). A match occurs if the profile id and at least one of the input or output cluster IDs given in the request can be found. It is addressed as cluster 0x8006, and carries as payload: Status(1), 16-bit NWK address(2), matchLength(1), and matchList.

The matchList is a list of endpoints which match the request.

Attribute Discovery Example

Once you have a desired node/endpoint/cluster address combination, you will want to manage the attributes. For example, to discover the attributes of the basic cluster on a node, one would use an Explicit Addressing ZigBee Command Frame as follows:

Address the node with its 64-bit address, use 0xFFFE for the 16-bit address, set the source endpoint to the return address to which you would like the response delivered, use a destination endpoint of 0x00 (ZDO endpoint), destination cluster Id of 0x00 (Basic cluster), ZDP profile id of 0x0000, 0x00 for broadcast radius (or whatever maximum hop count you'd like), and 0x20 for enabling APS end-to-end security (or 0x00 if not - some clusters require the higher security). The second byte of the ZCL payload sets an upper limit on how many attributes you would like returned at a time. The first byte is the starting index of attributes in which you are interested.

4. Data, Addressing and Routing

Addressing

All ZigBee devices have two different addresses, a 64-bit and a 16-bit address. The characteristics of each are described below.

64-bit Device Addresses

The 64-bit address is a unique device address assigned during manufacturing. This address is unique to each physical device. The 64-bit address includes a 3-byte Organizationally Unique Identifier (OUI) assigned by the IEEE. The 64-bit address is also called the extended address.

16-bit Device Addresses

A device receives a 16-bit address when it joins a ZigBee network. For this reason, the 16-bit address is also called the "network address". The 16-bit address of 0x0000 is reserved for the coordinator. All other devices receive a randomly generated address from the router or coordinator device that allows the join. The 16-bit address can change under certain conditions:

- An address conflict is detected where two devices are found to have the same 16-bit address
- A device leaves the network and later joins (it can receive a different address)

All ZigBee transmissions are sent using the source and destination 16-bit addresses. The routing tables on ZigBee devices also use 16-bit addresses to determine how to route data packets through the network. However, since the 16-bit address is not static, it is not a reliable way to identify a device.

To solve this problem, the 64-bit destination address is often included in data transmissions to guarantee data is delivered to the correct destination. The ZigBee stack can discover the 16-bit address, if unknown, before transmitting data to a remote.

Application Layer Addressing

ZigBee devices can support multiple application profiles, cluster IDs, and endpoints. (See "ZigBee Application Layers - In Depth" in chapter 3.) Application layer addressing allows data transmissions to be addressed to specific profile IDs, cluster IDs, and endpoints. Application layer addressing is useful if an application must

- Interoperate with other ZigBee devices
- Utilize service and network management capabilities of the ZDO
- Operate on a public application profile such as Smart Energy.

The API firmware provides a simple yet powerful interface that can easily send data to any profile ID, endpoint, and cluster ID combination on any device in a ZigBee network.

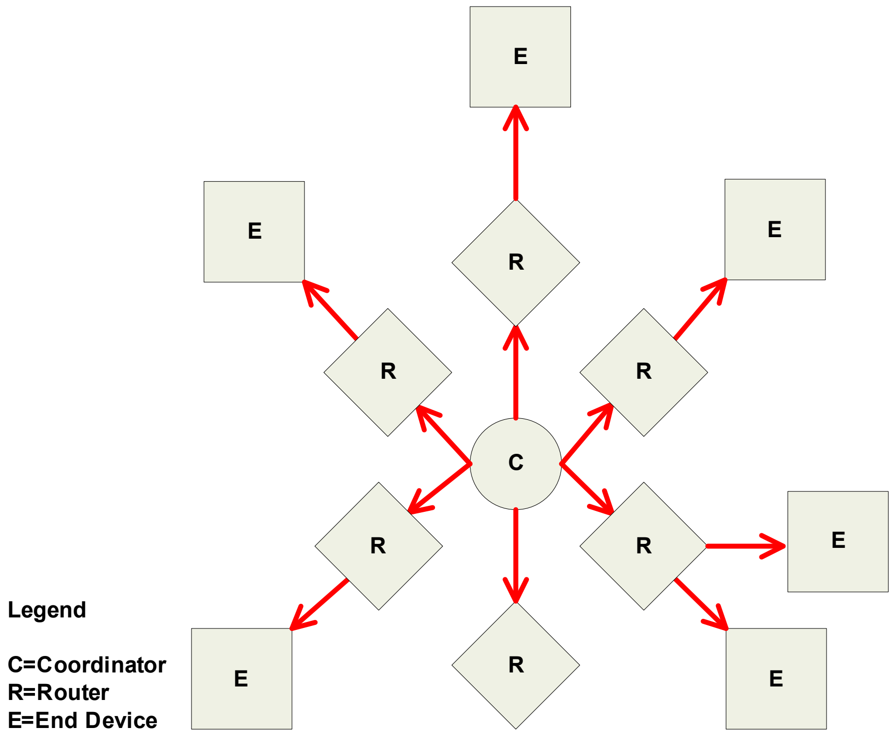
Data Transmission

ZigBee data packets can be sent as either unicast or broadcast transmissions. Unicast transmissions route data from one source device to one destination device, whereas broadcast transmissions are sent to many or all devices in the network.

Broadcast Transmissions

Broadcast transmissions within the ZigBee protocol are intended to be propagated throughout the entire network such that all nodes receive the transmission. To accomplish this, all devices that receive a broadcast transmission will retransmit the packet 3 times.

Broadcast Data Transmission



Each node that transmits the broadcast will also create an entry in a local broadcast transmission table. This entry is used to keep track of each received broadcast packet to ensure the packets are not endlessly transmitted. Each entry persists for 8 seconds. The broadcast transmission table holds 8 entries.

For each broadcast transmission, the ZigBee stack must reserve buffer space for a copy of the data packet. This copy is used to retransmit the packet as needed.Large broadcast packets will require more buffer space.

Since broadcast transmissions are retransmitted by each device in the network, broadcast messages should be used sparingly.

Unicast Transmissions

Unicast transmissions are sent from one source device to another destination device. The destination device could be an immediate neighbor of the source, or it could be several hops away. Unicast transmissions that are sent along a multiple hop path require some means of establishing a route to the destination device. See the "RF Packet Routing" section in chapter 4 for details.

Address Resolution

As mentioned previously, each device in a ZigBee network has both a 16-bit (network) address and a 64-bit (extended) address. The 64-bit address is unique and assigned to the device during manufacturing, and the 16-bit address is obtained after joining a network. The 16-bit address can also change under certain conditions.

When sending a unicast transmission, the ZigBee network layer uses the 16-bit address of the destination and each hop to route the data packet. If the 16-bit address of the destination is not known, the ZigBee stack includes a discovery provision to automatically discover the destination device's 16-bit address before routing the data.

To discover a 16-bit address of a remote, the device initiating the discovery sends a broadcast address discovery transmission. The address discovery broadcast includes the 64-bit address of the remote device whose 16-bit address is being requested. All nodes that receive this transmission check the 64-bit address in the payload and compare it to their own 64-bit address.

If the addresses match, the device sends a response packet back to the initiator. This response includes the remote's 16-bit address. When the discovery response is received, the initiator will then transmit the data.

Address Table

Each ZigBee device maintains an address table that maps a 64-bit address to a 16-bit address. When a transmission is addressed to a 64-bit address, the ZigBee stack searches the address table for an entry with a matching 64-bit address, in hopes of determining the destination's 16-bit address. If a known 16-bit address is not found, the ZigBee stack will perform address discovery to discover the device's current 16-bit address.

Sample Address Table

64-bit Address	16-bit Address
0013 A200 4000 0001	0x4414
0013 A200 400A 3568	0x1234
0013 A200 4004 1122	0xC200
0013 A200 4002 1123	0xFFFE (unknown)

The XBee modules can store up to 10 address table entries. For applications where a single device (i.e. coordinator) may send unicast transmissions to more than 10 devices, the application should implement an address table to store the 16-bit and 64-bit addresses for each remote device. Any XBee that will send data to more than 10 remotes should also use API firmware. The application can then send both the 16-bit and 64-bit addresses to the XBee in the API transmit frames which will significantly reduce the number of 16-bit address discoveries and greatly improve data throughput.

If an application will support an address table, the size should ideally be larger than the maximum number of destination addresses the device will communicate with. Each entry in the address table should contain a 64-bit destination address and its last known 16-bit address.

When sending a transmission to a destination 64-bit address, the application should search the address table for a matching 64-bit address. If a match is found, the 16-bit address should be populated into the 16-bit address field of the API frame. If a match is not found, the 16-bit address should be set to 0xFFFE (unknown) in the API transmit frame.

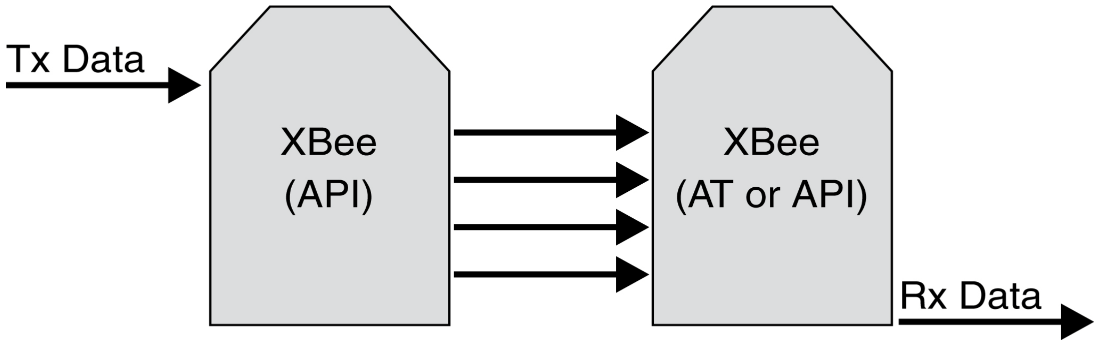
The API provides indication of a remote device's 16-bit address in the following frames:

- All receive data frames
  ZigBee Explicit Rx Indicator (0x91)
  ZigBee Route Record Indicator (0xA1)
- All transmit data frames
  Explicit Addressing ZigBee Command Frame (0x11)
  ZigBee Tx Status (0x8B)

The application should always update the 16-bit address in the address table when one of these frames is received to ensure the table has the most recently known 16-bit address. If a transmission failure occurs, the application should set the 16-bit address in the table to 0xFFFE (unknown).

Fragmentation

Each unicast transmission may support up to 84 bytes of RF payload. (Enabling security or using source routing can reduce this number. See the NP command for details.) However, the XBee SE firmware supports a new ZigBee feature called fragmentation that allows a single large data packet to be broken up into multiple RF transmissions and reassembled by the receiver before sending data out its UART. This is shown in the image below.



The API transmit frame can include up to 128 bytes of data, which will be broken up into multiple transmissions and reassembled on the receiving side. If one or more of the fragmented messages are not received by the receiving device, the receiver will drop the entire message, and the sender will indicate a transmission failure in the Tx Status API frame.

Applications that do not wish to use fragmentation should avoid sending more than the maximum number of bytes in a single RF transmission. See the "Maximum RF Payload Size" section for details.

Data Transmission Examples

Example 1: Send a transmission to the coordinator

Use the explicit transmit request frame (0x11) to send data to the coordinator. The 64-bit address can either be set to 0x0000000000000000, or to the 64-bit address of the coordinator. The 16-bit address should be set to 0xFFFE when using the 64-bit address of all 0x00s.

Suppose an ASCII "1" will be sent to the coordinator, addressed to destination endpoint 0xE8, cluster ID 0x0011, and profile ID 0xC105. The explicit transmit API frame for this transmission might look like the following:

7E 00 15 11 01 0000 0000 0000 0000 FFFE E8 E8 0011 C105 00 00 31 18

Notice the 16-bit address is set to 0xFFFE. This is required when sending to a 64-bit address of 0x00s.

Example 2: Send a broadcast transmission

This example will use the explicit transmit request frame (0x11) to send an ASCII "1" in a broadcast transmission.

To send an ASCII "1" as a broadcast transmission, the following API frame can be used:

7E 0015 11 01 000000000000FFFF FFFE 5E 5E 1234 0109 00 00 31 B5

Notice the destination 16 bit address is set to 0xFFFE for broadcast transmission.

RF Packet Routing

Unicast transmissions may require some type of routing. ZigBee includes several different ways to route data, each with its own advantages and disadvantages. These are summarized in the table below.

Routing Approach	Description	When to Use
Ad hoc On-demand Distance Vector (AODV) Mesh Routing	Routing paths are created between source and destination, possibly traversing multiple nodes ("hops"). Each device knows who to send data to next to eventually reach the destination	Use in networks that will not scale beyond about 40 destination devices.
Many-to-One Routing	A single broadcast transmission configures reverse routes on all devices into the device that sends the broadcast	Useful when many remote devices must send data to a single gateway or collector device.
Source Routing	Data packets include the entire route the packet should traverse to get from source to destination	Improves routing efficiency in large networks (over 40 remote devices)

Note – End devices do not make use of these routing protocols. Rather, an end device sends a unicast transmission to its parent and allows the parent to route the data packet in its behalf.

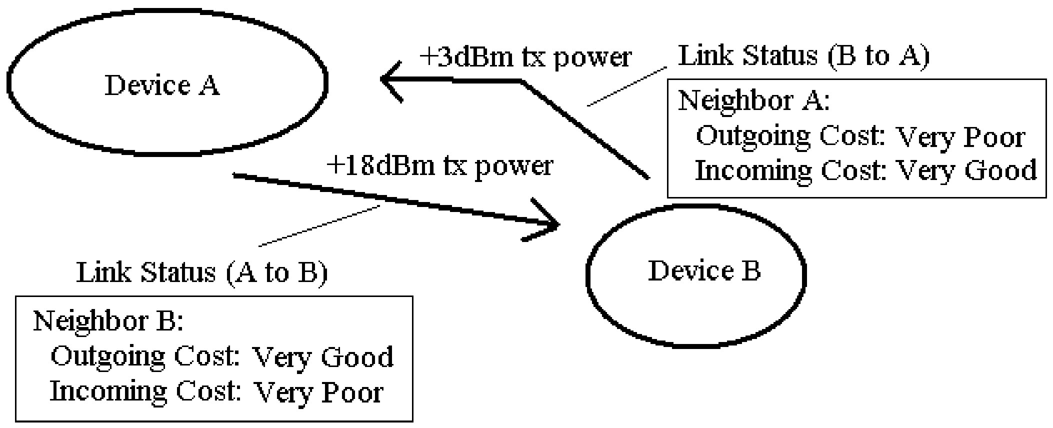
Note - A network cannot revert from Many-to-One routing to AODV routing without first doing a network reset (NR).

Link Status Transmission

Before discussing the various routing protocols, it is worth understanding the primary mechanism in ZigBee for establishing reliable bi-directional links. This mechanism is especially useful in networks that may have a mixture of devices with varying output power and/or receiver sensitivity levels.

Each coordinator or router device periodically sends a link status message. This message is sent as a 1-hop broadcast transmission, received only by one-hop neighbors. The link status message contains a list of neighboring devices and incoming and outgoing link qualities for each neighbor. Using these messages, neighboring devices can determine the quality of a bi-directional link with each neighbor and use that information to select a route that works well in both directions.

For example, consider a network of two neighboring devices that send periodic link status messages. Suppose that the output power of device A is +18dBm, and the output power of device B is +3dBm (considerably less than the output power of device A). The link status messages might indicate the following:

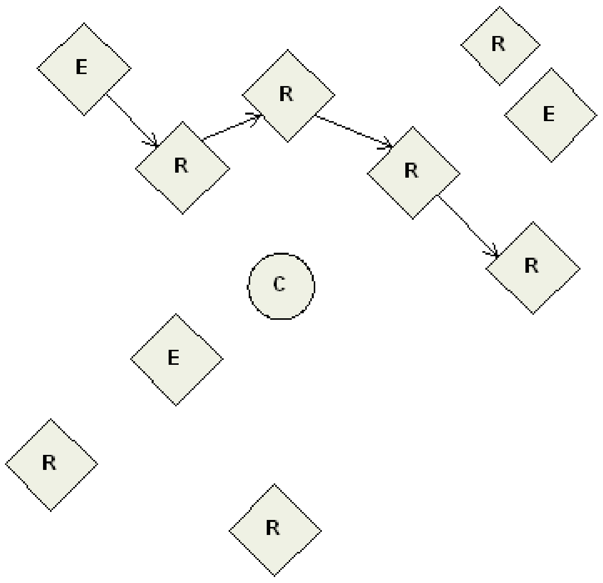


This mechanism enables devices A and B to recognize that the link is not reliable in both directions and select a different neighbor when establishing routes. (Such links are called asymmetric links, meaning the link quality is not similar in both directions.)

AODV Mesh Routing

ZigBee employs mesh routing to establish a route between the source device and the destination. Mesh routing allows data packets to traverse multiple nodes (hops) in a network to route data from a source to a destination. Routers and coordinators can participate in establishing routes between source and destination devices using a process called route discovery. The Route discovery process is based on the AODV (Ad-hoc On-demand Distance Vector routing) protocol.

Sample Transmission Through a Mesh Network



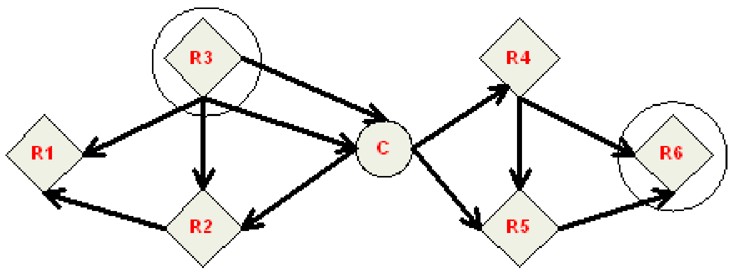
AODV (Ad-hoc On-demand Distance Vector) Routing Algorithm

Routing under the AODV protocol is accomplished using tables in each node that store the next hop (intermediary node between source and destination nodes) for a destination node. If a next hop is not known, route discovery must take place in order to find a path. Since only a limited number of routes can be stored on a Router, route discovery will take place more often on a large network with communication between many different nodes.

Node	Destination Address	Next Hop Address
R3	Router 6	Coordinator
C	Router 6	Router 5
R5	Router 6	Router 6

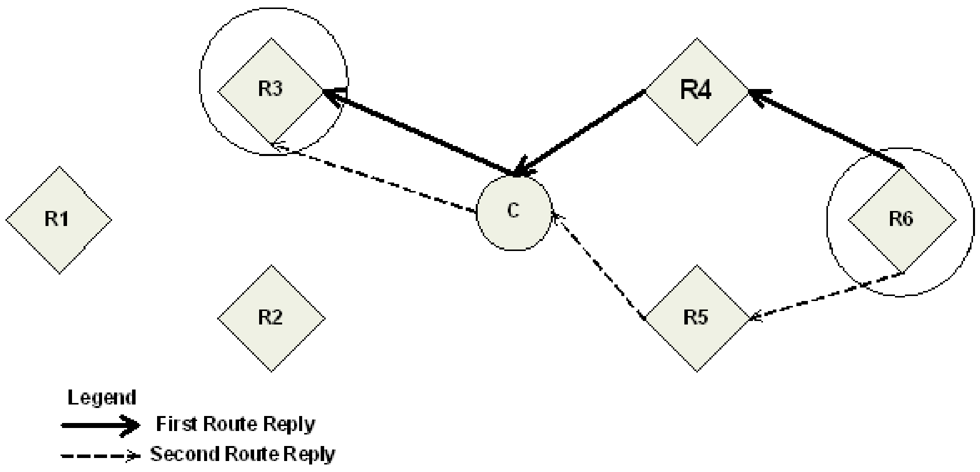
When a source node must discover a route to a destination node, it sends a broadcast route request command. The route request command contains the source network address, the destination network address and a path cost field (a metric for measuring route quality). As the route request command is propagated through the network (refer to the Broadcast Transmission), each node that re-broadcasts the message updates the path cost field and creates a temporary entry in its route discovery table.

Sample Route Request (Broadcast) Transmission Where R3 is Trying to Discover a Route to R6



When the destination node receives a route request, it compares the 'path cost' field against previously received route request commands. If the path cost stored in the route request is better than any previously received, the destination node will transmit a route reply packet to the node that originated the route request. Intermediate nodes receive and forward the route reply packet to the source node (the node that originated route request).

Sample Route Reply (Unicast) Where R6 Sends a Route Reply to R3.



Note: R6 could send multiple replies if it identifies a better route.

Retries and Acknowledgments

ZigBee includes acknowledgment packets at both the MAC and Application Support (APS) layers. When data is transmitted to a remote device, it may traverse multiple hops to reach the destination. As data is transmitted from one node to its neighbor, an acknowledgment packet (MAC ACK) is transmitted in the opposite direction to indicate that the transmission was successfully received. If the ACK is not received, the transmitting device will retransmit the data, up to 4 times. This ACK is called the MAC layer acknowledgment.

In addition, the device that originated the transmission expects to receive an acknowledgment packet (Nwk ACK) from the destination device. This ACK will traverse the same path that the data traversed, but in the opposite direction. If the originator fails to receive this ACK, it will retransmit the data, up to 2 times until an ACK is received. This ACK is called the ZigBee APS layer acknowledgment.

Refer to the ZigBee specification for more details.

Many-to-One Routing

In networks where many devices must send data to a central collector or gateway device, AODV mesh routing requires significant overhead. If every device in the network had to discovery a route before it could send data to the data collector, the network could easily become inundated with broadcast route discovery messages.

Many-to-one routing is an optimization for these kinds of networks. Rather than require each device to do its own route discovery, a single many-to-one broadcast transmission is sent from the data collector to establish reverse routes on all devices. This is shown in the figure below. The left side shows the many broadcasts the devices can send when they create their own routes and the route replies generated by the data collector The right side shows the benefits of many-to-one routing where a single broadcast creates reverse routes to the data collector on all routers.

The many-to-one broadcast is a route request message with the target discovery address set to the address of the data collector. Devices that receive this route request create a reverse many-to-one routing table entry to create a path back to the data collector. The ZigBee stack on a device uses historical link quality information about each neighbor to select a reliable neighbor for the reverse route.

When a device sends data to a data collector, and it finds a many-to-one route in its routing table, it will transmit the data without performing a route discovery. The many-to-one route request should be sent periodically to update and refresh the reverse routes in the network.

Applications that require multiple data collectors can also use many-to-one routing. If more than one data collector device sends a many-to-one broadcast, devices will create one reverse routing table entry for each collector.

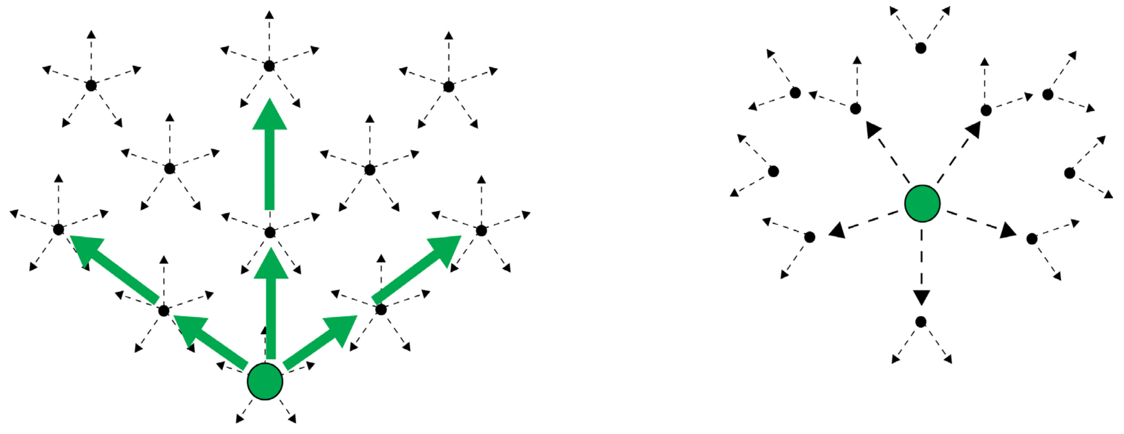
In SE firmware, the AR command is used to enable many-to-one broadcasting on a device. The AR command sets a time interval (measured in 10 second units) for sending the many to one broadcast transmission. (See the command table for details.)

Source Routing

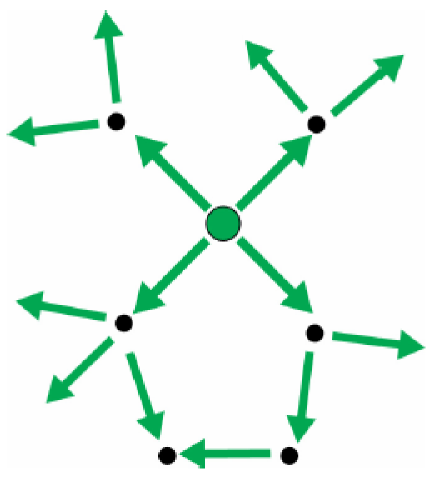
In applications where a device must transmit data to many remotes, AODV routing would require performing one route discovery for each destination device to establish a route. If there are more destination devices than there are routing table entries, established AODV routes would be overwritten with new routes, causing route discoveries to occur more regularly. This could result in larger packet delays and poor network performance.

ZigBee source routing helps solve these problems. In contrast to many-to-one routing that establishes routing paths from many devices to one data collector, source routing allows the collector to store and specify routes for many remotes.

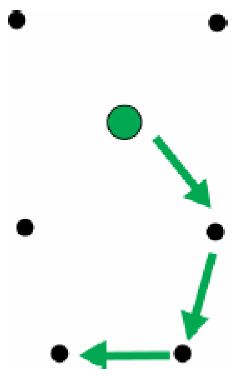
To use source routing, a device must use the API firmware, and it must send periodic many-to-one route request broadcasts (AR command) to create a many-to-one route to it on all devices. When remote devices send RF data using a many-to-one route, they first send a route record transmission. The route record transmission is unicast along the many-to-one route until it reaches the data collector. As the route record traverses the many-to-one route, it appends the 16-bit address of each device in the route into the RF payload. When the route record reaches the data collector, it contains the address of the sender, and the 16-bit address of each hop in the route. The data collector can store the routing information and retrieve it later to send a source routed packet to the remote. This is shown in the images below.



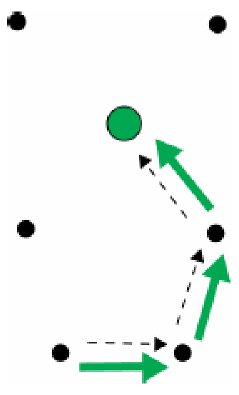
- - - ▶	Route Request Broadcast
➡	Route Reply Unicast
●	Data Collector
•	Router



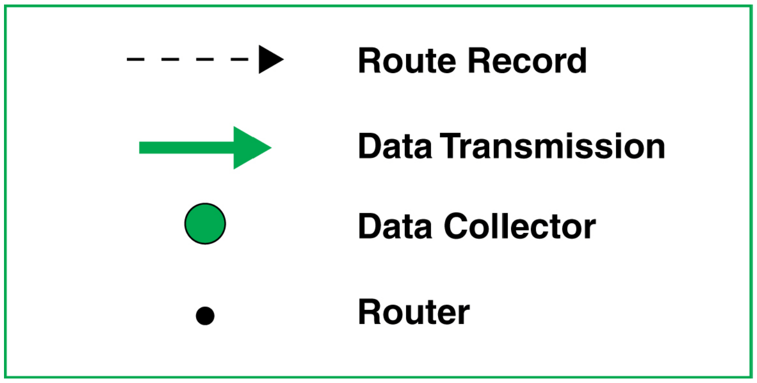
The data collector sends a many-to-one route request broadcast to create reverse routes on all devices.



A remote device sends an RF data packet to the data collector. (This is prefaced by a route record transmission to the data collector.)



After obtaining a source route, the data collector sends a source routed transmission to the remote device.



Acquiring Source Routes

Acquiring source routes requires the remote device to send a unicast transmission to a data collector (device that sends many-to-one route request broadcasts). There are several ways to force remotes to send route record transmissions.

1. If the application on remote devices periodically sends data to the data collector, each transmission will force a route record to occur.

2. Periodic IO sampling can be enabled on remotes to force them to send data at a regular rate. Each IO sample would be prefaced by a route record. (See chapter 8 for details.)

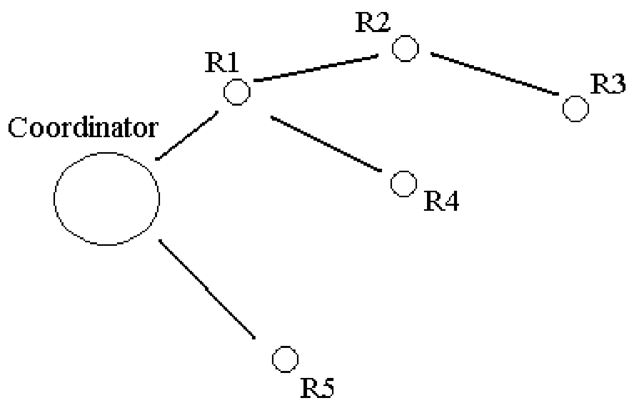
Storing Source Routes

When a data collector receives a route record, it sends it out the UART as a Route Record Indicator API frame (0xA1). To use source routing, the application should receive these frames and store the source route information.

Sending a Source Routed Transmission

To send a source routed transmission, the application should send a Create Source Route API frame (0x21) to the XBee to create a source route in its internal source route table. After sending the Create Source Route API frame, the application can send data transmission or remote command request frames as needed to the same destination, or any destination in the source route. Once data must be sent to a new destination (a destination not included in the last source route), the application should first send a new Create Source Route API frame. The XBee can buffer one source route that includes up to 10 hops (excluding source and destination).

For example, suppose a network exists with a coordinator and 5 routers (R1, R2, R3, R4, R5) with known source routes as shown below.



To send a source-routed packet to R3, the application must send a Create Source Route API frame (0x21) to the XBee, with a destination of R3, and 2 hops (R1 and R2). If the 64- bit address of R3 is 0x0013A200 404a1234 and the 16-bit addresses of R1, R2, and R3 are:

Device	16-bit address
R1	0xAABB
R2	0xCCDD
R3	0xEEFF

Then the Create Source Route API frame would be:

```
7E  0012  21 00  0013A200 404A1234  EEFF  00 02  CCDD  AABB 5C
```

Where:

0x0012 - length

0x21 - API ID (create source route)

0x00 - frame ID (set to 0 always)

0x0013A200 404A1234 - 64-bit address of R3 (destination)

0xEEFF - 16-bit address of R3 (destination)

0x00 - Route options (set to 0)

0x02 - Number of intermediate devices in the source route

0xCCDD - Address of furthest device (1-hop from target)

0xAABB - Address of next-closer device

0x5C - Checksum (0xFF - SUM (all bytes after length))

Repairing Source Routes

It is possible in a network to have an existing source route fail (i.e. a device in the route moves or goes down, etc.). If a device goes down in a source routed network, all routes that used the device will be broken.

As mentioned previously, source routing must be used with many-to-one routing. (A device that uses source routing must also send a periodic many-to-one broadcast in order to keep routes fresh). If a source route is broken, remote devices must send in new route record transmissions to the data collector to provide it with a new source route. This requires that remote devices send data transmissions into the data collector. See the earlier "Acquiring Source Routes" section for details.

Retries and Acknowledgments

ZigBee includes acknowledgment packets at both the MAC and Application Support (APS) layers. When data is transmitted to remote device, it may traverse multiple hops to reach the destination. As data is transmitted from one node to its neighbor, an acknowledgment packet (MAC ACK) is transmitted in the opposite direction to indicate that the transmission was successfully received. If the ACK is not received, the transmitting device will retransmit the data, up to 4 times. This ACK is called the MAC layer acknowledgment.

In addition, the device that originated the transmission expects to receive an acknowledgment packet (Nwk ACK) from the destination device. This ACK will traverse the same path that the data traversed, but in the opposite direction. If the originator fails to receive this ACK, it will retransmit the data, up to 2 times until an ACK is received. This ACK is called the ZigBee APS layer acknowledgment.

Refer to the ZigBee specification for more details.

Encrypted Transmissions

Encrypted transmissions are routed similar to non-encrypted transmissions with one exception. As an encrypted packet propagates from one device to another, each device decrypts the packet using the network key, and authenticates the packet by verifying packet integrity. It then re-encrypts the packet with its own source address and frame counter values, and sends the message to the next hop. This process adds some overhead latency to unicast transmissions, but it helps prevent replay attacks. See chapter 5 for details.

Maximum RF Payload Size

XBee SE firmware includes a command (NP) that returns the maximum number of RF payload bytes that can be sent in a unicast transmission. Querying the NP command, like most other commands, returns a HEXADECIMAL value. This number will change based on whether security is enabled or not. If security is enabled (EE command), the maximum number of RF payload bytes decreases since security requires additional overhead.

After reading the NP value, the following conditions can affect the maximum number of data bytes in a single RF transmission:

- Broadcast transmissions can support 8 bytes more than unicast transmissions.
- If source routing is used, the 16-bit addresses in the source route are inserted into the RF payload space. For example, if NP returns 84 bytes, and a source route must traverse 3 intermediate hops (3 16-bit addresses), the total number of bytes that can be sent in one RF packet is 78.
- Enabling APS encryption (API tx option bit set) will reduce the number of payload bytes by 9.

Throughput

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

Data Throughput*

Configuration	Data Throughput
1 hop, RR, SD	35kbps
1 hop, RR, SE	19kbps
1 hop, RE, SD	25kbps
1 hop, RE, SE	16kbps
1 hop, ER, SD	21kbps
1 hop, ER, SE	16kbps
4 hops, RR, SD	10kbps
4 hops, RR, SE	5kbps

RR = router to router,

RE = router to end device (non-sleeping),

ER = end device (non-sleeping) to router,

SE = security enabled.

4 hops = 5 nodes total, 3 intermediate router nodes

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

ZDO Transmissions

ZigBee defines a ZigBee Device Objects layer (ZDO) that can provide device and service discovery and network management capabilities. This layer is described below.

ZigBee Device Objects (ZDO)

The ZigBee Device Objects (ZDO) is supported to some extent on all ZigBee devices. The ZDO is an endpoint that implements services described in the ZigBee Device Profile in the ZigBee specification. Each service has an assigned cluster ID, and most service requests have an associated response. The following table describes some common ZDO services.

Cluster Name	Cluster ID	Description
Network Address Request	0x0000	Request a 16-bit address of the radio with a matching 64-bit address (required parameter).
Active Endpoints Request	0x0005	Request a list of endpoints from a remote device.
LQI Request	0x0031	Request data from a neighbor table of a remote device.
Routing Table Request	0x0032	Request to retrieve routing table entries from a remote device.
Network Address Response	0x8000	Response that includes the 16-bit address of a device.

Cluster Name	Cluster ID	Description
LQI Response	0x8031	Response that includes neighbor table data from a remote device.
Routing Table Response	0x8032	Response that includes routing table entry data from a remote device.

Refer to the ZigBee specification for a detailed description of all ZigBee Device Profile services.

Sending a ZDO Command

To send a ZDO command, an explicit transmit API frame must be used and formatted correctly. The source and destination endpoints must be set to 0, and the profile ID must be set to 0. The cluster ID must be set to match the cluster ID of the appropriate service. For example, to send an active endpoints request, the cluster ID must be set to 0x0005.

The first byte of payload in the API frame is an application sequence number (transaction sequence number) that can be set to any single byte value. This same value will be used in the first byte of the ZDO response. All remaining payload bytes must be set as required by the ZDO. All multi-byte values must be sent in little endian byte order.

Receiving ZDO Commands and Responses

In XBee SE firmware, ZDO commands can easily be sent using the API. In order to receive incoming ZDO commands, receiver application addressing must be enabled with the AO command. (See examples later in this section.) Not all incoming ZDO commands are passed up to the application.

When a ZDO message is received on endpoint 0 and profile ID 0, the cluster ID indicates the type of ZDO message that was received. The first byte of payload is generally a sequence number that corresponds to a sequence number of a request. The remaining bytes are set as defined by the ZDO. Similar to a ZDO request, all multi-byte values in the response are in little endian byte order.

Example 1: Send a ZDO LQI Request to read the neighbor table contents of a remote.

Looking at the ZigBee specification, the cluster ID for an LQI Request is 0x0031, and the payload only requires a single byte (start index). This example will send an LQI request to a remote device with a 64-bit address of 0x0013A200 40401234. The start index will be set to 0, and the transaction sequence number will be set to 0x76

API Frame:

7E 0016 11 01 0013A200 40401234 FFFE 00 00 0031 0000 00 00 76 00 CE

0x0016 - length

0x11 - Explicit transmit request

0x01 - frame ID (set to a non-zero value to enable the transmit status message, or set to 0 to disable)

0x0013A200 40401234 - 64-bit address of the remote

0xFFFE - 16-bit address of the remote (0xFFFE = unknown). Optionally, set to the 16-bit address of the destination if known.

0x00 - Source endpoint

0x00 - Destination endpoint

0x0031 - Cluster ID (LQI Request, or Neighbor table request)

0x0000 - Profile ID (ZigBee Device Profile)

0x00 - Broadcast radius

0x00 - Tx Options

0x76 - Transaction sequence number

0x00 - Required payload for LQI request command

0xCE - Checksum (0xFF - SUM (all bytes after length))

Description:

This API frame sends a ZDO LQI request (neighbor table request) to a remote device to obtain data from its neighbor table. Recall that the AO command must be set correctly on an API device to enable the explicit API receive frames in order to receive the ZDO response.

Example 2: Send a ZDO Network Address Request to discover the 16-bit address of a remote.

Looking at the ZigBee specification, the cluster ID for a network Address Request is 0x0000, and the payload only requires the following:

[64-bit address] + [Request Type] + [Start Index]

This example will send a Network Address Request as a broadcast transmission to discover the 16-bit address of the device with a 64-bit address of 0x0013A200 40401234. The request type and start index will be set to 0, and the transaction sequence number will be set to 0x44

API Frame:

7E 001F 11 01 00000000 0000FFFF FFFE 00 00 0000 0000 00 00 44 34124040 00A21300 00 00 33

0x001F - length

0x11 - Explicit transmit request

0x01 - frame ID (set to a non-zero value to enable the transmit status message, or set to 0 to disable)

0x00000000 0000FFFF - 64-bit address for a broadcast transmission

0xFFFE - Set to this value for a broadcast transmission.

0x00 - Source endpoint

0x00 - Destination endpoint

0x0000 - Cluster ID (Network Address Request)

0x0000 - Profile ID (ZigBee Device Profile)

0x00 - Broadcast radius

0x00 - Tx Options

0x44 - Transaction sequence number

0x34124040 00A21300 00 00 - Required payload for Network Address Request command

0x33 - Checksum (0xFF - SUM (all bytes after length))

Description:

This API frame sends a broadcast ZDO Network Address Request to obtain the 16-bit address of a device with a 64-bit address of 0x0013A200 40401234. Note the bytes for the 64-bit address were inserted in little endian byte order. All multi-byte fields in the API payload of a ZDO command must have their data inserted in little endian byte order. Also recall that the AO command must be set correctly on an API device to enable the explicit API receive frames in order to receive the ZDO response.

Transmission Timeouts

The ZigBee stack includes two kinds of transmission timeouts, depending on the nature of the destination device. For destination devices such as routers whose receiver is always on, a unicast timeout is used. The unicast timeout estimates a timeout based on the number of unicast hops the packet should traverse to get data to the destination device. For transmissions destined for end devices, the ZigBee stack uses an extended timeout that includes the unicast timeout (to route data to the end device's parent), and it includes a timeout for the end device to finish sleeping, wake, and poll the parent for data.

The ZigBee stack includes some provisions for a device to detect if the destination is an end device or not. The ZigBee stack uses the unicast timeout unless it knows the destination is an end device.

The XBee API includes a transmit options bit that can be set to specify if the extended timeout should be used for a given transmission. If this bit is set, the extended timeout will be used when sending RF data to the specified destination. To improve routing reliability, applications should set the extended timeout bit when sending data to end devices if:

- The application sends data to 10 or more remote devices, some of which are end devices, AND
- The end devices may sleep longer than the unicast timeout

Equations for these timeouts are computed in the following sections.

Note: The timeouts in this section are worst-case timeouts and should be padded by a few hundred milliseconds. These worst-case timeouts apply when an existing route breaks down (e.g. intermediate hop or destination device moved).

Unicast Timeout

The unicast timeout is settable with the NH command. The actual unicast timeout is computed as ((50 * NH) + 100). The default NH value is 30 which equates to a 1.6 second timeout.

The unicast timeout includes 3 transmission attempts (1 attempt and 2 retries). The maximum total timeout is about:

3 * ((50 * NH) + 100).

For example, if NH=30 (0x1E), the unicast timeout is about

3 * ((50 * 30) + 100), or

3 * (1500 + 100), or

3 * (1600), or

4800 ms, or

4.8 seconds.

Extended Timeout

The worst-case transmission timeout when sending data to an end device is somewhat larger than when transmitting to a router or coordinator. As described later in chapter 6, RF data packets are actually sent to the parent of the end device, who buffers the packet until the end device wakes to receive it. The parent will buffer an RF data packet for up to (1.2 * SP) time.

To ensure the end device has adequate time to wake and receive the data, the extended transmission timeout to an end device is:

(50 * NH) + (1.2 * SP)

This timeout includes the packet buffering timeout (1.2 * SP) and time to account for routing through the mesh network (50 * NH).

If an acknowledgment is not received within this time, the sender will resend the transmission up to two more times. With retries included, the longest transmission timeout when sending data to an end device is:

3 * ((50 * NH) + (1.2 * SP))

The SP value in both equations must be entered in millisecond units. (The SP command setting uses 10ms units and must be converted to milliseconds to be used in this equation.)

For example, suppose a router is configured with NH=30 (0x1E) and SP=0x3E8 (10,000 ms), and that it is either trying to send data to one of its end device children, or to a remote end device. The total extended timeout to the end device is about:

3 * ((50 * NH) + (1.2 * SP)), or

3 * (1500 + 12000), or

3 * (13500), or

40500 ms, or

40.5 seconds.

Transmission Examples

Example 1: Send a unicast API data transmission to the coordinator using 64-bit address 0, with payload "TxData".

API Frame:

7E 001A 11 01 0000000000000000 FFFE 5E 5E 1234 0109 00 00 547844617461 9E

Field Composition:

001A - length

0x11 - API ID (Explicit Addressing ZigBee Command Frame)

0x01 - frame ID (set greater than 0 to enable the tx-status response)

0x00000000 00000000 - 64-bit address of coordinator (SE definition)

0xFFFE - Required 16-bit address if sending data to 64-bit address of 0.

0x5E - source endpoint address

0x5E - destination endpoint address

0x1234 - cluster ID address

0x0109 - Profile ID address (Smart Energy)

0x00 - Broadcast radius (0 = max hops)

0x00 - Tx options

0x54 78 44 61 74 61 - ASCII representation of "TxData" string

0x9E - Checksum (0xFF - SUM (all bytes after length))

Description:

This transmission sends the string "TxData" to the coordinator, without knowing the coordinator device's 64-bit address. A 64-bit address of 0 is defined as the coordinator in SE firmware. If the coordinator's 64-bit address was known, the 64-bit address of 0 could be replaced with the coordinator's 64-bit address, and the 16-bit address could be set to 0x0000 or 0xFFFE.

Example 2 - Send a broadcast API data transmission that all devices can receive (including sleeping end devices), with payload "TxData".

API Frame:

7E 001A 11 01 00000000 0000FFFF FFFE 5E 5E 1234 0109 00 00 54 78 44 61 74 61 A0

Field Composition:

0x001A - length

0x11 - API ID (tx data)

0x01 - frame ID (set to a non-zero value to enable the tx-status response)

0x00000000 0000FFFF - Broadcast definition (including sleeping end devices)

0xFFFE - Required 16-bit address to send broadcast transmission.

0x5E - source endpoint address

0x5E - destination endpoint address

0x1234 - cluster ID address

0x0109 - Profile ID address (Smart Energy)

0x00 - Broadcast radius (0 = max hops)

0x00 - Tx options

0x54 78 44 61 74 61 - ASCII representation of "TxData" string

0xAD - Checksum (0xFF - SUM (all bytes after length))

Description:

This transmission sends the string "TxData" as a broadcast transmission. Since the destinationaddress is set to 0xFFFF, all devices, including sleeping end devices can receive this broadcast.

Because receiver application addressing is enabled by default, the XBee will report all received data frames in the explicit format (0x91) to indicate the source and destination endpoints, cluster ID, and profile ID that each packet was received on. (Status messages like modem status and route record indicators are not affected.)

5. Security

ZigBee supports various levels of security that can be configured depending on the needs of the application. Security provisions include:

- 128-bit AES encryption
- Two security keys that can be preconfigured or obtained during joining
- Support for a trust center
- Provisions to ensure message integrity, confidentiality, and authentication.

The first half of this chapter describes various security features defined in the ZigBee-PRO specification, while the last half illustrates how the XBee and XBee-PRO modules can be configured to support these features.

Security Modes

The ZigBee standard supports three security modes – residential, standard, and high security. Residential security was first supported in the ZigBee 2006 standard. This level of security requires a network key be shared among devices. Standard security adds a number of optional security enhancements over residential security, including an APS layer link key. High security adds entity authentication, and a number of other features not widely supported.

XBee SE modules support high security mode when certificate keys are installed and authentication is enabled.

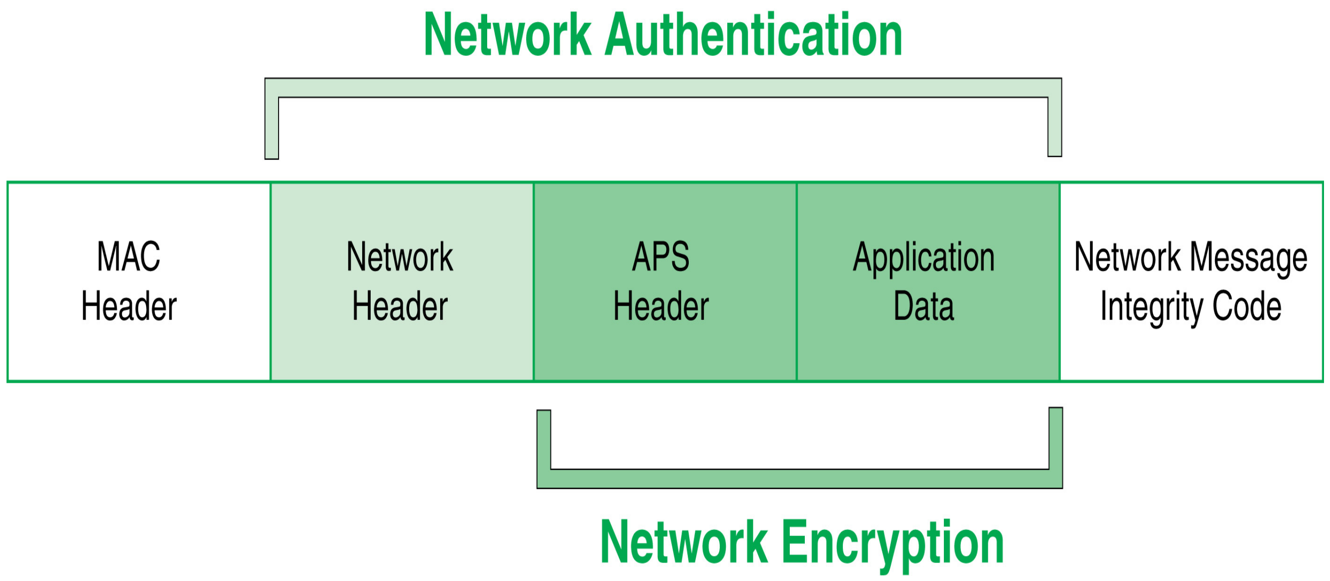
ZigBee Security Model

ZigBee security is applied to the Network and APS layers. Packets are encrypted with 128-bit AES encryption. A network key and optional link key can be used to encrypt data. Only devices with the same keys are able to communicate together in a network. Routers and end devices that will communicate on a secure network must obtain the correct security keys.

Network Layer Security

The network key is used to encrypt the APS layer and application data. In addition to encrypting application messages, network security is also applied to route request and reply messages, APS commands, and ZDO commands. Network encryption is not applied to MAC layer transmissions such as beacon transmissions, etc. If security is enabled in a network, all data packets will be encrypted with the network key.

Packets are encrypted and authenticated using 128-bit AES. This is shown in the figure below.



Frame Counter

The network header of encrypted packets includes a 32-bit frame counter. Each device in the network maintains a 32-bit frame counter that is incremented for every transmission. In addition, devices track the last known 32-bit frame counter for each of its neighbors. If a device receives a packet from a neighbor with a smaller frame counter than it has previously seen, the packet is discarded. The frame counter is used to protect against replay attacks.

If the frame counter reaches a maximum value of 0xFFFFFFFF, it does not wrap to 0 and no more transmissions can be sent. Due to the size of the frame counters, reaching the maximum value is a very unlikely event for most applications. The following table shows the required time under different conditions, for the frame counter to reach its maximum value.

Average Transmission Rate	Time until 32-bit frame counter expires
1 / second	136 years
10 / second	13.6 years

To clear the frame counters without compromising security, the network key can be changed in the network. When the network key is updated, the frame counters on all devices reset to 0. (See the Network Key Updates section for details.)

Message Integrity Code

The network header, APS header, and application data are all authenticated with 128-bit AES. A hash is performed on these fields and is appended as a 4-byte message integrity code (MIC) to the end of the packet. The MIC allows receiving devices to ensure the message has not been changed. The MIC provides message integrity in the ZigBee security model. If a device receives a packet and the MIC does not match the device's own hash of the data, the packet is dropped.

Network Layer Encryption and Decryption

Packets with network layer encryption are encrypted and decrypted by each hop in a route. When a device receives a packet with network encryption, it decrypts the packet and authenticates the packet. If the device is not the destination, it then encrypts and authenticates the packet, using its own frame counter and source address in the network header section.

Since network encryption is performed at each hop, packet latency is slightly longer in an encrypted network than in a non-encrypted network. Also, security requires 18 bytes of overhead to include a 32-bit frame counter, an 8-byte source address, 4-byte MIC, and 2 other bytes. This reduces the number of payload bytes that can be sent in a data packet.

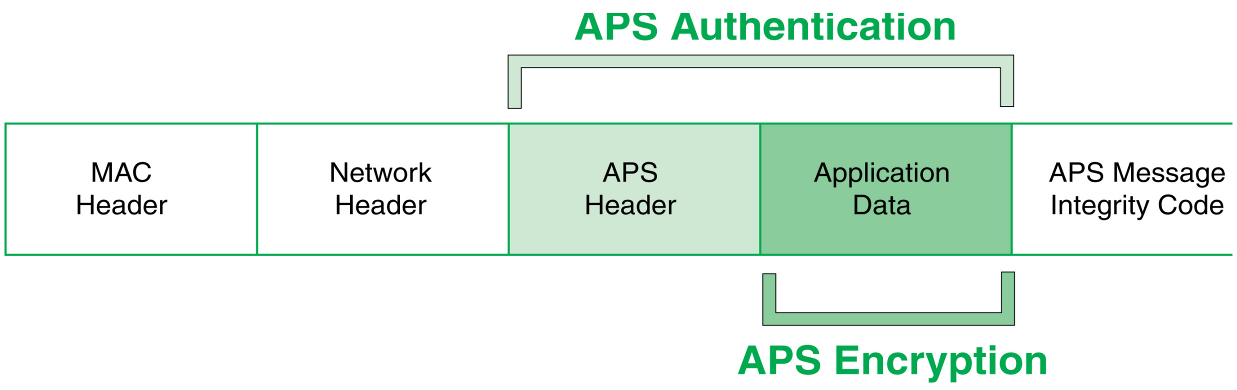
Network Key Updates

ZigBee supports a mechanism for changing the network key in a network. When the network key is changed, the frame counters in all devices reset to 0.

APS Layer Security

APS layer security can be used to encrypt application data using a key that is shared between source and destination devices. Where network layer security is applied to all data transmissions and is decrypted and re-encrypted on a hop-by-hop basis, APS security is optional and provides end-to-end security using an APS link key that only the source and destination device know. APS security can be applied on a packet-by-packet basis. APS security cannot be applied to broadcast transmissions.

If APS security is enabled, packets are encrypted and authenticated using 128-bit AES. This is shown in the figure below:



Message integrity Code

If APS security is enabled, the APS header and data payload are authenticated with 128-bit AES. A hash is performed on these fields and appended as a 4-byte message integrity code (MIC) to the end of the packet. This MIC is different than the MIC appended by the network layer. The MIC allows the destination device to ensure the message has not been changed. If the destination device receives a packet and the MIC does not match the destination device's own hash of the data, the packet is dropped.

APS Link Keys

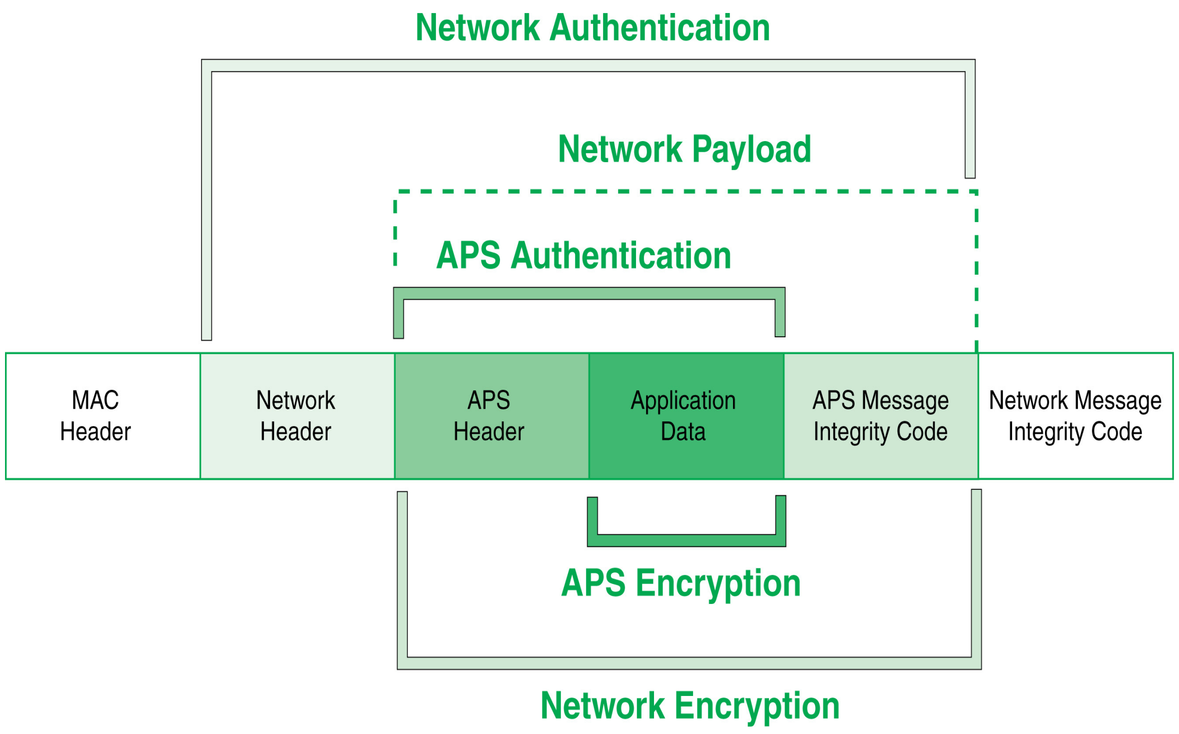
There are two kinds of APS link keys – trust center link keys and application link keys. A trust center link key is established between a device and the trust center, where an application link key is established between a device and another device in the network where neither device is the trust center.

APS Layer Encryption and Decryption

Packets with APS layer encryption are encrypted at the source and only decrypted by the destination. Since APS encryption appends a 4 byte MIC and other fields, the maximum data payload is reduced by 9 bytes when APS encryption is used.

Network and APS Layer Encryption

Network and APS layer encryption can both be applied to data. The following figure demonstrates the authentication and encryption performed on the final ZigBee packet when both are applied.



Trust Center

ZigBee defines a trust center device that is responsible for authenticating devices that join the network. The trust center also manages link key distribution in the network.

Forming and Joining a Secure Network

The coordinator is responsible for selecting a network encryption key. This key can either be preconfigured or randomly selected. In addition, the coordinator generally operates as a trust center and must therefore select the trust center link key. The trust center link key can also be preconfigured or randomly selected.

Devices that join the network must obtain the network key when they join. When a device joins a secure network, the network and link keys can be sent to the joining device. If the joining device has a pre-configured trust center link key, the network key will be sent to the joining device encrypted by the link key. Otherwise, if the joining device is not pre-configured with the link key, the device could only join the network if the network key is sent unencrypted ("in the clear"). The trust center must decide whether or not to send the network key unencrypted to joining devices that are not pre-configured with the link key. Sending the network key unencrypted is not recommended as it can open a security hole in the network. To maximize security, devices should be pre-configured with the correct link key.

Implementing Security on the XBee

If security is enabled in the XBee SE firmware, devices acquire the network key when they join a network. Data transmissions are always encrypted with the network key, and can optionally be end-to-end encrypted with the APS link key. The following sections discuss the security settings and options in the XBee SE firmware.

Enabling Security

To enable security on a device, the EE command must be set to 1. If the EE command value is changed and changes are applied (e.g. AC command), the XBee module will leave the network (PAN ID and channel) it was operating on, and attempt to form or join a new network.

If EE is set to 1, all data transmissions will be encrypted with the network key. When security is enabled, the maximum number of bytes in a single RF transmission will be reduced. See the NP command for details.

Note: The EE command must be set the same on all devices in a network. Changes to the EE command should be written to non-volatile memory (to be preserved through power cycle or reset events) using the WR command.

Setting the Network Security Key

The coordinator must select the network security key for the network. The NK command (write-only) is used to set the network key. If NK=0 (default), a random network key will be selected. (This should suffice for most applications.) Otherwise, if NK is set to a non-zero value, the network security key will use the value specified by NK. NK is only supported on the coordinator.

Routers and end devices with security enabled (EE=1) acquire the network key when they join a network. They will receive the network key encrypted with the link key if they share a pre-configured link key with the coordinator. See the following section for details.

Setting the APS Trust Center Link Key

The coordinator must also select the trust center link key, using the KY command. If KY=0 (default), the coordinator will select a random trust center link key (not recommended). Otherwise, if KY is set greater than 0, this value will be used as the pre-configured trust center link key. KY is write-only and cannot be read.

Note: Application link keys (sent between two devices where neither device is the coordinator) are not supported in SE firmware at this time.

Random Trust Center Link Keys

If the coordinator selects a random trust center link key (KY=0, default), then it will allow devices to join the network without having a pre-configured link key. However, this will cause the network key to be sent unencrypted over-the-air to joining devices and is not recommended.

Pre-configured Trust Center Link Keys

If the coordinator uses a pre-configured link key (KY > 0), then the coordinator will not send the network key unencrypted to joining devices. Only devices with the correct pre-configured link key will be able to join and communicate on the network.

Enabling APS Encryption

APS encryption is an optional layer of security that uses the link key to encrypt the data payload. Unlike network encryption that is decrypted and encrypted on a hop-by-hop basis, APS encryption is only decrypted by the destination device. The XBee must be configured with security enabled (EE set to 1) to use APS encryption.

APS encryption can be enabled in API firmware on a per-packet basis. To enable APS encryption for a given transmission, the "enable APS encryption" transmit options bit (0x20) should be set in the API transmit frame. Enabling APS encryption decreases the maximum payload size by 9 bytes.

Using a Trust Center

The EO command can be used to define the coordinator as a trust center. If the coordinator is a trust center, it will be alerted to all new join attempts in the network. The trust center also has the ability to update or change the network key on the network.

Updating the Network Key with a Trust Center

If the trust center has started a network and the NK value is changed, the coordinator will update the network key on all devices in the network. (Changes to NK will not force the device to leave the network.) The network will continue to operate on the same channel and PAN ID, but the devices in the network will update their network key, increment their network key sequence number, and restore their frame counters to 0.

Authentication

By default, routers and end devices run with authentication enabled. To disable authentication, set EO to 0 on a router or end device.

Note: Authentication presumes a valid certificate has been installed on the device.

Coordinators always run with authentication enabled (EO2) and their EO setting is read-only.

XBee Security Examples

This section covers some sample XBee configurations to support different security modes. Several AT commands are listed with suggested parameter values. The notation in this section includes an '=' sign to indicate what each command register should be set to - for example, EE=1. This is not the correct notation for setting command values in the XBee. In the API, the two byte command is used in the command field, and parameters are populated as binary values in the parameter field.

Example 1: Forming a network with security (pre-configured link keys)

1. Start a coordinator with the following settings:
   a. ID=2234 (arbitrarily selected)
   b. EE=1
   c. NK=0
   d. KY=4455
   e. WR (save networking parameters to preserve them through power cycle)
2. Configure one or more routers or end devices with the following settings:
   a. ID=2234
   b. EE=1
   c. KY=4455
   d. WR (save networking parameters to preserve them through power cycle)
3. Read the AI setting on the coordinator and joining devices until they return 0 (formed or joined a network).

In this example, EE, ID, and KY are set the same on all devices. After successfully joining the secure network, all application data transmissions will be encrypted by the network key. Since NK was set to 0 on the coordinator, a random network key was selected. And since the link key (KY) was configured the same on all devices, to a non-zero value, the network key was sent encrypted by the pre-configured link key (KY) when the devices joined.

Example 2: Forming a network with security (obtaining keys during joining)

1. Start a coordinator with the following settings:
   a. ID=2235
   b. EE=1
   c. NK=0
   d. KY=0
   e. WR (save networking parameters to preserve them through power cycle)
2. Configure one or more routers or end devices with the following settings:
   a. ID=2235
   b. EE=1
   c. KY=0
   d. WR (save networking parameters to preserve them through power cycle)
3. Read the AI setting on the coordinator and joining devices until they return 0 (formed or joined a network).

In this example, EE, ID, and KY are set the same on all devices. Since NK was set to 0 on the coordinator, a random network key was selected. And since KY was set to 0 on all devices, the network key was sent unencrypted ("in the clear") when the devices joined. This approach introduces a security vulnerability into the network and is not recommended.

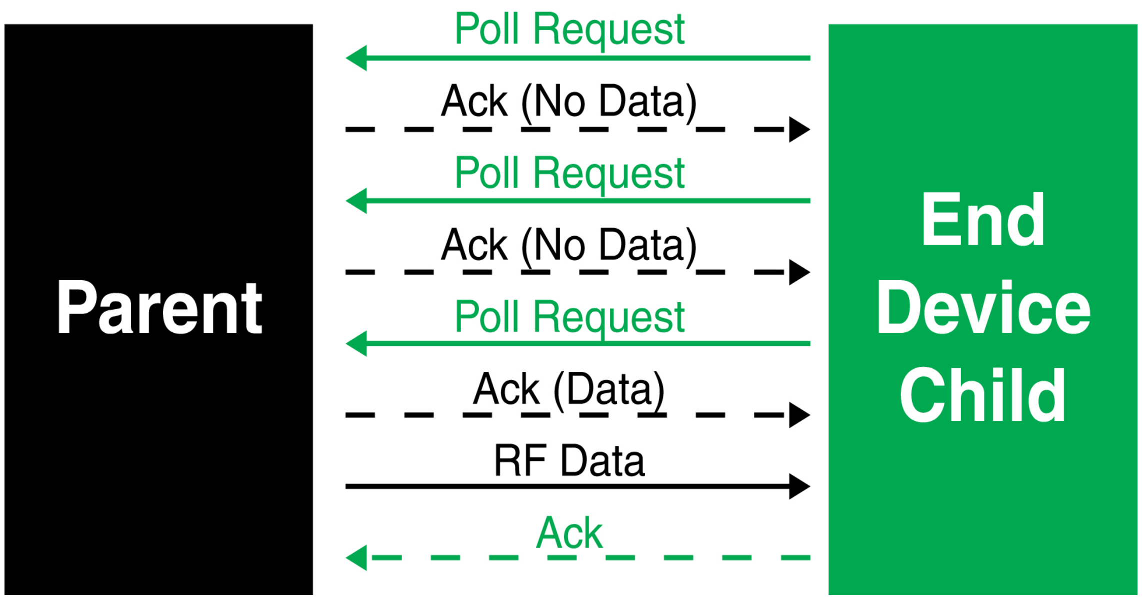
6. Managing End Devices

ZigBee end devices are intended to be battery-powered devices capable of sleeping for extended periods of time. Since end devices may not be awake to receive RF data at a given time, routers and coordinators are equipped with additional capabilities (including packet buffering and extended transmission timeouts) to ensure reliable data delivery to end devices.

End Device Operation

When an end device joins a ZigBee network, it must find a router or coordinator device that is allowing end devices to join. Once the end device joins a network, a parent-child relationship is formed between the end device and the router or coordinator that allowed it to join. See chapter 3 for details.

When the end device is awake, it sends poll request messages to its parent. When the parent receives a poll request, it checks a packet queue to see if it has any buffered messages for the end device. It then sends a MAC layer acknowledgment back to the end device that indicates if it has data to send to the end device or not.



If the end device receives the acknowledgment and finds that the parent has no data for it, the end device can return to idle mode or sleep. Otherwise, it will remain awake to receive the data. This polling mechanism allows the end device to enter idle mode and turn its receiver off when RF data is not expected in order to reduce current consumption and conserve battery life.

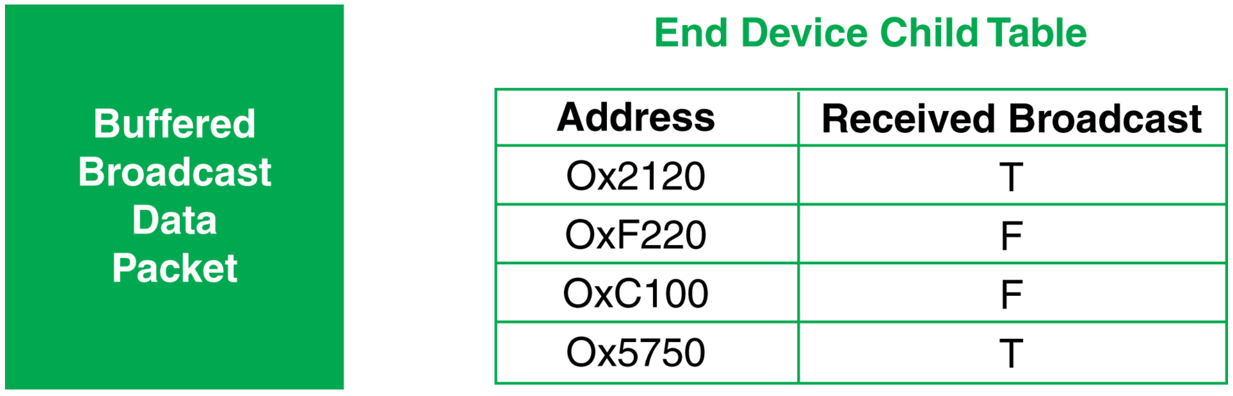
The end device can only send data directly to its parent. If an end device must send a broadcast or a unicast transmission to other devices in the network, it sends the message directly to its parent and the parent performs any necessary route or address discoveries to route the packet to the final destination.

Parent Operation

Each router or coordinator maintains a child table that contains the addresses of its end device children. A router or coordinator that has unused entries in its child table is said to have end device capacity, or the ability to allow new end devices to join. If the child table is completely filled (such that the number of its end device children matches the number of child table entries), the device cannot allow any more end devices to join to it.

Since the end device children are not guaranteed to be awake at a given time, the parent is responsible for managing incoming data packets in behalf of its end device children. If a parent receives an RF data transmission destined for one of its end device children, and if the parent has enough unused buffer space, it will buffer the packet. The data packet will remain buffered until a timeout expires, or until the end device sends a poll request to retrieve the data.

The parent can buffer one broadcast transmission for all of its end device children. When a broadcast transmission is received and buffered, the parent sets a flag in its child table when each child polls and retrieves the packet. Once all children have received the broadcast packet, the buffered broadcast packet is discarded. If all children have not received a buffered broadcast packet and a new broadcast is received, the old broadcast packet is discarded, the child table flags are cleared, and the new broadcast packet is buffered for the end device children. This is demonstrated in the figure below.



End Device Child Table

Address	Received Broadcast
Ox2120	T
OxF220	F
OxC100	F
Ox5750	T

When an end device sends data to its parent that is destined for a remote device in the network, the parent buffers the data packet until it can establish a route to the destination. The parent may perform a route or 16-bit address discovery in behalf of its end device children. Once a route is established, the parent sends the data transmission to the remote device.

End Device Poll Timeouts

To better support mobile end devices (end devices that can move around in a network), parent router and coordinator devices have a poll timeout for each end device child. If an end device does not send a poll request to its parent within the poll timeout, the parent will remove the end device from its child table. This allows the child table on a router or coordinator to better accommodate mobile end devices in the network.

Packet Buffer Usage

Packet buffer usage on a router or coordinator varies depending on the application. The following activities can require use of packet buffers for up to several seconds:

- Route and address discoveries
- Application broadcast transmissions
- Stack broadcasts (i.e. ZDO "Device Announce" messages when devices join a network)
- Unicast transmissions (buffered until acknowledgment is received from destination or retries exhausted)
- Unicast messages waiting for end device to wake.

Applications that use regular broadcasting or that require regular address or route discoveries will use up a significant number of buffers, reducing the buffer availability for managing packets for end device children. Applications should reduce the number of required application broadcasts, and consider implementing an external address table or many-to-one and source routing if necessary to improve routing efficiency.

Non-Parent Device Operation

Devices in the ZigBee network treat data transmissions to end devices differently than transmissions to other routers and coordinators. Recall that when a unicast transmission is sent, if a network acknowledgment is not received within a timeout, the device resends the transmission. When transmitting data to remote coordinator or router devices, the transmission timeout is relatively short since these devices are powered and responsive. However, since end devices may sleep for some time, unicast transmissions to end devices use an extended timeout mechanism in order to allow enough time for the end device to wake and receive the data transmission from its parent.

If a non-parent device does not know the destination is an end device, it will use the standard unicast timeout for the transmission. However, provisions exist in the Ember ZigBee stack for the parent to inform the message sender that the destination is an end device. Once the sender discovers the destination device is an end device, future transmissions will use the extended timeout. See the XBee Router / Coordinator Configuration section in this chapter for details.

XBee End Device Configuration

XBee end devices support two different sleep modes:

- Pin Sleep
- Cyclic Sleep.

Pin sleep allows an external microcontroller to determine when the XBee should sleep and when it should wake by controlling the Sleep_RQ pin. In contrast, cyclic sleep allows the sleep period and wake times to be configured through the use of AT commands. The sleep mode is configurable with the SM command.

In both pin and cyclic sleep modes, XBee end devices poll their parent while they are awake to retrieve buffered data. When a poll request has been sent, the end device enables the receiver until an acknowledgment is received from the parent. (It generally takes less than 10ms from the time the poll request is sent until the acknowledgment is received.) The acknowledgment indicates if the parent has buffered data for the end device child or not. If the acknowledgment indicates the parent has pending data, the end device will leave the receiver on to receive the data. Otherwise, the end device will turn off the receiver and enter idle mode (until the next poll request is sent) to reduce current consumption (and improve battery life).

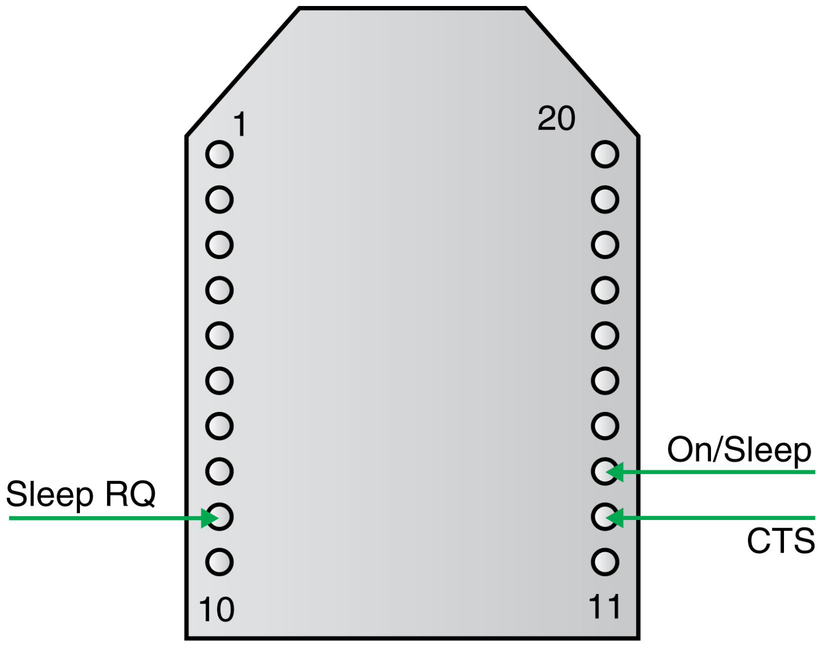
Once the module enters sleep mode, the On/Sleep pin (pin 13) is de-asserted (low) to indicate the module is entering sleep mode. If CTS hardware flow control is enabled (D7 command), the CTS pin (pin 12) is de-asserted (high) when entering sleep to indicate that serial data should not be sent to the module. The module will not respond to serial or RF data when it is sleeping. Applications that must communicate serially to sleeping end devices are encouraged to observe CTS flow control.

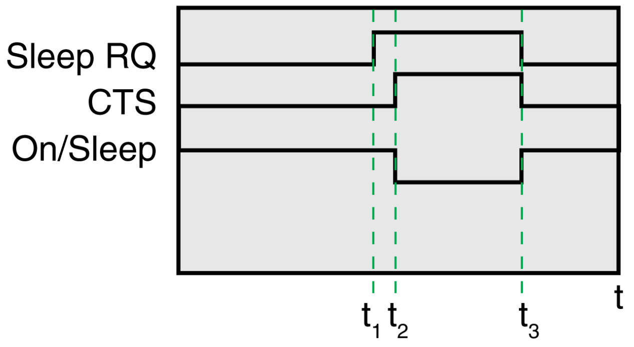
When the XBee wakes from sleep, the On/Sleep pin is asserted (high), and if flow control is enabled, the CTS pin is also asserted (low). If the module has not joined a network, it will scan all SC channels after waking to try and find a valid network to join.

Pin Sleep

Pin sleep allows the module to sleep and wake according to the state of the Sleep_RQ pin (pin 9). Pin sleep mode is enabled by setting the SM command to 1.

When Sleep_RQ is asserted (high), the module will finish any transmit or receive operations and enter a low power state. For example, if the module has not joined a network and Sleep_RQ is asserted (high), the module will sleep once the current join attempt completes (i.e. when scanning for a valid network completes). The module will wake from pin sleep when the Sleep_RQ pin is de-asserted (low).





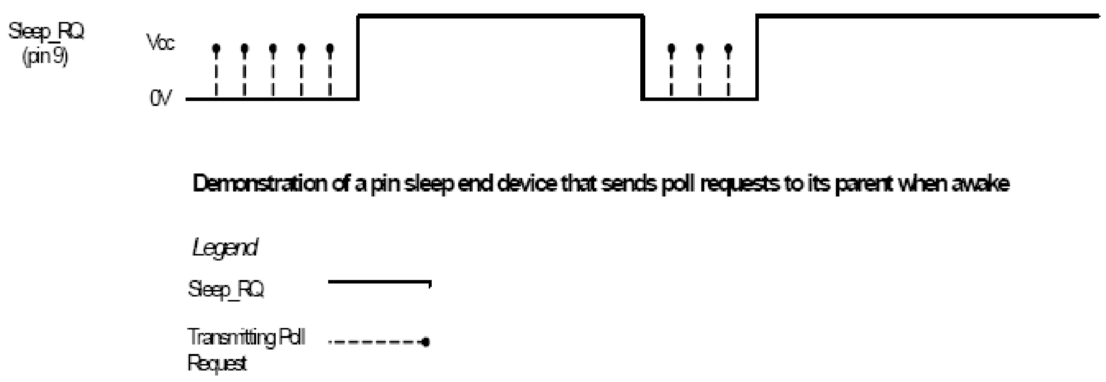
In the figure above, t1, t2, and t3 represent the following events:

- T1 - Time when Sleep_RQ is asserted (high)
- T2 - Time when the XBee enters sleep (CTS state change only if hardware flow control is enabled)
- T3 - Time when Sleep_RQ is de-asserted (low) and the module wakes.

The time between T1 and T2 varies depending on the state of the module. In the worst case scenario, if the end device is trying to join a network, or if it is waiting for an acknowledgment from a data transmission, the delay could be up to a few seconds.

When the XBee is awake and is joined to a network, it sends a poll request to its parent to see if the parent has any buffered data for it. The end device will continue to send poll requests every 100ms while it is awake.

Demonstration of Pin Sleep

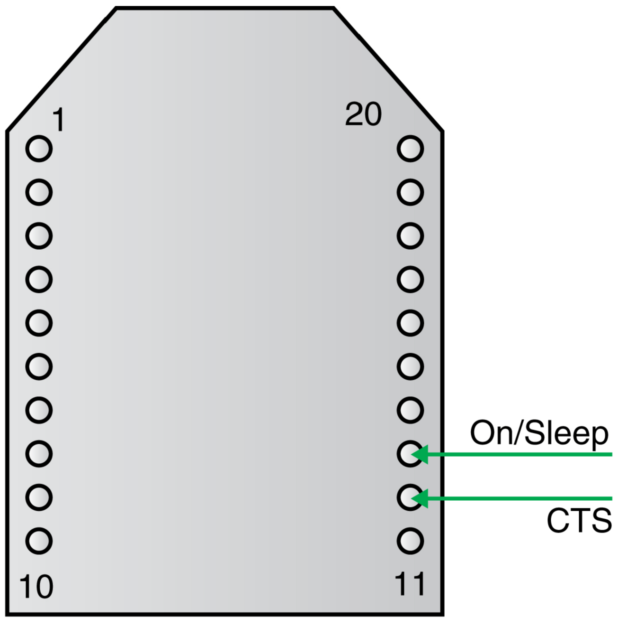

Sleep_RQ (pin 9)
Vcc
0V
Demonstration of a pin sleep end device that sends poll requests to its parent when awake
Legend
Sleep_RQ
Transmitting Poll Request

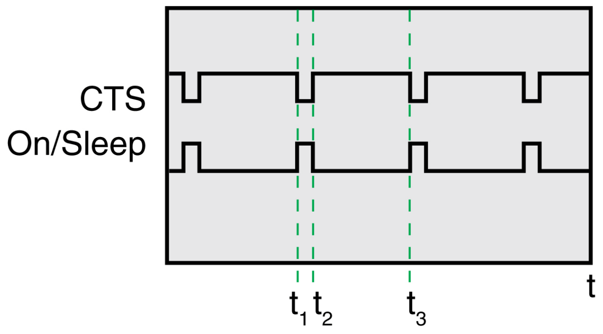
Parent and remote devices must be configured to buffer data correctly and to utilize adequate transmission timeouts. See the XBee Router / Coordinator Configuration section in this chapter for details.

Cyclic Sleep

Cyclic sleep allows the module to sleep for a specified time and wake for a short time to poll its parent for any buffered data messages before returning to sleep again. Cyclic sleep mode is enabled by setting the SM command to 4 or 5. SM5 is a slight variation of SM4 that allows the module to be woken prematurely by asserting the Sleep_RQ pin (pin 9). In SM5, the XBee can wake after the sleep period expires, or if a high-to-low transition occurs on the Sleep_RQ pin. Setting SM to 4 disables the pin wake option.

In cyclic sleep, the module sleeps for a specified time, and then wakes and sends a poll request to its parent to discover if the parent has any pending data for the end device. If the parent has buffered data for the end device, or if serial data is received, the XBee will remain awake for a time. Otherwise, it will enter sleep mode immediately.





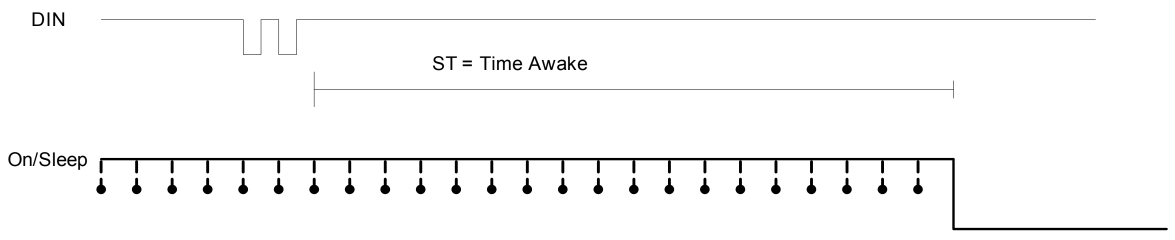
In the figure above, t1, t2, and t3 represent the following events:

- T1 - Time when the module wakes from cyclic sleep
- T2 - Time when the module returns to sleep
- T3 - Later time when the module wakes from cyclic sleep.

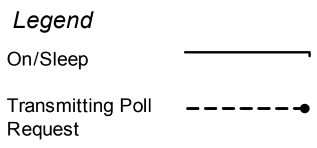
The wake time and sleep time are configurable with software commands as described in the sections below.

Wake Time (Until Sleep)

In cyclic sleep mode (SM=4 or 5), if serial or RF data is received, the module will start a sleep timer (time until sleep). Any data received serially or over the RF link will restart the timer. The sleep timer value is settable with the ST command. While the module is awake, it will send poll request transmissions to check its parent for buffered data messages. The module returns to sleep when the sleep timer expires, or if the SI command is sent to it. The following image shows this behavior.



A cyclic sleep end device enters sleep mode when no serial or RF data is received for ST time .



Sleep Period

The sleep period is configured based on the SP, SN, and SO commands. The following table lists the behavior of these commands.

Com-mand	Range	Description
SP	0x20 - 0xAF0 (x 10 ms) (320 - 28,000 ms)	Configures the sleep period of the module.
SN	1 - 0xFFFF	Configures the number of sleep periods multiplier.
SO	0 - 0xFF	Defines options for sleep mode behavior. 0x02 - Always wake for full ST time 0x04 - Enable extended sleep (sleep for full (SP * SN) time)

The XBee module supports both a short cyclic sleep and an extended cyclic sleep that make use of these commands. These two modes allow the sleep period to be configured according to the application requirements.

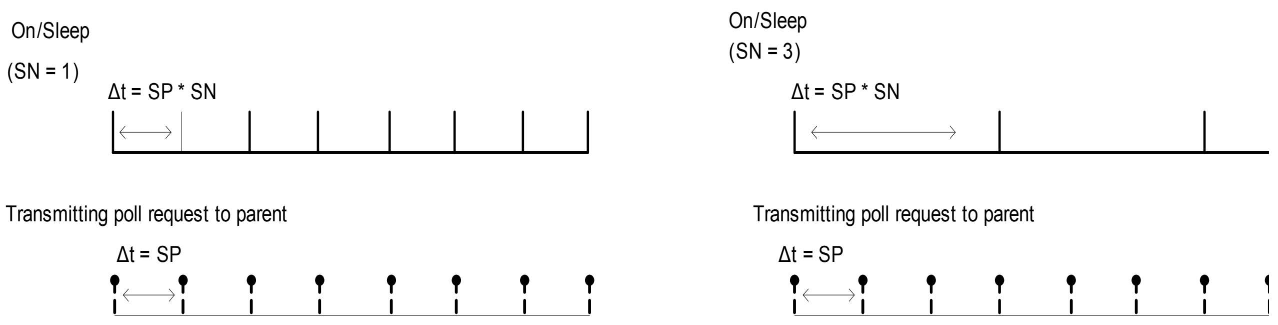
Short Cyclic Sleep

In short cyclic sleep mode, the sleep behavior of the module is defined by the SP and SN commands, and the SO command must be set to 0x00 (default) or 0x02. In short cyclic sleep mode, the SP command defines the sleep period and is settable up to 28 seconds. When the XBee enters short cyclic sleep, it remains in a low power state until the SP time has expired.

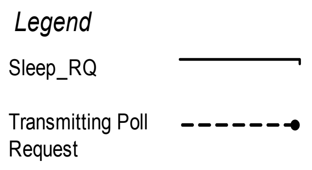
The Smart Energy specification recommends a nominal sleep cycle of between 7.5 and 60 seconds to mitigate congestion due to poll requests between an end device and its parent while maintaining an adequate poll rate. The default SP setting is 0x2EE, or 7.5 seconds.

After the sleep period expires, the XBee sends a poll request transmission to its parent to determine if its parent has any buffered data waiting for the end device. Since router and coordinator devices can buffer data for end device children up to 30 seconds, the SP range (up to 28 seconds) allows the end device to poll regularly enough to receive buffered data. If the parent has data for the end device, the end device will start its sleep timer (ST) and continue polling to receive data. If the end device wakes and finds that its parent has no data for it, the end device can return to sleep immediately.

The SN command can be used to control when the On/Sleep line is asserted (high). If SN is set to 1 (default), the On/Sleep line will be set high each time the XBee wakes from sleep. Otherwise, if SN is greater than 1, the On/Sleep line will only be set high if RF data is received, or after SN wake cycles occur. This allows an external device to remain powered off until RF data is received, or until a number of sleep periods have expired (SN sleep periods). This mechanism allows the XBee to wake at regular intervals to poll its parent for data without waking an external device for an extended time (SP * SN time). This is shown in the figure below.


On/Sleep
(SN = 1)
Δt = SP * SN
Transmitting poll request to parent
Δt = SP
On/Sleep
(SN = 3)
Δt = SP * SN
Transmitting poll request to parent
Δt = SP

Setting SN > 1 allows the XBee to silently poll for data without asserting On /Sleep. If RF data is received when polling, On/Sleep will immediately assert .



Note: SP controls the packet buffer time on routers and coordinators. SP should be set on all router and coordinator devices to match the longest end device SP time. See the XBee Router / Coordinator Configuration section for details.

Extended Cyclic Sleep

In extended cyclic sleep operation, an end device can sleep for a multiple of SP time which can extend the sleep time up to several days. The sleep period is configured using the SP and SN commands. The total sleep period is equal to (SP * SN) where SP is measured in 10ms units. The SO command must be set correctly to enable extended sleep.

Since routers and coordinators can only buffer incoming RF data for their end device children for up to 30 seconds, if an end device sleeps longer than 30 seconds, devices in the network need some indication when an end device is awake before they can send data to it. End devices that use extended cyclic sleep should send a transmission when they wake to inform other devices that they are awake and can receive data. It is recommended that extended sleep end devices set SO to wake for the full ST time in order to provide other devices with enough time to send messages to the end device.

Similar to short cyclic sleep, end devices running in this mode will return to sleep when the sleep timer expires, or when the SI command is received.

Transmitting RF Data

An end device may transmit data when it wakes from sleep and has joined a network. End devices transmit directly to their parent and then wait for an acknowledgment to be received. The parent will perform any required address and route discoveries to help ensure the packet reaches the intended destination before reporting the transmission status to the end device.

Receiving RF Data

After waking from sleep, an end device sends a poll request to its parent to determine if the parent has any buffered data for it. In pin sleep mode, the end device polls while the Sleep_RQ pin is de-asserted (low). In cyclic sleep mode, the end device will only poll once before returning to sleep unless the sleep timer (ST) is started (serial or RF data is received). If the sleep timer is started, the end device will continue to poll every 100ms until the sleep timer expires.

If an end device receives RF data from its parent, it sends another poll after a very short delay to check for more data. The end device continues to poll at a faster rate as long as it receives data from its parent. This feature greatly improves data throughput to end devices. When the end device no longer receives data from its parent, it resumes polling at the regular rate.

Waking End Devices with the Commissioning Pushbutton

A high-to-low transition on the AD0/DIO0 pin (pin 20) will cause an end device to wake for 30 seconds. See the Commissioning Pushbutton section in chapter 7 for details.

Parent Verification

Since an end device relies on its parent to maintain connectivity with other devices in the network, XBee end devices include provisions to verify its connection with its parent. End devices monitor their link with their parent when sending poll messages and after a power cycle or reset event as described below.

When an end device wakes from sleep, it sends a poll request to its parent. In cyclic sleep, if RF or serial data is not received and the sleep timer is not started, the end device polls one time and returns to sleep for another sleep period. Otherwise, the end device continues polling. If the parent does not send an acknowledgment response to three consecutive poll request transmissions, the end device assumes the parent is out of range, and attempts to find a new parent.

After a power-up or reset event, the end device does an orphan scan to locate its parent. If the parent does not send a response to the orphan scan, the end device attempts to find a new parent.

Rejoining

Once all devices have joined a ZigBee network, the permit-joining attribute should be disabled (NJ0) so that new devices are no longer allowed to join the network. Permit-joining can be enabled later as needed for short times. This provides some protection in preventing other devices from joining a live network.

If an end device cannot communicate with its parent, the end device must be able to join a new parent to maintain network connectivity. However, if permit-joining is disabled in the network, the end device will not find a device that is allowing new joins.

To overcome this problem, ZigBee supports rejoining, where an end device can obtain a new parent in the same network even if joining is not enabled. When an end device joins using rejoining, it performs a PAN ID scan to discover nearby networks. If a network is discovered that has the same 64-bit PAN ID as the end device, it will join the network by sending a rejoin request to one of the discovered devices. The device that receives the rejoin request will send a rejoin response if it can allow the device to join the network (i.e. child table not full). The rejoin mechanism can be used to allow a device to join the same network even if permit-joining is disabled.

If a device is commanded to leave a network, it will erase the ephemeral link key from its memory, but the trust center will keep a record of that device with its ephemeral link key in its key table. Later, to join that device back into the network, then a Zigbee Register Joining Device frame should be sent to the trust center to re-register that device’s preconfigured link key with the device’s EUI64 address into the key table. Otherwise, the trust center will consider a subsequent joining attempt by that device to be an attempt by a malicious device to spoof the identity of the joining device, and the join will fail.

XBee Router/Coordinator Configuration

XBee routers and coordinators may require some configuration to ensure the following are set correctly:

- RF packet buffering timeout
- Child poll timeout
- Transmission timeout.

The value of these timeouts depends on the sleep time used by the end devices. Each of these timeouts are discussed below.

RF Packet Buffering Timeout

When a router or coordinator receives an RF data packet intended for one of its end device children, it buffers the packet until the end device wakes and polls for the data, or until a packet buffering timeout occurs. This timeout is settable using the SP command. The actual timeout is (1.2 * SP), with a minimum timeout of 1.2 seconds and a maximum of 30 seconds. Since the packet buffering timeout is set slightly larger than the SP setting, SP should be set the same on routers and coordinators as it is on cyclic sleep end devices. For pin sleep devices, SP should be set as long as the pin sleep device can sleep, up to 30 seconds.

Note: In pin sleep and extended cyclic sleep, end devices can sleep longer than 30 seconds. If end devices sleep longer than 30 seconds, parent and non-parent devices must know when the end device is awake in order to reliably send data. For applications that require sleeping longer than 30 seconds, end devices should transmit data when they wake to alert other devices that they can send data to the end device.

Child Poll Timeout

Router and coordinator devices maintain a timestamp for each end device child indicating when the end device sent its last poll request to check for buffered data packets. If an end device does not send a poll request to its parent for a certain period of time, the parent will assume the end device has moved out of range and will remove the end device from its child table. This allows routers and coordinators to be responsive to changing network conditions. The NC command can be issued at any time to read the number of remaining (unused) child table entries on a router or coordinator.

The child poll timeout is settable with the SP and SN commands. SP and SN should be set such that SP * SN matches the longest expected sleep time of any end devices in the network. The actual timeout is calculated as (3 * SP * SN), with a minimum of 5 seconds. For networks consisting of pin sleep end devices, the SP and SN values on the coordinator and routers should be set such that SP * SN matches the longest expected sleep period of any pin sleep device. The 3 multiplier ensures the end device will not be removed unless 3 sleep cycles pass without receiving a poll request. The poll timeout is settable up to a couple of months.

Adaptive Polling

The PO command determines the regular polling rate. But if RF data has been recently received by an end device, it is likely that more RF data may yet be received. In that event, the end device will poll at a faster rate, gradually decreasing its adaptive poll rate until polling resumes at the regular rate as defined by the PO command.

Transmission Timeout

As mentioned in chapter 4, when sending RF data to a remote router, since routers are always on, the timeout is based on the number of hops the transmission may traverse. This timeout it settable using the NH command. (See chapter 4 for details.)

Since end devices may sleep for lengthy periods of time, the transmission timeout to end devices also includes some allowance for the sleep period of the end device. When sending data to a remote end device, the transmission timeout is calculated using the SP and NH commands. If the timeout occurs and an acknowledgment has not been received, the source device will resend the transmission until an acknowledgment is received, up to two more times.

The transmission timeout per attempt is:

3 * ((unicast router timeout) + (end device sleep time)), or

3 * ((50 * NH) + (1.2 * SP)), where SP is measured in 10ms units.

For best results, SP should be set on routers and coordinator devices to match the SP setting on the end devices.

Note: The NH command is used to determine the timeout when transmitting to routers.

Putting it all Together

Short Sleep Periods

Pin and cyclic sleep devices that sleep less than 30 seconds can receive data transmissions at any time since their parent device(s) will be able to buffer data long enough for the end devices to wake and poll to receive the data. SP should be set the same on all devices in the network. If end devices in a network have more than one SP setting, SP on the routers and coordinators should be set to match the largest SP setting of any end device. This will ensure the RF packet buffering, poll timeout, and transmission timeouts are set correctly.

Extended Sleep Periods

Pin and cyclic sleep devices that might sleep longer than 30 seconds cannot receive data transmissions reliably unless certain design approaches are taken. Specifically, the end devices should use IO sampling or another mechanism to transmit data when they wake to inform the network they can receive data. SP and SN should be set on routers and coordinators such that (SP * SN) matches the longest expected sleep time. This configures the poll timeout so end devices are not expired from the child table unless a poll request is not received for 3 consecutive sleep periods.

As a general rule of thumb, SP and SN should be set the same on all devices in almost all cases.

Sleep Examples

This section covers some sample XBee configurations to support different sleep modes. Several AT commands are listed with suggested parameter values. The notation in this section includes an '=' sign to indicate what each command register should be set to - for example, SM=4. This is not the correct notation for setting command values in the XBee. In the API, the two byte command is used in the command field, and parameters are populated as binary values in the parameter field.

Example 1

Configure a device to sleep for 20 seconds, but set SN such that the On/Sleep line will remain de-asserted for up to 1 minute.

The following settings should be configured on the end device.

SM = 4 (cyclic sleep) or 5 (cyclic sleep, pin wake)

SP = 0x7D0 (2000 decimal). This causes the end device to sleep for 20 seconds since SP is measured in units of 10ms.

SN = 3. (With this setting, the On/Sleep pin will assert once every 3 sleep cycles, or when RF data is received)

SO = 0

All router and coordinator devices on the network should set SP to match SP on the end device. This ensures that RF packet buffering times and transmission timeouts will be set correctly.

Since the end device wakes after each sleep period (SP), the SN command can be set to 1 on all routers and the coordinator.

All router and coordinator devices on the network should set SP to match SP on the end device. This ensures that RF packet buffering times and transmission timeouts will be set correctly.

Example 2

Configure a device for extended sleep: to sleep for 4 minutes.

SP and SN must be set such that SP * SN = 4 minutes. Since SP is measured in 10ms units, the following settings can be used to obtain 4 minute sleep.

SM = 4 (cyclic sleep) or 5 (cyclic sleep, pin wake)

SP = 0x7D0 (2000 decimal, or 20 seconds)

SN = 0x0B (12 decimal)

SO = 0x04 (enable extended sleep)

With these settings, the module will sleep for SP * SN time, or (20 seconds * 12) = 240 seconds = 4 minutes.

For best results, the end device should send a transmission when it wakes to inform the coordinator (or network) when it wakes. It should also remain awake for a short time to allow devices to send data to it. The following are recommended settings.

ST = 0x7D0 (2 second wake time)

SO = 0x06 (enable extended sleep and wake for ST time)

SP and SN should be set to the same values on all routers and coordinators that could allow the end device to join. This will ensure the parent does not timeout the end device from its child table too quickly.

The SI command can optionally be sent to the end device to cause it to sleep before the sleep timer expires.

7. Network Commissioning and Diagnostics

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

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

Device Discovery

ZDO Discovery

The ZigBee Device Profile includes provisions to discover devices in a network that are supported on all ZigBee devices (including non-Digi products). These include the LQI Request (cluster ID 0x0031) and the Network Update Request (cluster ID 0x0038). The LQI Request can be used to read the devices in the neighbor table of a remote device, and the Network Update Request can be used to have a remote device do an active scan to discover all nearby ZigBee devices. Both of these ZDO commands can be sent using the XBee Explicit API transmit frame (0x11). See the API chapter for details. Refer to the ZigBee specification for formatting details of these two ZDO frames.

Joining Announce

All ZigBee devices send a ZDO Device Announce broadcast transmission when they join a ZigBee network (ZDO cluster ID 0x0013). These frames will be sent out the XBee's UART as an Explicit Rx Indicator API frame (0x91) if AO is set to 1. The device announce payload includes the following information:

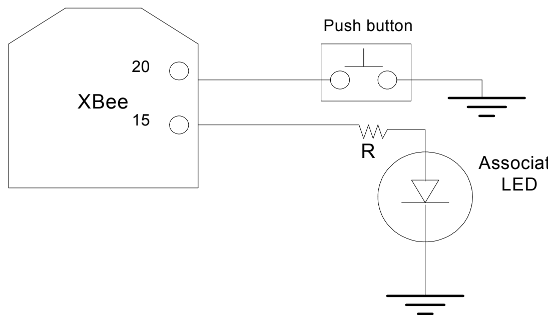
[Sequence Number] + [16-bit address] + [64-bit address] + [Capability]

The 16-bit and 64-bit addresses are received in little-endian byte order (LSB first). See the ZigBee specification for details.

Commissioning Pushbutton and Associate LED

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

Commissioning Pushbutton and Associate LED Functionalities



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

Commissioning Pushbutton

The commissioning pushbutton definitions provide a variety of simple functions to aid in deploying devices in a network. The commissioning button functionality on pin 20 is enabled by default on Smart Energy devices.

Button Presses	If module is joined to a network	If module is not joined to a network
1	• Wakes an end device for 60 seconds	• Wakes an end device for 60 seconds
2	• Sends a broadcast transmission to enable joining on the coordinator and all devices in the network for 1 minute.	• N/A
4	• Causes the device to leave the PAN. • Issues RE to restore module parameters to default values, including ID and SC. Exception: no RE is applied to the SE Range Extender. • The device attempts to join a network based on its ID and SC settings.	• Issues RE to restore module parameters to default values, including ID and SC. Exception: no RE is applied to the SE Range Extender. • The device attempts to join a network based on its ID and SC settings.

Button presses may be simulated in software using the CB command. CB should be issued with a parameter set to the number of button presses to execute. (e.g. sending CB1 will execute the action(s) associated with a single button press.)

Associate LED

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

Joined Indication

The Associate pin indicates the network status of a device. If the module is not joined to a network, the Associate pin is set high. Once the module successfully joins a network, the Associate pin blinks at a regular time interval.

Joined Status of a Device

The LT command defines the blink time of the Associate pin. If set to 0, the device uses the default blink time (500ms for coordinator, 250ms for routers and end devices).

Smart Energy Range Extender

The Smart Energy Range Extender's Associate LED indicates its join status; once joined it also indicates the status of its connection to the network's Coordinator. LT settings do not affect the blink rates of the Smart Energy Range Extender.

LED Status	Network Association
On, solid green	Joined, and network connection to Coordinator is working
On, 3 sec blink	Not joined, AI register indicates reason
On, 1 sec blink	Trying to join
On, 1/4 sec blink	Joined, but connection to Coordinator is not working

Once the Range Extender is joined to a network, the status of its connection to the Coordinator is updated every 30 seconds.

The state of the LED1 on development boards is the reverse of the Associate LED. When the Associate LED is on, LED1 is off; when the Associate LED is off, LED1 is on.

8. API Operation

API (Application Programming Interface) Operations are available for communicating with an external processor through its UART port. API operation requires that communication with the module be done through a structured interface (data is communicated in frames in a defined order). The API specifies how commands, command responses and module status messages are sent and received from the module using a UART Data Frame.

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

API Frame Specifications

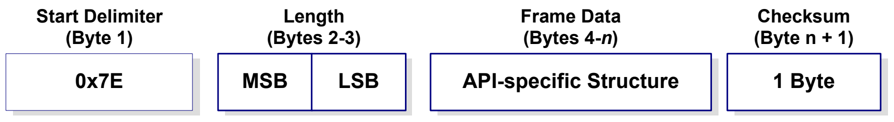
Two API modes are supported and both can be enabled using the AP (API Enable) command. Use the following AP parameter values to configure the module to operate in a particular mode:

- AP = 1: API Operation
- AP = 2: API Operation (with escaped characters)

API Operation (AP parameter = 1)

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

UART Data Frame Structure:



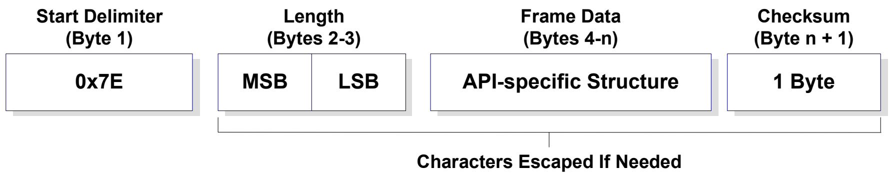
MSB = Most Significant Byte, LSB = Least Significant Byte

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

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

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

UART Data Frame Structure - with escape control characters:



MSB = Most Significant Byte, LSB = Least Significant Byte

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

Data bytes that need to be escaped:

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

Example - Raw 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 above example, the length of the raw data (excluding the checksum) is 0x0002 and the checksum of the non-escaped data (excluding frame delimiter and length) is calculated as: 0xFF - (0x23 + 0x11) = (0xFF - 0x34) = 0xCB.

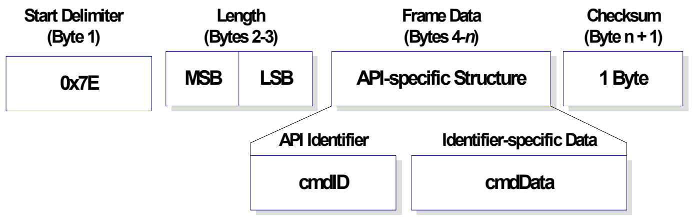
Length

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

Frame Data

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

UART Data Frame & API-specific Structure:



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

Table 8-01. API Frame Names and Values

API Frame Names	API ID
AT Command	0x08
AT Command - Queue Parameter Value	0x09
Explicit Addressing ZigBee Command Frame	0x11
ZigBee Create Source Route	0x21
ZigBee Register Joining Device	0x24
AT Command Response	0x88
Modem Status	0x8A
ZigBee Transmit Status	0x8B
ZigBee Explicit Rx Indicator (AO=1)	0x91
Route Record Indicator	0xA1
Device Authenticated Indicator	0xA2
Many-to-One Route Request Indicator	0xA3
ZigBee Register Joining Device Status	0xA4

Checksum

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

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

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

API Examples

Example: Create an API AT command frame to configure an XBee to allow joining (set NJ to 0xFE). The frame should look like:

0x7E 0x00 0x05 0x08 0x01 0x4E 0x4A 0xFE 60

Where 0x0005 = length

0x08 = AT Command API frame type

0x01 = Frame ID (set to non-zero value)

0x4E4A = AT Command ('NJ')

0xFE = value to set command to

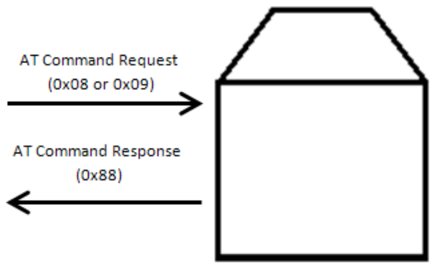
0x60 = Checksum

The checksum is calculated as [0xFF - (0x08 + 0x01 + 0x4E + 0x4A + 0xFE) & 0xFF]

API UART Exchanges

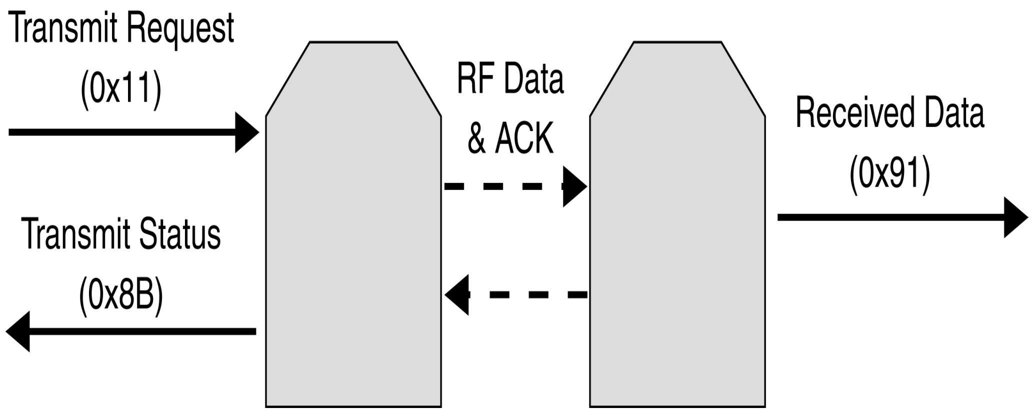
AT Commands

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



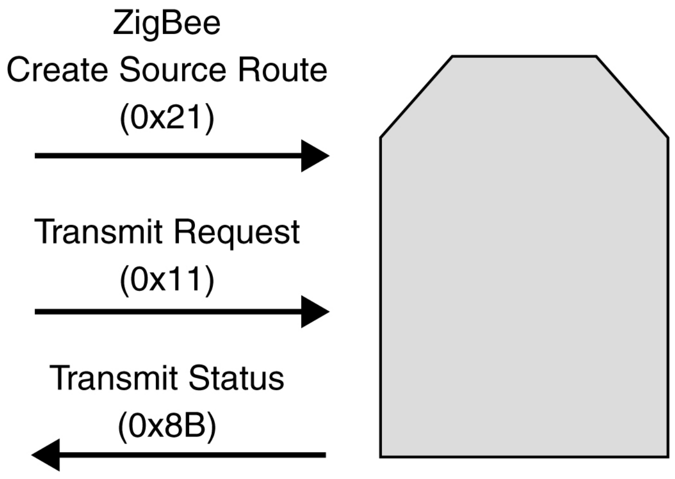
Transmitting and Receiving RF Data

The following image shows the API exchanges that take place at the UART when sending RF data to another device. The transmit status frame is always sent at the end of a data transmission unless the frame ID is set to 0 in the transmit request. If the packet cannot be delivered to the destination, the transmit status frame will indicate the cause of failure.



Source Routing

The following image shows the API frame exchanges that take place at the UART when sending a source routed transmission.



Supporting the API

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

```
                case 0x11:
Parse_ExplicitAddressingZigBeeCommandFrame (papiFrame);
break;
                case 0x17:
Parse_ZigBeeRemoteATCommand(papiFrame); break;
                case 0x24:
Parse_ZigBeeRegisterJoiningDevice (papiFrame); break;
                case 0x8B: Parse_ZigBeeTxStatus(papiFrame);
break;
                case 0x90: Parse_ZigBeeRxIndicator(papiFrame);
break;
                case 0x21:
Parse_ZigBeeCreateSourceRoute(papiFrame); break;
                case 0x91:
Parse_ZigBeeExplicitRxIndicator (papiFrame); break;
                case 0xA1:
Parse_ZigBeeRouteRecordIndicator(papiFrame); break;
                case 0xA2:
Parse_ZigBeeDeviceAuthenticatedIndicator (papiFrame); break;
                case 0xA3:
Parse_ZigBeeManyToOneRouteRequestIndicator (papiFrame);
break;
                case 0xA4:
Parse_ZigBeeRegisterJoiningDeviceStatus (papiFrame); break;
                default:
                // Discard any other API frame types that are not
being used
                break;
        }
}
```

API Frames

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

AT Command

Frame Type: 0x08
Used to query or set module parameters on the local device. This API command applies changes after executing the command. (Changes made to module parameters take effect once changes are applied.) The API example below illustrates an API frame when modifying the NJ parameter value of the module.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x05	
	Frame-specific Data	Frame Type	3	0x08	
		Frame ID	4	0x52 (R)	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		AT Command	5	0x4E (N)	Command Name - Two ASCII characters that identify the AT Command.
			6	0x4A (J)	
		Parameter Value	7	0xE0	If present, indicates the requested parameter value to set the given register. If no characters present, register is queried.
	Checksum		8	0x2D	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

The above example illustrates an AT Command when setting an NJ value.

AT Command - Queue Parameter Value

Frame Type: 0x09
This API type allows module parameters to be queried or set. In contrast to the “AT Command” API type, new parameter values are queued and not applied until either the “AT Command” (0x08) API type or the AC (Apply Changes) command is issued. Register queries (reading parameter values) are returned immediately.

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

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x05	
	Frame-specific Data	Frame Type	3	0x09	
		Frame ID	4	0x01	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		AT Command	5	0x42 (B)	Command Name - Two ASCII characters that identify the AT Command.
			6	0x44 (D)	
		Parameter Value (BD7 = 115200 baud)	7	0x07	If present, indicates the requested parameter value to set the given register. If no characters present, register is queried.
	Checksum		8	0x68	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

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

Explicit Addressing ZigBee Command Frame

Frame Type: 0x11

Allows ZigBee application layer fields (endpoint and cluster ID) to be specified for a data transmission.

It is similar to the ZB API Frame ZigBee Transmit Request (0x10), but also requires ZigBee application layer addressing fields to be specified (endpoints, cluster ID, profile ID). An Explicit Addressing Request API frame causes the module to send data as an RF packet to the specified destination, using the specified source and destination endpoints, cluster ID, and profile ID.

The 64-bit destination address should be set to 0x000000000000FFFF for a broadcast transmission (to all devices). The coordinator can be addressed by either setting the 64-bit address to all 0x00s and the 16-bit address to 0xFFFE, OR by setting the 64-bit address to the coordinator's 64-bit address and the 16-bit address to 0x0000. For all other transmissions, setting the 16-bit address to the correct 16-bit address can help improve performance when transmitting to multiple destinations. If a 16-bit address is not known, this field should be set to 0xFFFE (unknown). The Transmit Status frame (0x8B) will indicate the discovered 16-bit address, if successful.

The broadcast radius can be set 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. The options byte may be set to 0x20 to invoke APS end-to-end security. This requires an application link key to have been previously established between the source and destination nodes.

The maximum number of payload bytes can be read with the NP command. Note: if source routing is used, the RF payload will be reduced by two bytes per intermediate hop in the source route.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x1A	
	Frame-specific Data	Frame Type	3	0x11	
		Frame ID	4	0x01	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		64-bit Destination Address	MSB 5	0x00	Set to the 64-bit address of the destination device. The following addresses are also supported: 0x0000000000000000 - Reserved 64-bit address for the coordinator 0x000000000000FFFF - Broadcast address
			6	0x00	
			7	0x00	
			8	0x00	
			9	0x00	
			10	0x00	
			11	0x00	
			12	0x00	
		16-bit Destination Network Address	MSB 13	0xFF	Set to the 16-bit address of the destination device, if known. Set to 0xFFFE if the address is unknown, or if sending a broadcast.
			LSB 14	0xFE	
		Source Endpoint	15	0xA0	Source endpoint for the transmission.
		Destination Endpoint	16	0xA1	Destination endpoint for the transmission.
		Cluster ID	17	0x15	Cluster ID used in the transmission
			18	0x54	
		Profile ID	19	0xC1	Profile ID used in the transmission
			20	0x05	
		Broadcast Radius	21	0x00	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value.
		Transmit Options	22	0x00	0, or 0x20 if APS end-to-end security should be invoked.
		Data Payload	23	0x54	Data that is sent to the destination device
			24	0x78	
			25	0x44	
			26	0x61	
			27	0x74	
			28	0x61	
	Checksum		29	0x3A	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Send a data transmission to the coordinator (64-bit address of 0x00s) using a source endpoint of 0xA0, destination endpoint 0xA1, cluster ID =0x1554, and profile ID 0xC105. Payload will be "TxData".

ZigBee Create Source Route

Frame Type: 0x21

This frame creates a source route in the module. A source route specifies the complete route a packet should traverse to get from source to destination. Source routing should be used with many-to-one routing for best results.

Note: Both the 64-bit and 16-bit destination addresses are required when creating a source route. These are obtained when a Route Record Indicator (0xA1) frame is received.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x10	
	Frame-specific Data	Frame Type	3	0x21	
		Frame ID	4	0x01	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		64-bit Destination Address	MSB 5	0x00	Extended (64-bit) Address of the destination node.
			6	0x13	
			7	0xA2	
			8	0x00	
			9	0x40	
			10	0x40	
			11	0x11	
			LSB 12	0x22	
		16-bit Destination Network Address	MSB 13	0x33	Network (16-bit) Address of the destination node.
			LSB 14	0x44	
		Receive Options	15	0x00	Set to 0x00.
		# of Route Records	16	0x01	The number of route records (16-bit addresses) which follow.
		Route Record	17	0xEE	Each record holds a 16-bit address. The first record is the 16-bit address of the neighbor of the device that sent the route record. The last record is the 16-bit address of the last hop of the route record (neighbor of the recipient).
			18	0xFF	
	Checksum		19	0x10	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

This message is received through the UART to inform a concentrator (aggregator) of the source route to use when addressing a remote node. It is sent before sending a data packet to the concentrator by the remote node, so the concentrator knows how to route a response

ZigBee Register Joining Device

Frame Type: 0x24

A KY command can be used to set the new device's initial link key." as "A KY command can be used on a new device to set its initial link key." Registers a new device into the trust center's key table. A KY command can be used to set the new device’s initial link key.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x1C	
	Frame-specific Data	Frame Type	3	0x24	
		Frame ID	4	0x00	Identifies the UART data frame for the host to match with a subsequent TX Status. If set to 0, no response is sent.
		64-bit Destination Address	MSB 5	0x01	Set to the 64-bit address of the destination device.
			6	0x02	
			7	0x03	
			8	0x04	
			9	0x05	
			10	0x06	
			11	0x07	
			LSB 12	0x08	
		16-bit Destination Network Address	MSB 13	0xFF	Set to the 16-bit address of the destination device, if known. Set to 0xFFFE if the address is unknown.
			LSB 14	0xFE	
		Options	15	0x00	Set to 0.
		Key	16	0x01	The initial trust center link key of the device, settable up to 16 bytes. If less than 16 bytes are supplied, the upper unspecified bytes of the key will be zero padded. If a zero-length key field is supplied, then the device is removed from the link key table.
			17	0x02	
			18	0x03	
			19	0x04	
			20	0x05	
			21	0x06	
			22	0x07	
			23	0x08	
			24	0x09	
			25	0x0A	
			26	0x0B	
			27	0x0C	
			28	0x0D	
			29	0x0E	
			30	0x0F	
	Checksum		31	0x42	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

AT Command Response

Frame Type: 0x88
In response to an AT Command message, the module will send an AT Command Response message. Some commands will send back multiple frames.

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

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

Modem Status

Frame Type: (0x8A)
RF module status messages are sent from the module in response to specific conditions.

Example: The following API frame is returned when an API coordinator forms a network.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x02	
	Frame-specific Data	Frame Type	3	0x8A	
		Status	4	0x06	0 = Hardware reset 1 = Watchdog timer reset 2 =Joined network (routers and end devices) 3 =Disassociated 6 =Coordinator started 7 = Network security key was updated 13 = Voltage supply limit exceeded (XBee-PRO(S2B)) 0x10 = Key establishment complete 0x11 = Key configuration registers were changed while a join was already in progress 0x80 = stack error
	Checksum		5	0x6F	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Note: New modem status codes may be added in future firmware releases.

ZigBee Transmit Status

Frame Type: 0x8B

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

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x07	
	Frame-specific Data	Frame Type	3	0x8B	
		Frame ID	4	0x01	Identifies the UART data frame being reported. Note: If Frame ID = 0 in the AT Command frame, then no AT Command Response frame will be sent.
		16-bit address of destination	5	0x7D	16-bit Network Address the packet was delivered to (if success). If not success, this address matches the Destination Network Address that was provided in the Transmit Request Frame.
			6	0x84	
		Transmit Retry Count	7	0x00	The number of application transmission retries that took place.
		Delivery Status	8	0x00	0x00 = Success 0x02 = CCA Failure 0x18 = No Buffers 0x21 = Network ACK Failure 0x22 = Not Joined to Network 0x23 = Self-addressed 0x24 = Address Not Found 0x25 = Route Not Found 0x26 = Relay of Broadcast not heard 0x2B = Invalid Binding Table Index 0x2C = Invalid Endpoint 0x2D = Attempted Broadcast with APS encryption 0x2E = Attempted Unicast with APS encryption but EE=0 0x32 = Resource Error 0x74 = Data payload too large 0xBB = Key not authorized
		Discovery Status	9	0x01	0x00 = No Discovery Overhead 0x01 = Address Discovery 0x02 = Route Discovery 0x03 = Address and Route Discovery
	Checksum		10	0x71	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Suppose a unicast data transmission was sent to a destination device with a 16-bit address of 0x7D84. (The transmission could have been sent with the 16-bit address set to 0x7D84 or 0xFFFE.)

ZigBee Explicit Rx Indicator

Frame Type:0x91

When the modem receives a ZigBee RF packet it is sent out the UART using this message type (when AO=1 or 3).

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x18	
	Frame-specific Data	Frame Type	3	0x91	
		64-bit Source Address	MSB 4	0x00	64-bit address of sender. Set to 0xFFFFFFFFFFFFFFFF (unknown 64-bit address) if the sender's 64-bit address is unknown.
			5	0x13	
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x52	
			10	0x2B	
			LSB 11	0xAA	
		16-bit Source Network Address	MSB 12	0x7D	16-bit address of sender.
			LSB 13	0x84	
		Source Endpoint	14	0xE0	Endpoint of the source that initiated the transmission
		Destination Endpoint	15	0xE0	Endpoint of the destination the message is addressed to.
		Cluster ID	16	0x22	Cluster ID the packet was addressed to.
			17	0x11	
		Profile ID	18	0xC1	Profile ID the packet was addressed to.
			19	0x05	
		Receive Options	20	0x02	0x01 – Packet Acknowledged 0x02 – Packet was a broadcast packet
		Received Data	21	0x52	Received RF data
			22	0x78	
			23	0x44	
			24	0x61	
			25	0x74	
			26	0x61	
	Checksum		27	0x52	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Route Record Indicator

Frame Type: 0xA1

The route record indicator is received whenever a device sends a ZigBee route record command. This is used with many-to-one routing to create source routes for devices in a network.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x13	
	Frame-specific Data	Frame Type	3	0xA1	
		64-bit Source Address	MSB 4	0x00	64-bit address of the device that initiated the route record.
			5	0x13	
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x40	
			10	0x11	
			LSB 11	0x22	
		Source (updater) 16-bit Address	12	0x33	16-bit address of the device that initiated the route record.
			13	0x44	
		Receive Options	14	0x01	0x01 - Packet Acknowledged. 0x02 - Packet was a broadcast.
		Number of Addresses	15	0x03	The number of addresses in the source route (excluding source and destination).
		Address 1	16	0xEE	(neighbor of destination)
			17	0xFF	
		Address 2 (closer hop	18	0xCC	Address of intermediate hop
			19	0xDD	
		Address n (neighbor of source)	20	0xAA	Two bytes per 16-bit address.
			21	0xBB	
	Checksum		22	0x80	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Suppose device E sends a route record that traverses multiple hops en route to data collector device A as shown below.

A B C D E

If device E has the 64-bit and 16-bit addresses of 0x0013A200 40401122 and 0x3344, and if devices B, C, and D have the following 16-bit addresses:

B = 0xAABB

C = 0xCCDD

D = 0xEEFF

The data collector will send the above API frame out its UART.

ZigBee Device Authenticated Indicator

Frame Type: 0xA2

This frame is sent out the UART of the Trust Center when a new device is authenticated on a Smart Energy network.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x0C	
	Frame-specific Data	Frame Type	3	0xA2	
		64-bit Source Address	MSB 4	0x01	Extended 64-bit address of the authenticated device.
			5	0x02	
			6	0x03	
			7	0x04	
			8	0x05	
			9	0x06	
			10	0x07	
			LSB 11	0x08	
		16-bit Source Address	MSB 12	0xFF	16-bit address of the authenticated device.
			LSB 13	0xFE	
		Status	14	0x00	Success
	Checksum		15	0x3C	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

ZigBee Many-to-One Route Request Indicator

Frame Type: 0xA3

This message is sent out the UART when a device receives a many-to-one route request. It is only supported on routers and coordinator type devices. End devices will not receive this API frame.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x0C	
	Frame-specific Data	Frame Type	3	0xA3	
		64-bit Source Address	MSB 4	0x01	64-bit address of the device that sent the many-to-one route request
			5	0x02	
			6	0x03	
			7	0x04	
			8	0x05	
			9	0x06	
			10	0x07	
			LSB 11	0x08	
		16-bit Source Address	MSB 12	0xFF	16-bit address of the device that initiated the many-to-one route request.
			LSB 13	0xFE	
		Reserved	14	0x00	Set to 0.
	Checksum		15	0x3B	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

ZigBee Register Joining Device Status

Frame Type: 0xA4

This frame is sent out the UART of the Trust Center when a new device is authenticated on a Smart Energy network.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x03	
	Frame-specific Data	Frame Type	3	0xA4	
		Frame ID	4	0x04	Identifies the UART data frame for the host to match with a subsequent TX Status. It matches the Frame ID of the registration request (0x24).
		Status	5	0x00	**Version 3x19** 0x00 - Success 0xB3 - Invalid address 0xFF - Key not found **Version 3x1A** 0x00 - Success 0x01 - Key too long 0xB1 - Address not found in the key table 0xB2 - Key value is invalid; (a key value of 0x00 or 0xFF is reserved) 0xB4 - Key table is already full
	Checksum		6	0x57	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

ZigBee Join Notification Status

Frame Type: 0xA5

This frame is sent out the UART of the Trust Center when a device attempts to join, rejoin, or leave the network. It is enabled by setting bit 0x02 in the DO register.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x0E	
	Frame-specific Data	Frame Type	3	0xA5	Join Notification Status
		16-bit Parent Address	4	0x12	16-bit address of the parent device
			5	0x34	
		16-bit New Node Address	6	0x12	Proposed 16-bit address for the joining device
			7	0x34	
		64-bit New Node Address	8	0x01	64-bit MAC address of the joining device
			9	0x23	
			10	0x45	
			11	0x67	
			12	0x89	
			13	0xAB	
			14	0xCD	
			15	0xEF	
		Status	16	0x00	0x00 - Standard Security Secured Rejoin 0x01 - Standard Security Unsecured Join 0x02 - Device Left 0x03 - Standard Security Unsecured Rejoin 0x04 - High Security Secured Rejoin 0x05 - High Security Unsecured Join 0x07 - High Security Unsecured Rejoin
	Checksum		17	0x0E	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Sending ZigBee Device Objects (ZDO) Commands with the API

ZigBee Device Objects (ZDOs) are defined in the ZigBee Specification as part of the ZigBee Device Profile. These objects provide functionality to manage and map out the ZigBee network and to discover services on ZigBee devices. ZDOs are typically required when developing a ZigBee product that will interoperate in a public profile such as home automation or Smart Energy, or when communicating with ZigBee devices from other vendors. The ZDO can also be used to perform several management functions such as frequency agility (energy detect and channel changes - Mgmt Network Update Request), discovering routes (Mgmt Routing Request) and neighbors (Mgmt LQI Request), and managing device connectivity (Mgmt Leave and Mgmt Permit Joining Request).

The following table shows some of the more prominent ZDOs with their respective cluster identifier. Each ZDO command has a defined payload. See the "ZigBee Device Profile" section of the ZigBee Specification for details

ZDO Command	Cluster ID
Network Address Request	0x0000
IEEE Address Request	0x0001
Node Descriptor Request	0x0002
Simple Descriptor Request	0x0004
Active Endpoints Request	0x0005
Match Descriptor Request	0x0006
Mgmt LQI Request	0x0031
Mgmt Routing Request	0x0032
Mgmt Leave Request	0x0034
Mgmt Permit Joining Request	0x0036
Mgmt Network Update Request	0x0038

The Explicit Transmit API frame (0x11) is used to send ZigBee Device Objects commands to devices in the network. Sending ZDO commands with the Explicit Transmit API frame requires some formatting of the data payload field.

When sending a ZDO command with the API, all multiple byte values in the ZDO command (API payload) (e.g. u16, u32, 64-bit addresses) must be sent in little endian byte order for the command to be executed correctly on a remote device.

[AO][1][API Output Mode][C;Reserved;Explicit;Reserved;Explicit with ZDO Passthru][Set the API output mode register value. 1 - Received RF data formatted as Explicit Rx-Indicator. 3 - same as one. AO is set by default to 1, which causes received RF data to be formatted as an explicit receive API frame. AO may be set to 3, which causes received ZDO requests to be passed out the UART.

The following table shows how the Explicit API frame can be used to send an "Active Endpoints" request to discover the active endpoints on a device with a 16-bit address of 0x1234.

	Frame Fields			Offset	Example	Description
API Packet	Start Delimiter			0	0x7E	
	Length			MSB 1	0x00	Number of bytes between the length and the checksum
				LSB 2	0x17	
	Frame-specific Data	Frame Type		3	0x11	
		Frame ID		4	0x01	Identifies the UART data frame for the host to correlate with a subsequent transmit status. If set to 0, no transmit status frame will be sent out the UART.
		64-bit Destination Address		MSB 5	0x00	64-bit address of the destination device (big endian byte order). For unicast transmissions, set to the 64-bit address of the destination device, or to 0x0000000000000000 to send a unicast to the coordinator. Set to 0x000000000000FFFF for broadcast.
				6	0x00	
				7	0x00	
				8	0x00	
				9	0x00	
				10	0x00	
				11	0xFF	
				LSB 12	0xFF	
		16-bit Destination Network Address		MSB 13	0xFF	16-bit address of the destination device (big endian byte order). Set to 0xFFFE for broadcast, or if the 16-bit address is unknown.
				LSB 14	0xFE	
		Source Endpoint		15	0x00	Set to 0x00 for ZDO transmissions (endpoint 0 is the ZDO endpoint).
		Destination Endpoint		16	0x00	Set to 0x00 for ZDO transmissions (endpoint 0 is the ZDO endpoint).
		Cluster ID		MSB 17	0x00	Set to the cluster ID that corresponds to the ZDO command being sent. 0x0005 = Active Endpoints Request
				LSB 18	0x05	
		Profile ID		MSB 19	0x00	Set to 0x0000 for ZDO transmissions (Profile ID 0x0000 is the ZigBee Device Profile that supports ZDOs).
				LSB 20	0x00	
		Broadcast Radius		21	0x00	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value.
		Transmit Options		22	0x00	All bits must be set to 0.
		Data Payload	Transaction Sequence Number	23	0x01	The required payload for a ZDO command. All multi-byte ZDO parameter values (u16, u32, 64-bit address) must be sent in little endian byte order. The Active Endpoints Request includes the following payload: [16-bit NwkAddrOfInterest] Note the 16-bit address in the API example (0x1234) is sent in little endian byte order (0x3412).
			ZDO Payload	24	0x34	
				25	0x12	
	Checksum			26	0xA6	0xFF minus the 8 bit sum of bytes from offset 3 to this byte.

Sending ZigBee Cluster Library (ZCL) Commands with the API

The ZigBee Cluster Library defines a set of attributes and commands (clusters) that can be supported in multiple ZigBee profiles. The ZCL commands are typically required when developing a ZigBee product that will interoperate in a public profile such as home automation or Smart Energy, or when communicating with ZigBee devices from other vendors. Applications that are not designed for a public profile or for interoperability applications can skip this section.

The following table shows some prominent clusters with some of their respective attributes and commands.

Cluster (Cluster ID)	Attributes (Attribute ID)	Command ID
Basic (0x0000)	Application Version (0x0001) Hardware Version (0x0003) Model Identifier (0x0005)	Reset to defaults (0x00)
Identify (0x0003)	Identify Time (0x0000)	Identify (0x00) Identify Query (0x01)
Time (0x000A)	Time (0x0000) Time Status (0x0001) Time Zone (0x0002)	
Thermostat (0x0201)	Local Temperature (0x0000) Occupancy (0x0002)	-Setpoint raise / lower (0x00)

The ZCL defines a number of profile-wide commands that can be supported on any profile, also known as general commands. These commands include the following.

Command (Command ID)	Description
Read Attributes (0x00)	Used to read one or more attributes on a remote device.
Read Attributes Response (0x01)	Generated in response to a read attributes command.
Write Attributes (0x02)	Used to change one or more attributes on a remote device.
Write Attributes Response (0x04)	Sent in response to a write attributes command.
Configure Reporting (0x06)	Used to configure a device to automatically report on the values of one or more of its attributes.
Report Attributes (0x0A)	Used to report attributes when report conditions have been satisfied.
Discover Attributes (0x0C)	Used to discover the attribute identifiers on a remote device.
Discover Attributes Response (0x0D)	Sent in response to a discover attributes command.

The Explicit Transmit API frame (0x11) is used to send ZCL commands to devices in the network. Sending ZCL commands with the Explicit Transmit API frame requires some formatting of the data payload field.

When sending a ZCL command with the API, all multiple byte values in the ZCL command (API Payload) (e.g. u16, u32, 64-bit addresses) must be sent in little endian byte order for the command to be executed correctly on a remote device.

Note: When sending ZCL commands, the AO command should be set to 1 to enable the explicit receive API frame. This will provide indication of the source 64- and 16-bit addresses, cluster ID, profile ID, and endpoint information for each received packet. This information is required to properly decode received data.

The following table shows how the Explicit API frame can be used to read the hardware version attribute from a device with a 64-bit address of 0x0013A200 40401234 (unknown 16-bit address). This example uses arbitrary source and destination endpoints. Recall the hardware version attribute (attribute ID 0x0003) is part of the basic cluster (cluster ID 0x0000). The Read Attribute general command ID is 0x00.

	Frame Fields				Offset	Example	Description
API Packet	Start Delimiter				0	0x7E	
	Length				MSB 1	0x00	Number of bytes between the length and the checksum
					LSB 2	0x2E	
	Frame-specific Data	Frame Type			3	0x11	
		Frame ID			4	0x01	Identifies the UART data frame for the host to correlate with a subsequent transmit status. If set to 0, no transmit status frame will be sent out the UART.
		64-bit Destination Address			MSB 5	0x00	64-bit address of the destination device (big endian byte order). For unicast transmissions, set to the 64-bit address of the destination device, or to 0x0000000000000000 to send a unicast to the coordinator. Set to 0x000000000000FFFF for broadcast.
					6	0x13	
					7	0xA2	
					8	0x00	
					9	0x40	
					10	0x40	
					11	0x12	
					MSB 12	0x34	
		16-bit Destination Network Address			13	0xFF	16-bit address of the destination device (big endian byte order). Set to 0xFFFE for broadcast, or if the 16-bit address is unknown.
					LSB 14	0xFE	
		Source Endpoint			15	0x41	Set to the source endpoint on the sending device. (0x41 arbitrarily selected).
		Destination Endpoint			16	0x42	Set to the destination endpoint on the remote device. (0x42 arbitrarily selected)
		Cluster ID			MSB 17	0x00	Set to the cluster ID that corresponds to the ZCL command being sent. 0x0000 = Basic Cluster
					LSB 18	0x00	
		Profile ID			MSB 19	0xD1	Set to the profile ID supported on the device. (0xD123 arbitrarily selected).
					LSB 20	0x23	
		Broadcast Radius			21	0x00	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value.
		Transmit Options			22	0x00	All bits must be set to 0.
		Data Payload	ZCL Frame Header	Frame Control	23	0x00	Bitfield that defines the command type and other relevant information in the ZCL command. See the ZCL specification for details.
				Transaction Sequence Number	24	0x01	A sequence number used to correlate a ZCL command with a ZCL response. (The hardware version response will include this byte as a sequence number in the response.) The value 0x01 was arbitrarily selected.
				Command ID	25	0x00	Since the frame control "frame type" bits are 00, this byte specifies a general command. Command ID 0x00 is a Read Attributes command.
			ZCL Payload	Attribute ID	26	0x03	The payload for a "Read Attributes" command is a list of Attribute Identifiers that are being read. Note the 16-bit Attribute ID (0x0003) is sent in little endian byte order (0x0300). All multi-byte ZCL header and payload values must be sent in little endian byte order.
					27	0x00	
	Checksum				28	0xFA	0xFF minus the 8 bit sum of bytes from offset 3 to this byte.

In the above example, the Frame Control was constructed as follows:

Name	Bits	Example Value Description
Frame Type	0-1	00 - Command acts across the entire profile
Manufacturer Specific	2	0 - The manufacturer code field is omitted from the ZCL Frame Header.
Direction	3	0 - The command is being sent from the client side to the server side.
Disable Default Response	4	0 - Default response not disabled
Reserved	5-7	Set to 0.

See the ZigBee Cluster Library specification for details.

Sending Public Profile Commands with the API

Commands in the Smart Energy Application Profile can be sent with the XBee API using the Explicit Transmit API frame (0x11). Sending public profile commands with the Explicit Transmit API frame requires some formatting of the data payload field. Most of the public profile commands fit into the ZigBee Cluster Library (ZCL) architecture as described in the previous section.

The following table shows how the Explicit API frame can be used to send a demand response and load control message (cluster ID 0x701) in the Smart Energy profile (profile ID 0x0109) in the revision 14 Smart Energy specification. The message will be a "Load Control Event" (command ID 0x00) and will be sent to a device with 64-bit address of 0x0013A200 40401234 with a 16-bit address of 0x5678. The event will start a load control event for water heaters and smart appliances, for a duration of 1 minute, starting immediately.

Note: When sending public profile commands, the AO command should be set to 1 to enable the explicit receive API frame. This will provide indication of the source 64- and 16-bit addresses, cluster ID, profile ID, and endpoint information for each received packet. This information is required to properly decode received data.

	Frame Fields				Offset	Example	Description
API Packet	Start Delimiter				0	0x7E	
	Length				MSB 1	0x00	Number of bytes between the length and the checksum
					LSB 2	0x2E	
	Frame-specific Data	Frame Type			3	0x11	
		Frame ID			4	0x01	Identifies the UART data frame for the host to correlate with a subsequent transmit status. If set to 0, no transmit status frame will be sent out the UART.
		64-bit Destination Address			MSB 5	0x00	64-bit address of the destination device (big endian byte order). For unicast transmissions, set to the 64-bit address of the destination device, or to 0x0000000000000000 to send a unicast to the coordinator. Set to 0x000000000000FFFF for broadcast.
					6	0x13	
					7	0xA2	
					8	0x00	
					9	0x40	
					10	0x40	
					11	0x12	
					12	0x34	
		16-bit Destination Network Address			MSB 13	0x56	16-bit address of the destination device (big endian byte order). Set to 0xFFFE for broadcast, or if the 16-bit address is unknown.
					LSB 14	0x78	
		Source Endpoint			15	0x41	Set to the source endpoint on the sending device. (0x41 arbitrarily selected).
		Destination Endpoint			16	0x42	Set to the destination endpoint on the remote device. (0x42 arbitrarily selected)
		Cluster ID			MSB 17	0x07	Set to the cluster ID that corresponds to the ZCL command being sent. 0x0701 = Demand response and load control cluster ID
					LSB 18	0x01	
		Profile ID			MSB 19	0x01	Set to the profile ID supported on the device. 0x0109 = Smart Energy profile ID.
					LSB 20	0x09	
		Broadcast Radius			21	0x00	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value.
		Transmit Options			22	0x00	All bits must be set to 0.
		Data Payload (continued)	ZCL Frame Header	Frame Control	23	0x09	Bitfield that defines the command type and other relevant information in the ZCL command. See the ZCL specification for details.
				Transaction Sequence Number	24	0x01	A sequence number used to correlate a ZCL command with a ZCL response. (The hardware version response will include this byte as a sequence number in the response.) The value 0x01 was arbitrarily selected.
				Cluster-specific Command ID	25	0x00	Since the frame control "frame type" bits are 01, this byte specifies a cluster-specific command. Command ID 0x00 in the Demand Response and Load Control cluster is a Load Control Event command. (See Smart Energy specification.)

Frame Fields				Offset	Example	Description
Frame-specific Data	**Data Payload** (continued)	**ZCL Payload - Load Control Event Data**	**Issuer Event ID**	26	0x78	4-byte unique identifier. Note the 4-byte ID is sent in little endian byte order (0x78563412). The event ID in this example (0x12345678) was arbitrarily selected.
				27	0x56	
				28	0x34	
				29	0x12	
			Device Class	30	0x14	to apply the load control event. A bit value of 0x0014 enables smart appliances and water heaters. Note the 2-byte bit field value is sent in little endian byte order.
				31	0x00	
			Utility Enrollment Group	32	0x00	Used to identify sub-groups of devices in the device-class. 0x00 addresses all groups.
			Start Time	33	0x00	UTC timestamp representing when the event should start. A value of 0x00000000 indicates "now".
				34	0x00	
				35	0x00	
				36	0x00	
			Duration in Minutes	37	0x01	This 2-byte value must be sent in little endian byte order.
				38	0x00	
			Criticality Level	39	0x04	Indicates the criticality level of the event. In this example, the level is "voluntary".
			Cooling Temperature	40	0xFF	Requested offset to apply to the normal cooling set point. A value of 0xFF indicates the temperature offset value is not used.
			Heating Temperature Offset	41	0xFF	Requested offset to apply to the normal heating set point. A value of 0xFF indicates the temperature offset value is not used.
			Cooling Temperature Set Point	42	0x00	Requested cooling set point in 0.01 degrees Celsius. A value of 0x8000 means the set point field is not used in this event. Note the 0x80000 is sent in little endian byte order.
				43	0x80	
			Heating Temperature Set Point	44	0x00	Requested heating set point in 0.01 degrees Celsius. A value of 0x8000 means the set point field is not used in this event. Note the 0x80000 is sent in little endian byte order.
				45	0x80	
			Average Load Adjustment Percentage	46	0x80	Maximum energy usage limit. A value of 0x80 indicates the field is not used.
			Duty Cycle	47	0xFF	Defines the maximum "On" duty cycle. A value of 0xFF indicates the duty cycle is not used in this event.
			Duty Cycle Event Control	48	0x00	A bitmap describing event options.
Checksum				49	0x5B	0xFF minus the 8 bit sum of bytes from offset 3 to this byte.

In the above example, the Frame Control field (offset 23) was constructed as follows:

Name	Bits	Example Value Description
Frame Type	0-1	01 - Command is specific to a cluster
Manufacturer Specific	2	0 - The manufacturer code field is omitted from the ZCL Frame Header.
Direction	3	1 - The command is being sent from the server side to the client side.
Disable Default Response	4	0 - Default response not disabled
Reserved	5-7	Set to 0.

9. XBee Command Reference Tables

Addressing

Addressing Commands)

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
MY	**16-bit Network Address**. Read the 16-bit network address of the module. A value of 0xFFFE means the module has not joined a ZigBee network	CRE	0 - 0xFFFE [read-only]	0xFFFE
MP	**16-bit Parent Network Address**. Read the 16-bit network address of the module's parent. A value of 0xFFFE means the module does not have a parent.	E	0 - 0xFFFE [read-only]	0xFFFE
NC	**Number of Remaining Children**. Read the number of end device children that can join the device. If NC returns 0, then the device cannot allow any more end device children to join.	CR	0 - MAX_CHILDREN (maximum varies)	[read-only]
SH	**Serial Number High**. Read the high 32 bits of the module's unique 64-bit address.	CRE	0 - 0xFFFFFFFF [read-only]	factory-set
SL	**Serial Number Low**. Read the low 32 bits of the module's unique 64-bit address.	CRE	0 - 0xFFFFFFFF [read-only]	factory-set
NP	**Maximum RF Payload Bytes**. This value returns the maximum number of RF payload bytes that can be sent in a unicast transmission. If APS encryption is used (API transmit option bit enabled), the maximum payload size is reduced by 9 bytes. If source routing is used (AR < 0xFF), the maximum payload size is reduced further. **Note**: NP returns a hexadecimal value. (e.g. if NP returns 0x54, this is equivalent to 84 bytes)	CRE	0x80	[read-only]

Node types that support the command: C=Coordinator, R=Router, E=End Device

Networking

Networking Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
CH	**Operating Channel**. Read the channel number used for transmitting and receiving between RF modules. Uses 802.15.4 channel numbers. A value of 0 means the device has not joined a PAN and is not operating on any channel.	CRE	**XBee** 0, 0x0B - 0x1A **XBee-PRO (S2)** 0, 0x0B - 0x18 **XBee-PRO (S2B)** 0, 0x0B - 0x19	[read-only]
CB	**Commissioning Button.** Simulate commissioning pushbutton presses. Actions for one, two and four presses are defined. 1 - Wake for one minute. If not joined, attempt to join a network. If joining attempt fails, blink numeric error code on associate pin. 2 - If joined, broadcast a "permit joining" message to the network. Joining will be enabled for NJ time. 4 - If joined, leave the network and reset to default parameter values. (Exception: the Smart Energy Range Extender by design will not reset its values.) Attempt to join using settings. **Note:** CB1 is disabled while a join is in progress. If ID, ZS, or SC is changed while a form, scan, join, or authentication is in progress, the node will leave the network and try to form/join again.	CRE	1, 2, 4	-
ID	**Extended PAN ID**. Set/read the 64-bit extended PAN ID. If set to 0, the coordinator will select a random extended PAN ID, and the router / end device will join any extended PAN ID. Changes to ID should be written to non-volatile memory using the WR command to preserve the ID setting if a power cycle occurs.	CRE	0 - 0xFFFFFFFFFFFFFFFF	0
OP	**Operating Extended PAN ID**. Read the 64-bit extended PAN ID. The OP value reflects the operating extended PAN ID that the module is running on. If ID > 0, OP will equal ID.	CRE	0x01 - 0xFFFFFFFFFFFFFFFF	[read-only]
NH	**Maximum Unicast Hops**. Set / read the maximum hops limit. This limit sets the maximum broadcast hops value (BH) and determines the unicast timeout. The timeout is computed as (50 * NH) + 100 ms. The default unicast timeout of 1.6 seconds (NH=0x1E) is enough time for data and the acknowledgment to traverse about 8 hops.	CRE	0 - 0x0B	0x0B
BH	**Broadcast Hops.** Set/Read the maximum number of hops for each broadcast data transmission. Setting this to 0 will use the maximum number of hops.	CRE	0 - 0x1E	0
OI	**Operating 16-bit PAN ID**. Read the 16-bit PAN ID. The OI value reflects the actual 16-bit PAN ID the module is running on. .	CRE	0 - 0xFFFF	[read-only]
SC	**Scan Channels**. Set/Read the list of channels to scan. ***Coordinator*** - Bit field list of channels to choose from prior to starting network. ***Router/End Device*** - Bit field list of channels that will be scanned to find a Coordinator/ Router to join. Changes to SC should be written using WR command to preserve the SC setting if a power cycle occurs. Bit (Channel): 0 (0x0B) 4 (0x0F) 8 (0x13) 12 (0x17) 1 (0x0C) 5 (0x10) 9 (0x14) 13 (0x18) 2 (0x0D) 6 (0x11) 10 (0x15) 14 (0x19) 3 (0x0E) 7 (0x12) 11 (0x16) 15 (0x1A)	CRE	**XBee** 1 - 0xFFFF [bitfield] **XBee-PRO (S2)** 1 - 0x3FFF [bitfield] (bits 14, 15 not allowed) **XBee-PRO (S2B)** **Prior to 3x22:** 1 - 0x7FFF (bitfield) **After 3x22:** 1 - 0xFFFF (bitfield) **Note:** Enabling channel 26 will limit transmit power to 3 dBm on all channels.	**Prior to 3x22:** 0x1FFE **After 3x22:** **XBee** 0xFFFF **XBee-PRO (S2)** 0x3FFF **XBee-PRO (S2B)** 0x7FFF
SD	**Scan Duration**. Set/Read the scan duration exponent. Changes to SD should be written using WR command. ***Coordinator*** - Duration of the Active and Energy Scans (on each channel) that are used to determine an acceptable channel and Pan ID for the Coordinator to startup on. ***Router / End Device*** - Duration of Active Scan (on each channel) used to locate an available Coordinator / Router to join during Association. Scan Time is measured as:(# Channels to Scan) * (2 ^ SD) * 15.36ms - The number of channels to scan is determined by the SC parameter. The XBee can scan up to 16 channels (SC = 0xFFFF). Sample Scan Duration times (13 channel scan): If SD = 0, time = 0.200 sec SD = 2, time = 0.799 sec SD = 4, time = 3.190 sec SD = 6, time = 12.780 sec **Note**: SD influences the time the MAC listens for beacons or runs an energy scan on a given channel. The SD time is not a good estimate of the router/end device joining time requirements. ZigBee joining adds additional overhead including beacon processing on each channel, sending a join request, etc. that extend the actual joining time.	CRE	0 - 7 [exponent]	3
ZS	**ZigBee Stack Profile**. Read the ZigBee stack profile value. This must be set the same on all devices that should join the same network.	CRE	2	2 [read-only]

Networking Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
NJ	**Node Join Time**. Set/Read the time that a Coordinator/Router allows nodes to join. This value can be changed at run time without requiring a Coordinator or Router to restart. The time starts once the Coordinator or Router has started. The timer is reset when NJ changes.	CR	0 - 0xFF [x 1 sec] 0x00 disables permit join Starting with 3x26 0xFF sets permit join to always enabled	0xFE (maximum of 254 s)
AR	**Aggregate Routing Notification**. Set/read time between consecutive aggregate route broadcast messages. If used, AR should be set on only one device to enable many-to-one routing to the device. Setting AR to 0 sends one broadcast. Setting AR to FF disables the aggregate route broadcast.	CR	0 - 0xFF [seconds]	0xFF

Security

Security Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
EO	**Encryption Options.** Configure options for encryption. Coordinator is read-only, with a value of 2. Router and End Device is read-write and defaults to a value of 0. If set to 8, then Authentication (key establishment) is enabled.	CRE	0 - 0xFF	C: 2 RE: 0 prior to 3x22 RE: 8 as of 3x22
NK	**Network Encryption Key**. Set the 128-bit AES network encryption key. This command is write-only; NK cannot be read. If set to 0 (default), the module will select a random network key. As of 3x28, the KY value will be retained in spite of an RE command.	C	128-bit value	1 prior to rev 3x22 0 as of rev 3x22
KY	**Link Key**. Set the 128-bit AES link key. This command is write only; KY cannot be read. As of 3x28, the KY value will be retained in spite of an RE command.	CRE	128-bit value	C: 0 as of 3x22 C: 1 prior to 3x22 RE: 1
VC	**Verify Certificate**. Verify the presence of the certificate. This command is read only.		CRE 1 – 0x00: no certificate present; 0x01: certificate present	[read-only]
ZT	**Install Device Implicit Cert**. Set the 48-byte Device Implicit Certificate key. This command is write-only; ZT cannot be read. If set to zero (default), the key is disabled.	CRE	48-byte value	0
ZU	**Install Public Key.** Set the 22-byte CA Public key. This command is read-write; ZU may be read. If set to zero (default), the key is disabled.	CRE	22-byte value	0
ZV	**Install Private Key.** Set the 21-byte Device Private Key. This command is write-only; ZV cannot be read. If set to zero (default), the key is disabled.	CRE	21-byte value	0

RF Interfacing

RF Interfacing Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
PL	**Power Level**. Select/Read the power level at which the RF module transmits conducted power. For XBee-PRO (S2B), Power Level 4 is calibrated and the other power levels are approximate.For XBee (S2), only the default power level (PL=4) is guaranteed from -40 to 85° C..	CRE	**XBee (S2)** (boost mode disabled) 0 = -8dBm 1 = -4dBm 2 = -2dBm 3 = 0dBm 4 = +2dBm **XBee-PRO (S2)** 4 = 17 dBm **XBee-PRO (S2) (International Variant)** 4 = 10dBm **XBee-PRO (S2B)** (boost mode enabled) 0 = 10dBm 1 = 12dBm 2 = 14dBm 3 = 16dBm 4 = 18dBm **XBee-PRO (S2B) (International Variant)** (boost mode enabled) 0 = 2dBm 1 = 4dBm 2 = 6dBm 3 = 8dBm 4 = 10dBm	4
PM	**Power Mode**. Set/read the power mode of the device. Enabling boost mode will improve the receive sensitivity by 1dB and increase the transmit power by 2dB Note: Enabling boost mode on the XBee-PRO (S2) will not affect the output power. Boost mode imposes a slight increase in current draw. See section 1.2 for details.	CRE	0-1, 0= -Boost mode disabled, 1= Boost mode enabled.	1
DB	**Received Signal Strength**. This command reports the received signal strength of the last received RF data packet. The DB command only indicates the signal strength of the last hop. It does not provide an accurate quality measurement for a multi-hop link. DB can be set to 0 to clear it. The DB command value is measured in -dBm. For example if DB returns 0x50, then the RSSI of the last packet received was -80dBm.	CRE	0 - 0xFF Observed range for **XBee-PRO**: 0x1A - 0x58 **XBee**: 0x 1A - 0x5C	

1. Node types that support the command: C = Coordinator, R = Router, E = End Device

Serial Interfacing (I/O)

Serial Interfacing Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
AP	**API Enable**. Enable API Mode. The AP command is only supported when using API firmware: 31xx (API Coordinator), 33xx (API Router), 39xx (API End Device).	CRE	1 - 2 1 = API-enabled 2 = API-enabled (w/escaped control characters)	1
AO	**API Options**. Configure options for API. Current options select the type of receive API frame to send out the UART for received RF data packets. CRE 1 - Default 1 - Explicit Rx data indicator API frame enabled (0x91). ZDO Passthru is disabled. 3 - Explicit Rx data indicator API frame enabled (0x91). ZDO Passthru is enabled. ZDO requests which are not supported by the stack, and the ZDO requests Simple_Desc_req, Match_Desc_req, and Active_EP_req are passed out the UART port to the external processor in a 0x91 API frame. The external processor is responsible for their processing and response generation. If you enable option 3, the external processor needs to do the following: 1) respond to Simple_Desc_req; 2) respond to Match_Desc_req; 3) respond to Active_EP_req; 4) respond to other ZDO requests which are not supported by the stack. For example, remote devices which are attempting to Authenticate after joining will send a Match_Desc_req in an attempt to discover the endpoint which supports the Key Establishment Cluster in the Smart Energy Profile, which usually resides on the Coordinator (Energy Service Portal or Meter Device). 7 - Explicit Rx data indicator API frame enabled (0x91). ZDO Passthru of supported and unsupported ZDO requests are passed out the UART port to the external processor in a 0x91 API frame. For example, binding requests will be passed through.	CRE	1 - Explicit Rx data indicator API frame enabled (0x91) 3 - Explicit Rx data indicator frame enabled (0x91) and ZDO passthru enabled. 7 - Explicit Rx data indicator API frame enabled (0x91), supported and unsupported ZDO passthru is enabled.	1
BD	**Interface Data Rate**. Set/Read the serial interface data rate for communication between the module serial port and host. Any value above 0x07 will be interpreted as an actual baud rate. When a value above 0x07 is sent, the closest interface data rate represented by the number is stored in the BD register.	CRE	0x80 - 0xE1000 (non-standard rates up to 921kbps)	3
NB	**Serial Parity**. Set/Read the serial parity setting on the module.	CRE	0 = No parity 1 = Even parity 2 = Odd parity 3 = Mark parity	0
LT	**Set/read the Associate LED blink rate**. This value determines the blink rate of the Associate/DIO5 pin if D5=1 and the module has started a network. Setting LT to 0 will use the default blink time (500ms).	CRE	0X0A-0XFF X10 MS	0

1. Node types that support the command: C = Coordinator, R = Router, E = End Device

Diagnostics

Diagnostics Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
VR	**Firmware Version**. Read firmware version of the module. The firmware version returns 4 hexadecimal values (2 bytes) "ABCD". Digits ABC are the main release number and D is the revision number from the main release. "B" is a variant designator. XBee and XBee-PRO SE modules return: 0x3xxx versions. XBee and XBee-PRO ZB modules return: 0x2xxx versions. XBee and XBee-PRO ZNet modules return: 0x1xxx versions. ZNet firmware is not compatible with ZB firmware.	CRE	0 - 0xFFFF [read-only]	Factory-set
HV	**Hardware Version**. Read the hardware version of the module.version of the module. This command can be used to distinguish among different hardware platforms. The upper byte returns a value that is unique to each module type. The lower byte indicates the hardware revision. XBee SE and XBee SE modules return the following (hexadecimal) values: 0x19xx - XBee module 0x1Axx - XBee-PRO (S2) module 0x1Exx - XBee-PRO (S2B) module	CRE	0 - 0xFFFF [read-only]	Factory-set
CK	**Configuration Checksum**. Returns the checksum of the configuration registers.	CRE	0 – 0xFF	[read-only]
AI	**Association Indication**. Read information regarding last node join request: 0x00 - Successfully formed or joined a network. (Coordinators form a network, routers and end devices join a network.) 0x21 - Scan found no PANs 0x22 - Scan found no valid PANs based on current SC and ID settings 0x23 - Valid Coordinator or Routers found, but they are not allowing joining (NJ expired) 0x24 - No joinable beacons were found 0x25 - Unexpected state, node should not be attempting to join at this time 0x27 - Node Joining attempt failed (typically due to incompatible security settings) 0x2A - Coordinator Start attempt failed' 0x2B - Checking for an existing coordinator 0x2C - Attempt to leave the network failed 0x30 – Discovering key establishment endpoint 0x31 – Key establishment endpoint discovery failed 0x32 – Initiate key establishment response not received 0x33 – Ephemeral data response not received 0x34 - Confirm key response not received 0x36 – Received terminate request 0x3A - Key establishment transmission failed 0x3B – Invalid certificate 0x3C – Key establishment not allowed 0xAB - Attempted to join a device that did not respond. 0xAC - Secure join error - network security key received unsecured 0xAD - Secure join error - network security key not received 0xAF - Secure join error - joining device does not have the right preconfigured link key 0xFE - Stack initialization failure 0xFF - Scanning for a ZigBee network (routers and end devices) **Note**: New non-zero AI values may be added in later firmware versions. Applications should read AI until it returns 0x00, indicating a successful startup (coordinator) or join (routers and end devices)	CRE	0 - 0xFF [read-only]	--
TP	**Temperature Indication.** Read power compensation temperature sensor in units of degrees Celsius. **XBee** - Not supported **XBee-PRO (S2)** - Not supported **XBee-PRO (S2B)** - 0-0xFFFF	CRE	11001001 = -55 deg. C 11001110 = -50 deg. C 11100111 = -25 deg. C 00000000 = 0 deg. C 00001010 = 10 deg. C 00011001 = 25 deg. C 00110010 = 50 deg. C 01001011 = 75 deg. C 01100100 = 100 deg. C 01111101 = 125 deg. C [read-only]	-

Diagnostics Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
DO	**Device Options.** **Bit 0x01** enables Temperature Compensation on the XBee S2B. Disabling it shortens the overhead time for an End Device to wake from sleep from 13 ms to 2 ms, but transmit power may vary by as much as 8 dBm as a function of temperature. **Bit 0x02** enables the join notification message option on the SE coordinator. By default, it is disabled. This option appears in the SE coordinator builds. **Bit 0x04** enables best response during join, when disabled enables first response during join. By default, it is disabled. This option appears in the SE router and end device builds.	CRE	0 - 0xFF	0x01
%V	**Supply Voltage**. Reads the voltage level on the VCC pin. Scale the read value by 1200/1024 to get a reading in mV. For example, a %V reading of 0x900 (2304 decimal) represents 2700mV or 2.7V.	CRE	0 - 0xFFFF [read-only]	-

1. Node types that support the command:C = Coordinator, R = Router, E = End Device

Sleep Commands

Sleep Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
SM	**Sleep Mode** Sets the sleep mode on the RF module.	RE	1-Pin sleep enabled 4-Cyclic sleep enabled 5 - Cyclic sleep, pin wake	-
SN	**Number of Sleep Periods.** Sets the number of sleep periods to not assert the On/Sleep pin on wakeup if no RF data is waiting for the end device. This command allows a host application to sleep for an extended time if no RF data is present. It should be set at least equal to the longest SN of any child end device.	CRE	1 - 0xFFFF	1
SP	**Sleep Period.** This value determines how long the end device will sleep at a time, up to 28 seconds. (The sleep time can effectively be extended past 28 seconds using the SN command.) On the parent, this value determines how long the parent will buffer a message for the sleeping end device. It should be set at least equal to the longest SP time of any child end device.	CRE	0x20 - 0xAF0 x 10ms (Quarter second resolution)	0x02EE
ST	**Time Before Sleep** Sets the time before sleep timer on an end device.The timer is reset each time serial or RF data is received. Once the timer expires, an end device may enter low power operation. Applicable for cyclic sleep end devices only.	E	1 - 0xFFFE (x 1ms)	0x1388 (5 seconds)
SO Command	**Sleep Options**. Configure options for sleep. Unused option bits should be set to 0. Sleep options include: 0x02 - Always wake for ST time 0x04 - Sleep entire SN * SP time Sleep options should not be used for most applications.	E	0 - 0xFF	0
PO	**Polling Rate**. Sets the polling rate for the end device.	E	0-0x1770 [10 msec]	0

Execution Commands

Where most AT commands set or query register values, execution commands cause an action to be executed on the module. Execution commands are executed immediately and do not require changes to be applied.

Execution Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
AC	**Apply Changes**. Applies changes to all command registers causing queued command register values to be applied. For example, changing the serial interface rate with the BD command will not change the UART interface rate until changes are applied with the AC command. The CN command and 0x08 API command frame also apply changes.	CRE	-	-
SI	**Sleep Immediately**. Cause a cyclic sleep module to sleep immediately rather than wait for the ST timer to expire.	E	-	-
WR	**Write**. Write parameter values to non-volatile memory so that parameter modifications persist through subsequent resets.	CRE	-	-
RE	**Restore Defaults**. Restore module parameters to factory defaults. With 3x28, KY, ZT, ZU, and ZV will retain their values in spite of an RE command.	CRE	-	-
FR	**Software Reset.** Reset module. Responds immediately with an OK status, and then performs a software reset about 2 seconds later.	CRE	-	-
NR	**Network Reset**. Force a node to disassociate from the network. When applied, the link key table will be erased. A coordinator will require preconfigured link keys to be re-registered.	CRE	0 (optional)	-

Node types that support the command: C = Coordinator, R = Router, E = End Device

10. RF Module Support

This chapter provides customization information for the XBee/XBee-PRO SE modules. In addition to providing an extremely flexible and powerful API, the XBee and XBee-PRO SE modules are a robust development platform that have passed FCC and ETSI testing. Developers can customize default parameters, or even write or load custom firmware for Ember's EM250 chip.

X-CTU Configuration Tool

Digi provides a Windows X-CTU configuration tool for configuring module parameters and updating firmware. The XCTU has the capability to do the following:

- Update firmware on a local module (requires USB or serial connection)
- Read or write module configuration parameters on a local or remote device
- Save and load configuration profiles containing customized settings.

Contact Digi support for more information about the X-CTU.

XBee Bootloader

XBee modules use a modified version of Ember's bootloader. This bootloader version supports a custom entry mechanism that uses module pins DIN (pin 3), $\overline{\text{DTR}}$ / SLEEP_RQ (pin 9), and $\overline{\text{RTS}}$ (pin 16). To invoke the boot loader, do the following:

1. Set $\overline{\text{DTR}}$ / SLEEP_RQ low (TTL 0V) and RTS high.
2. Send a serial break to the DIN pin and power cycle or reset the module.
3. When the module powers up, $\overline{\text{DTR}}$ / SLEEP_RQ and DIN should be low (TTL 0V) and RTS should be high.
4. Terminate the serial break and send a carriage return at 115200bps to the module.
5. If successful, the module will send the Ember boot loader menu out the DOUT pin at 115200bps.
6. Commands can be sent to the boot loader at 115200bps.

Note: Hardware flow control should be disabled when entering and communicating with the EM250 bootloader.

Programming XBee Modules

Firmware on the XBee and XBee-PRO SE modules can be updated through one of two means:

- Serially
- SIF header.

Each method is described below.

Where possible, configuration settings are retained despite firmware changes. This can lead to difficulties, especially when changing firmware among variants of the same release with different default config settings. For example, the default setting of a SE Coordinator's EO register is 2, while a Router's default EO setting is 8. Retaining the EO setting after changing a firmware revision from Router to Coordinator (or the reverse) will disable the trust center functions (or authentication functions in the reverse case). Best practice is to do a RE (reset to factory defaults), set the configuration registers explicitly, then WR (write config settings).

Serial Firmware Updates

Serial firmware updates make use of the XBee custom bootloader which ships in all units. This modified bootloader is based on Ember's standalone bootloader, but with a modified entry mechanism. The modified entry mechanism uses module pins 3, 9, and 16 (DIN, DTR, and RTS respectively).

The X-CTU program can update firmware serially on the XBee and XBee-PRO SE modules. Contact Digi support for details.

If an application requires custom firmware to update the XBee firmware serially, the following steps are required.

Invoke XBee Bootloader

See the "XBee Bootloader" section above for steps to invoke the bootloader.

Send Firmware Image

After invoking the bootloader, the Ember bootloader will send the bootloader menu characters out the UART at 115200 bps. The application should do the following to upload a firmware image.

1. Look for the bootloader prompt "BL >" to ensure the bootloader is active

2. Send an ASCII "1" character to initiate a firmware update

3. After sending a "1", the EM250 waits for an XModem CRC upload of an .ebl image over the serial line at 115200 bps. The .ebl file must be sent to the EM250 in order.

If the upload is interrupted with a power cycle or reset event, the EM250 will detect an invalid application image and enter bootloader mode. The entire ebl image should be uploaded again to recover. If an error occurs while uploading, the EM250 bootloader returns an error code from the following table:

Hex Error Code	Description
0x21	The bootloader encountered an error while trying to parse the Start of Header (SOH) character in the XModem frame.
0x22	The bootloader detected an invalid checksum in the XModem frame.
0x23	The bootloader encountered an error while trying to parse the high byte of the CRC in the XModem frame.
0x24	The bootloader encountered an error while trying to parse the low byte of the CRC in the XModem frame.
0x25	The bootloader encountered an error in the sequence number of the current XModem frame.
0x26	The frame that the bootloader was trying to parse was deemed incomplete (some bytes missing or lost).
0x27	The bootloader encountered a duplicate of the previous XModem frame.
0x41	No .ebl header was received when expected.
0x42	Header failed CRC.
0x43	File failed CRC.
0x44	Unknown tag detected in .ebl image.
0x45	Invalid .ebl header signature.
0x46	Trying to flash odd number of bytes.
0x47	Indexed past end of block buffer.
0x48	Attempt to overwrite bootloader flash.
0x49	Attempt to overwrite SIMEE flash.
0x4A	Flash erase failed.
0x4B	Flash write failed.
0x4C	End tag CRC wrong length.
0x4D	Received data before query request/response

SIF Firmware Updates

The XBee/XBee-PRO modules have a 2x5 SIF header that can be used with Ember's InSight tools to upload firmware onto the modules. These tools include a USB device (USBLink) and Ethernet-enabled InSight Adapters. Contact Ember for details.

Warning: If programming firmware through the SIF interface, be aware that uploading firmware through the SIF header can potentially erase the XBee bootloader. If this happens, serial firmware updates will not work.

(The pinout for the SIF headers are shown in chapter 1.)

Writing Custom Firmware

The XBee/XBee-PRO module can be used as a hardware development platform for the EM250. Custom firmware images can be developed around the EmberZNet 2.5.x and 3.x mesh stacks (for the EM250) and uploaded to the XBee.

Warning: If programming firmware through the SIF interface, be aware that uploading firmware through the SIF header can potentially erase the XBee bootloader. If this happens, serial firmware updates will not work.

Regulatory Compliance

XBee modules are FCC and ETSI certified for operation on all 16 channels. The EM250 output power can be configured up to 3dBm with boost mode enabled.

XBee-PRO (S2) modules are certified for operation on 14 of the 16 band channels (channels 11 - 24). The scan channels mask of XBee-PRO (S2) devices must be set in the application to disable the upper two channels (e.g. 0x3FFF).

XBee-PRO (S2B) modules are certified for operation on 15 of the 16 band channels (channels 11 - 25). The scan channels mask of XBee-PRO (S2B) devices must be set in the application to disable the highest channel (e.g. 0x7FFF).

The XBee-PRO contains power compensation circuitry to adjust the output power near 18dBm or 10dBm depending on the part number. For best results, the EM250 should be configured with an output power level of 0dBm (or -2dBm if boost mode is enabled). The end product is responsible to adhere to these requirements.

Enabling GPIO 1 and 2

Most of the remaining sections in this chapter describe how to configure GPIO 1 and 2 to function correctly in custom applications that run on the XBee and XBee-PRO modules. In order for GPIO pins 1 and 2 to be configurable, the application must set the GPIO_CFG register to enable GPIO 1 and 2. Bits 4 - 7 in the GPIO_CFG register control the functionality of various GPIO lines. The following table lists values for these bits that enable GPIO 1 and 2. Other functionality is affected by these settings. See the EM250 datasheet from Ember for a complete listing of functionality.

GPIO_CFG[7:4]Enabled Functionality	Enabled Functionality
0000	GPIO 0, 1, 2, 3, 9, 10, 11, 12
0111	0111GPIO 0, 1, 2, 3, 12
1010	GPIO 0, 1, 2, 3
1101	GPIO 0, 1, 2, 3, 11, 12

Example 1

The following code enables GPIO 0, 1, 2, 3, 9, 10, 11, and 12 and maintains all other GPIO_CFG bits.

```
int16u x;
```

```
x = GPIO_CFG;
x &= (0xFF0F);// Clear bits 4 - 7
GPIO_CFG = x;
```

Example 2

The following code enables GPIO 0, 1, 2, 3, and 12 and maintains all other GPIO_CFG bits.

```
int16u x;
x = GPIO_CFG;
x &= (0xFF0F);// Clear bits 4 - 7
x |= 0x0070;// Set bits 4 - 7 to 0111 as shown in the table above.
GPIO_CFG = x;
```

Detecting XBee vs. XBee-PRO

For some applications, it may be necessary to determine if the code is running on an XBee or an XBee-PRO device. The GPIO1 pin on the EM250 is used to identify the module type (see table 1-03 in chapter 1). GPIO1 is connected to ground on the XBee module. The following code could be used to determine if a module is an XBee or XBee-PRO:

```
GPIO_DIRCLRL = GPIO(1);// Set GPIO1 as an input
GPIO_PUL |= GPIO(1);// Enable GPIO1 pullup resistor
ModuleIsXBeePro = (GPIO_INL & GPIO(1));//ModuleIsXBeePro > 0 if XBee-PRO, =0 if non-PRO.
```

Ensuring Optimal Output Power

XBee modules manufactured before February 2008 had an incorrect configuration setting that caused the default output power mode to be set incorrectly. Digi's SE, ZB, and ZNet firmware compensate for this by setting the output power mode in the application firmware.

Custom applications should call the emberSetTxPowerMode() function to set the output power mode as shown below:

XBee Applications

```
emberSetTxPowerMode(EMBER_TX_POWER_MODE_DEFAULT); or
emberSetTxPowerMode(EMBER_TX_POWER_MODE_BOOST);
```

XBee-PRO Applications:

```
emberSetTxPowerMode(EMBER_TX_POWER_MODE_ALTERNATE); or
emberSetTxPowerMode(EMBER_TX_POWER_MODE_BOOST_AND_ALTERNATE);
```

XBee-PRO modules must also set a couple of IO lines to enable output power compensation. This is shown below. Once the IO lines are initialized (after powerup), the XBee will enable the power amplifier and LNA as needed.

On Powerup:

```
   /* GPIO 2 should be set low for at least 10 milliseconds when coming up from power cycle. */
   GPIO_DIRSETL = GPIO(2);// Set GPIO 2 as an output
   GPIO_CLRL = GPIO(2);// Drive GPIO 2 low


   /* After at least 10ms, GPIO 2 should be set high to power the output power compensation
circuitry.
   At the same time GPIO 1 should be configured as an output and set low to enable the output
power compensation circuitry. */
   GPIO_DIRSETL = GPIO(1) | GPIO(2);// Set GPIO 1,2 as outputs
```

```
GPIO_CLRL = GPIO(1);// Drive GPIO 1 low
GPIO_SETL = GPIO(2);// Drive GPIO 2 high
```

Improving Low Power Current Consumption

To improve low power current consumption, the XBee should set a couple of unused IO lines as output low. This can be done during application initialization as shown below.

XBee (non-PRO) Initialization:

```
/* GPIO 1 and 2 are not used in the XBee (non-PRO) and should be set as outputs and driven low to
reduce current draw. */
GPIO_DIRSETL = GPIO(1) | GPIO(2);// Set GPIO 1,2 as outputs
GPIO_CLRL = GPIO(1) | GPIO(2);// Set GPIO 1,2 low
```

XBee-PRO modules should disable the power compensation circuitry when sleeping to reduce current draw. This is shown below.

When sleeping (end devices):

```
/* The power compensation shutdown line on XBee-PRO modules (GPIO 1) should be set high when entering sleep to reduce current consumption.  */
GPIO_SETL = GPIO(1);
```

When waking from sleep (end devices):

```
/* The power compensation shutdown line on XBee-PRO (GPIO 1) should be set low to enable the
power compensation circuitry and LNA. */
GPIO_CLRL = GPIO(1);
```

Appendix A: Definitions

Definitions

Terms and Definitions

ZigBee Node Types

Coordinator	A node that has the unique function of forming a network. The coordinator is responsible for establishing the operating channel and PAN ID for an entire network. Once established, the coordinator can form a network by allowing routers and end devices to join to it. Once the network is formed, the coordinator functions like a router (it can participate in routing packets and be a source or destination for data packets). -- One coordinator per PAN -- Establishes/Organizes PAN -- Can route data packets to/from other nodes -- Can be a data packet source and destination -- Mains-powered Refer to the XBee coordinator section for more information.
Router	A node that creates/maintains network information and uses this information to determine the best route for a data packet. A router must join a network before it can allow other routers and end devices to join to it. A router can participate in routing packets and is intended to be a mains-powered node. -- Several routers can operate in one PAN -- Can route data packets to/from other nodes -- Can be a data packet source and destination -- Mains-powered Refer to the XBee router section for more information.
End device	End devices must always interact with their parent to receive or transmit data. (See 'joining definition.) They are intended to sleep periodically and therefore have no routing capacity. An end device can be a source or destination for data packets but cannot route packets. End devices can be battery-powered and offer low-power operation. -- Several end devices can operate in one PAN -- Can be a data packet source and destination -- All messages are relayed through a coordinator or router -- Lower power modes

ZigBee Protocol

PAN	Personal Area Network - A data communication network that includes a coordinator and one or more routers/end devices.

Terms and Definitions

Joining	The process of a node becoming part of a ZigBee PAN. A node becomes part of a network by joining to a coordinator or a router (that has previously joined to the network). During the process of joining, the node that allowed joining (the parent) assigns a 16-bit address to the joining node (the child).
Network Address	The 16-bit address assigned to a node after it has joined to another node. The coordinator always has a network address of 0.
Operating Channel	The frequency selected for data communications between nodes. The operating channel is selected by the coordinator on power-up.
Energy Scan	A scan of RF channels that detects the amount of energy present on the selected channels. The coordinator uses the energy scan to determine the operating channel.
Route Request	Broadcast transmission sent by a coordinator or router throughout the network in attempt to establish a route to a destination node.
Route Reply	Unicast transmission sent back to the originator of the route request. It is initiated by a node when it receives a route request packet and its address matches the Destination Address in the route request packet.
Route Discovery	The process of establishing a route to a destination node when one does not exist in the Routing Table. It is based on the AODV (Ad-hoc On-demand Distance Vector routing) protocol.
ZigBee Stack	ZigBee is a published specification set of high-level communication protocols for use with small, low-power modules. The ZigBee stack provides a layer of network functionality on top of the 802.15.4 specification. For example, the mesh and routing capabilities available to ZigBee solutions are absent in the 802.15.4 protocol.

Appendix B: Agency Certifications

United States FCC

The XBee Module complies 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. [Figure A-01]

2.XBee Module may only be used with antennas that have been tested and approved for use with this module [refer to the antenna tables in this section].

OEM Labeling Requirements

WARNING: The Original Equipment Manufacturer (OEM) must ensure that FCC labeling requirements are met. This includes a clearly visible label on the outside of the final product enclosure that displays the contents shown in the figure below.

Required FCC Label for OEM products containing the XBee RF Module

> Contains FCC ID: OUR-XBEE2*
>
> The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (*i.*) this device may not cause harmful interference and (*ii.*) this device must accept any interference received, including interference that may cause undesired operation.

Required FCC Label for OEM products containing the XBee PRO (S2) RF Module

> Contains FCC ID:MCQ-XBEEPRO2*
>
> The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (*i.*) this device may not cause harmful interference and (*ii.*) this device must accept any interference received, including interference that may cause undesired operation.

Required FCC Label for OEM products containing the XBee PRO (S2B) RF Module

> Contains FCC ID:MCQ-XBEEPROS2B*
>
> The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (*i.*) this device may not cause harmful interference and (*ii.*) this device must accept any interference received, including interference that may cause undesired operation.

FCC Notices

IMPORTANT: The XBee and XBee PRO Module 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 Module can be installed utilizing antennas and cables constructed with standard connectors (Type-N, SMA, TNC, etc.) if the installation is performed professionally and according to FCC guidelines. For installations not performed by a professional, non-standard connectors (RPSMA, RPTNC, etc.) must be used.

The modules are FCC approved for fixed base station and mobile applications on channels 0x0B-0x1A for Xbee SE and on channels 0x0B - 0x18 for Xbee-PRO SE. If the antenna is mounted at least 20cm (8 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).

XBee Modules: XBee Modules have been tested and approved for use with all the antennas listed in the tables below. (Cable-loss IS required when using gain antennas as shown below.)

The antennas in the tables below have been approved for use with this module. Digi does not carry all of these antenna variants. Contact Digi Sales for available antennas.

Antennas approved for use with the XBee/XBee-PRO SE (Smart Energy) RF Modules (Cable loss is not required.)

OMNI-DIRECTIONAL ANTENNAS for All Available Channels					
Part Number	Type (Description)	Gain	Application*	Minimum Separation Required	Minimum Cable Loss/ Power Reduction/ Attenuation Required
A24-HASM-450	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1 dBi	Fixed/Mobile	20 cm	N/A
A24-HABSM	Dipole (Articulated RPSMA)	2.1 dBi	Fixed	20 cm	N/A
A24-HABUF-P5I	Dipole (Half-wave articulated bulkhead mount U.FL. w/ 5” pigtail)	2.1 dBi	Fixed	20 cm	N/A
A24-HASM-525	Dipole (Half-wave articulated RPSMA - 5.25")	2.1 dBi	Fixed/Mobile	20 cm	N/A
A24-QI	Monopole (Integrated whip)	1.5 dBi	Fixed	20 cm	N/A
29000294	Integral PCB antenna (S2B only)	-0.5 dBi	Fixed/Mobile	20 cm	N/A
29000095	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1 dBi	Fixed/Mobile	20 cm	N/A
29000430	Integral PCB antenna (S2 only)	-0.5 dBi	Fixed/Mobile	20 cm	N/A

Antennas approved for use with the XBee RF Module

YAGI CLASS ANTENNAS for Channels 11-26					
Part Number	Type (Description)	Gain	Application*	Min. Separation Required	Cable-loss
A24-Y6NF	Yagi (6-element)	8.8 dBi	Fixed	2 m	7.8 dB
A24-Y7NF	Yagi (7-element)	9.0 dBi	Fixed	2 m	8 dB
A24-Y9NF	Yagi (9-element)	10.0 dBi	Fixed	2 m	9 dB
A24-Y10NF	Yagi (10-element)	11.0 dBi	Fixed	2 m	10 dB
A24-Y12NF	Yagi (12-element)	12.0 dBi	Fixed	2 m	11 dB
A24-Y13NF	Yagi (13-element)	12.0 dBi	Fixed	2 m	11 dB
A24-Y15NF	Yagi (15-element)	12.5 dBi	Fixed	2 m	11.5 dB
A24-Y16NF	Yagi (16-element)	13.5 dBi	Fixed	2 m	12.5 dB
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5 dBi	Fixed	2 m	12.5 dB
A24-Y18NF	Yagi (18-element)	15.0 dBi	Fixed	2 m	14 dB

PANEL CLASS ANTENNAS for Channels 11 - 26					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-P8SF	Flat Panel	8.5 dBi	Fixed	2 m	8.2 dB
A24-P8NF	Flat Panel	8.5 dBi	Fixed	2 m	8.2 dB
A24-P13NF	Flat Panel	13.0 dBi	Fixed	2 m	12.7 dB
A24-P14NF	Flat Panel	14.0 dBi	Fixed	2 m	13.7 dB
A24-P15NF	Flat Panel	15.0 dBi	Fixed	2 m	14.7 dB
A24-P16NF	Flat Panel	16.0 dBi	Fixed	2 m	15.7 dB
A24-P19NF	Flat Panel	19.0 dBi	Fixed	2m	18.7 dB

OMNI-DIRECTIONAL ANTENNAS for Channels 11 - 26					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-C1	Surface Mount Integral Chip	-1.5 dBi	Fixed/Mobile	20 cm	N/A
A24-F2NF	Omni-directional (Fiberglass base station)	2.1 dBi	Fixed/Mobile	20 cm	N/A
A24-F3NF	Omni-directional (Fiberglass base station)	3.0 dBi	Fixed/Mobile	20 cm	0.3 dB
A24-F5NF	Omni-directional (Fiberglass base station)	5.0 dBi	Fixed/Mobile	20 cm	2.3 dB
A24-F8NF	Omni-directional (Fiberglass base station)	8.0 dBi	Fixed	2 m	5.3 dB
A24-F9NF	Omni-directional (Fiberglass base station)	9.5 dBi	Fixed	2 m	6.8 dB
A24-F10NF	Omni-directional (Fiberglass base station)	10.0 dBi	Fixed	2 m	7.3 dB
A24-F12NF	Omni-directional (Fiberglass base station)	12.0 dBi	Fixed	2 m	9.3 dB
A24-F15NF	Omni-directional (Fiberglass base station)	15.0 dBi	Fixed	2 m	12.3 dB
A24-W7NF	Omni-directional (Base station)	7.2 dBi	Fixed	2 m	4.5 dB
A24-M7NF	Omni-directional (Mag-mount base station)	7.2 dBi	Fixed	2 m	4.5 dB

Antennas approved for use with the XBee-PRO (S2) SE (Smart Energy) RF Modules

YAGI CLASS ANTENNAS for Channels 11-24					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-Y6NF	Yagi (6-element)	8.8 dBi	Fixed	2 m	N/A
A24-Y7NF	Yagi (7-element)	9.0 dBi	Fixed	2 m	N/A
A24-Y9NF	Yagi (9-element)	10.0 dBi	Fixed	2 m	N/A
A24-Y10NF	Yagi (10-element)	11.0 dBi	Fixed	2 m	N/A
A24-Y12NF	Yagi (12-element)	12.0 dBi	Fixed	2 m	N/A
A24-Y13NF	Yagi (13-element)	12.0 dBi	Fixed	2 m	N/A
A24-Y15NF	Yagi (15-element)	12.5 dBi	Fixed	2 m	N/A
A24-Y16NF	Yagi (16-element)	13.5 dBi	Fixed	2 m	N/A
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5 dBi	Fixed	2 m	N/A
A24-Y18NF	Yagi (18-element)	15.0 dBi	Fixed	2 m	0.5 dB

PANEL CLASS ANTENNAS for Channels 11 - 24					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-P8SF	Flat Panel	8.5 dBi	Fixed	2 m	N/A
A24-P8NF	Flat Panel	8.5 dBi	Fixed	2 m	N/A
A24-P13NF	Flat Panel	13.0 dBi	Fixed	2 m	N/A
A24-P14NF	Flat Panel	14.0 dBi	Fixed	2 m	N/A
A24-P15NF	Flat Panel	15.0 dBi	Fixed	2 m	N/A
A24-P16NF	Flat Panel	16.0 dBi	Fixed	2 m	N/A
A24-P19NF	Flat Panel	19.0 dBi	Fixed	2m	2.5 dB

OMNI-DIRECTIONAL ANTENNAS for Channels 11 - 24					
Part Number	Type (Description)	Gain	Application*	Min. Separation Required	Cable-loss
A24-C1	Surface Mount integral chip	-1.5dBi	Fixed/Mobile	20 cm	N/A
A24-F2NF	Omni-directional (Fiberglass base station)	2.1 dBi	Fixed/Mobile	20 cm	N/A
A24-F3NF	Omni-directional (Fiberglass base station)	3.0 dBi	Fixed/Mobile	20 cm	N/A
A24-F5NF	Omni-directional (Fiberglass base station)	5.0 dBi	Fixed/Mobile	20 cm	N/A
A24-F8NF	Omni-directional (Fiberglass base station)	8.0 dBi	Fixed	2 m	N/A
A24-F9NF	Omni-directional (Fiberglass base station)	9.5 dBi	Fixed	2 m	N/A
A24-F10NF	Omni-directional (Fiberglass base station)	10.0 dBi	Fixed	2 m	N/A
A24-F12NF	Omni-directional (Fiberglass base station)	12.0 dBi	Fixed	2 m	N/A
A24-F15NF	Omni-directional (Fiberglass base station)	15.0 dBi	Fixed	2 m	0.5 dB
A24-W7NF	Omni-directional (Base station)	7.2 dBi	Fixed	2 m	N/A
A24-M7NF	Omni-directional (Mag-mount base station)	7.2 dBi	Fixed	2 m	N/A

Antennas approved for use with the XBee-PRO (S2B) SE (Smart Energy) RF Modules

YAGI CLASS ANTENNAS for Channels 11 to 24					
Part Number	Type (Description)	Gain	Application*	Min. Separation	Minimum Cable Loss/ Power Reduction/ Attenuation Required for 18dBm Output
A24-Y6NF	Yagi (6-element)	8.8 dBi	Fixed	2 m	N/A
A24-Y7NF	Yagi (7-element)	9.0 dBi	Fixed	2 m	N/A
A24-Y9NF	Yagi (9-element)	10.0 dBi	Fixed	2 m	N/A
A24-Y10NF	Yagi (10-element)	11.0 dBi	Fixed	2 m	N/A
A24-Y12NF	Yagi (12-element)	12.0 dBi	Fixed	2 m	N/A
A24-Y13NF	Yagi (13-element)	12.0 dBi	Fixed	2 m	N/A
A24-Y15NF	Yagi (15-element)	12.5 dBi	Fixed	2 m	N/A
A24-Y16NF	Yagi (16-element)	13.5 dBi	Fixed	2 m	N/A
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5 dBi	Fixed	2 m	N/A
A24-Y18NF	Yagi (18-element)	15.0 dBi	Fixed	2 m	N/A

PANEL CLASS ANTENNAS for Channels 11 to 24					
Part Number	Type (Description)	Gain	Application*	Min. Separation	Minimum Cable Loss/ Power Reduction/ Attenuation Required for 18dBm Output
A24-P8SF	Flat Panel	8.5 dBi	Fixed	2 m	N/A
A24-P8NF	Flat Panel	8.5 dBi	Fixed	2 m	N/A
A24-P13NF	Flat Panel	13.0 dBi	Fixed	2 m	N/A
A24-P14NF	Flat Panel	14.0 dBi	Fixed	2 m	0.8 dB
A24-P15NF	Flat Panel	15.0 dBi	Fixed	2 m	1.8 dB
A24-P16NF	Flat Panel	16.0 dBi	Fixed	2 m	2.8 dB
A24-P19NF	Flat Panel	19.0 dBi	Fixed	2 m	5.8 dB

OMNI-DIRECTIONAL ANTENNAS for Channels 11 to 24					
Part Number	Type (Description)	Gain	Application*	Min. Separation	Minimum Cable Loss/ Power Reduction/ Attenuation Required for 18dBm Output
A24-F9NF	Omni-directional (Fiberglass base station)	9.5 dBi	Fixed	2 m	N/A
A24-F10NF	Omni-directional (Fiberglass base station)	10.0 dBi	Fixed	2 m	N/A
A24-F12NF	Omni-directional (Fiberglass base station)	12.0 dBi	Fixed	2 m	N/A
A24-F15NF	Omni-directional (Fiberglass base station)	15.0 dBi	Fixed	2 m	N/A

OMNI-DIRECTIONAL ANTENNAS for Channels 11 to 25					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation**	**Minimum Cable Loss/ Power Reduction/ Attenuation Required for 18dBm Output**
A24-F2NF	Omni-directional (Fiberglass base station)	2.1 dBi	Fixed/Mobile	20 cm	N/A
A24-F3NF	Omni-directional (Fiberglass base station)	3.0 dBi	Fixed/Mobile	20 cm	N/A
A24-F5NF	Omni-directional (Fiberglass base station)	5.0 dBi	Fixed/Mobile	20 cm	N/A
A24-F8NF	Omni-directional (Fiberglass base station)	8.0 dBi	Fixed	2 m	N/A
A24-W7NF	Omni-directional (Base station)	7.2 dBi	Fixed	2 m	N/A
A24-M7NF	Omni-directional (Mag-mount base station)	7.2 dBi	Fixed	2 m	N/A

*** 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 20cm 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.

RF Exposure

WARNING: 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.

The preceding statement must be included as a CAUTION statement in OEM product manuals in order to alert users of FCC RF Exposure compliance.

Europe (ETSI)

The XBee Module has been certified for use in several European countries. For a complete list, refer to www.digi.com

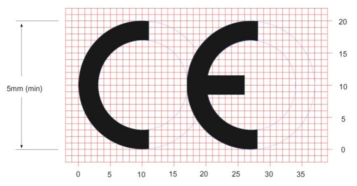
If the XBee Modules are incorporated into a product, the manufacturer must ensure compliance of the final product to the European harmonized EMC and low-voltage/safety standards. A Declaration of Conformity must be issued for each of these standards and kept on file as described in Annex II of the R&TTE Directive.

Furthermore, the manufacturer must maintain a copy of the XBee user manual documentation and ensure the final product does not exceed the specified power ratings, antenna specifications, and/ or installation requirements as specified in the user manual. If any of these specifications are exceeded in the final product, a submission must be made to a notified body for compliance testing to all required standards.

OEM Labeling Requirements

The 'CE' marking must be affixed to a visible location on the OEM product.

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 5mm except where this is not possible on account of the nature of the apparatus.
- The CE marking must be affixed visibly, legibly, and indelibly.

Restrictions

France: Outdoor use limited to 10 mW EIRP within the band 2454-2483.5 MHz.

Norway: Norway prohibits operation near Ny-Alesund in Svalbard. More information can be found at the Norway Posts and Telecommunications site (www.npt.no).

Declarations of Conformity

Digi has issued Declarations of Conformity for the XBee Modules concerning emissions, EMC and safety. Files can be obtained by contacting Digi Support.

Important Note:

Digi does not list the entire set of standards that must be met for each country. Digi customers assume full responsibility for learning and meeting the required guidelines for each country in their distribution market. For more information relating to European compliance of an OEM product incorporating the XBee Module, contact Digi, or refer to the following web sites:

CEPT ERC 70-03E - Technical Requirements, European restrictions and general requirements: Available at www.ero.dk/.

R&TTE Directive - Equipment requirements, placement on market: Available at www.ero.dk/.

Approved Antennas

XBee RF Module

The following antennas are approved for use with the embedded XBee RF Module:

- Dipole (2.1 dBi, Omni-directional, Articulated RPSMA, Digi part number A24-HABSM)
- Chip Antenna (-1.5 dBi)
- Attached Monopole Whip (1.5 dBi)
- Integral PCB Antenna (-0.5 dBi).

XBee-PRO (S2) RF Module

The following antennas are approved for use with the embedded XBee-PRO RF Module:

- Dipole (2.1 dBi, Omni-directional, Articulated RPSMA, Digi part number A24-HABSM)
- Chip Antenna (-1.5 dBi)
- Attached Monopole Whip (1.5 dBi)
- Integral PCB Antenna (-0.5 dBi)

XBee-PRO (S2B) RF Module

The following antennas are approved for use with the embedded XBee-PRO Plus RF Module:

- Dipole (2.1 dBi, Omni-directional, Articulated RPSMA, Digi part number A24-HABSM)
- Integral PCB Antenna (-1.5 dBi)
- Attached Monopole Whip (1.5 dBi)

Canada (IC)

This device complies with Industry Canada licence-exempt RSS standard(s). Operation is subject to the following two conditions: (1) this device may not cause interference, and (2) this device must accept any interference, including interference that may cause undesired operation of the device.

Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes : (1) l'appareil ne doit pas produire de brouillage, et (2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

Labeling Requirements

Labeling requirements for Industry Canada are similar to those of the FCC. A clearlyvisible label on the outside of the final product enclosure must display the following text:

Contains Model XBee Radio, IC: 4214A-XBEE2

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.

If it contains an XBee-PRO (S2) Module, the clearly visible label on the outside of the final product enclosure must display the following text:

Contains Model XBee PRO Radio, IC: 1846A-XBEEPRO2

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.

Contains Model XBee PRO S2B Radio, IC: 1846A-PROS2B

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 table above with the maximum permissible gain and required antenna impedance for each antenna type indicated. Antenna types not included in this list, having a gain greater than the maximum gain indicated for that type, are strictly prohibited for use with this device. 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 ci-dessous et ayant un gain admissible maximal et l'impédance requise pour chaque type d'antenne. Les types d'antenne non inclus dans cette liste, ou dont le gain est supérieur au gain maximal indiqué, sont strictement interdits pour l'exploitation de l'émetteur.

Detachable Antenna

Under Industry Canada regulations, this radio transmitter may only operate using an antenna of a type and maximum (or lesser) gain approved for the transmitter by Industry Canada. To reduce potential radio interference to other users, the antenna type and its gain should be so chosen that the equivalent isotropically radiated power (e.i.r.p.) is not more than that necessary for successful communication.

Conformément à la réglementation d'Industrie Canada, le présent émetteur radio peut fonctionner avec une antenne d'un type et d'un gain maximal (ou inférieur) approuvé pour l'émetteur par Industrie Canada. Dans le but de réduire les risques de brouillage radioélectrique à l'intention des autres utilisateurs, il faut choisir le type d'antenne et son gain de sorte que la puissance isotrope rayonnée équivalente (p.i.r.e.) ne dépasse pas l'intensité nécessaire àl'établissement d'une communication satisfaisante.

Australia (C-Tick)

These modules comply with requirements to be used in end products in Australia. All products with EMC and radio communications must have a registered C-Tick mark. Registration to use the compliance mark will only be accepted from Australian manufacturers or importers, or their agent, in Australia.

In order to have a C-Tick mark on an end product, a company must comply with a or b below.

a. have a company presence in Australia.

b. have a company/distributor/agent in Australia that will sponsor the importing of the end product.

Contact Digi for questions related to locating a contact in Australia.

Appendix C: Migrating from XBee ZB to XBee SE

The following list includes the significant differences in XBee SE compared to XBee ZB.

- API coordinator, router, and end device targets are supported, AT/Transparent targets are not.
- Routers and end devices will perform key establishment after joining (if enabled with EO command). (ZB firmware does not support key establishment.)
- New AI codes in SE firmware indicate the progress of key establishment.
- Fragmentation in SE supports up to 128 byte payloads. (ZB supports up to 255 byte payloads.)
- The coordinator and routers can have up to 6 end device children each, compared with 10 - 12 in ZB firmware.
- Routers and end devices perform up to 3 joining attempts. Range Extenders will make 3 quick attempts to join, followed by one join attempt per minute for 15 minutes, and once per hour after that. Joining attempts may be triggered by a button press, reset, or AT command (CB). (In ZB firmware, the XBee regularly attempts joining until successful.)
- Joining can be temporarily enabled by a button press, changing the NJ value, or AT command (CB).
- The explicit transmit and receive frames are required (0x11 and 0x91). ZigBee transmit and receive API frames (used in ZB) are not supported (0x10 and 0x90) in SE firmware.

The following ZB features are not supported in SE:

- IO sampling (IR, IS, IC)
- Analog and digital IO configuration (D1, D2, D3, D4, D5, D6, P0, P1, P2, RP, PR)
- Network discovery and diagnostics commands (NI, ND, DN, NR1, NW, JV)
- Remote AT commands (0x17 and 0x97 API frames)
- Loopback cluster ID
- Over-the-air firmware updates
- Some AT commands have been made read only (such as ZS and EE), and others have limited parameter ranges. See the command table for details.

Manufacturers of end products that use SE firmware must do the following (see Appendix D):

- Be members of the ZigBee Alliance
- Obtain certificates for the product
- Certify the end product with a ZigBee test lab.

Appendix D: Smart Energy Certificates

All devices that operate in a ZigBee smart energy deployment must have a certificate installed that provides the device with a private key and digital certificate that are used to derive a link key during key establishment. A certificate must be issued by a certificate authority. Each certificate is tied to the 64-bit extended address (serial number) of the device.

The certificate authority can generate certificates for general test and development use. These "test" certificates allow devices to implement key establishment, but they cannot communicate with devices that have live "production" certificates for a certified smart energy deployment.

The XBee SE modules have key establishment disabled by default, and no certificates installed on them. This is sufficient for development purposes only. When certifying or deploying a smart energy device, key establishment must be enabled (see EO command). Test certificates are required when certifying smart energy devices. Production certificates are required for certified smart energy deployments.

Please contact a certificate authority to obtain certificates. Currently the only authority is Certicom:

- See http://www.certicom.com/index.php/gencertregister to register for a few test certificates
- Contact jalfred@certicom.com for production certificates or a sizable number of test certificates.

Once a certificate is obtained, it can be programmed onto an XBee SE module in one of two ways:

1. Serially via API commands (contact se.developer@digi.com for assistance)

2. Through the SIF header via an Ember programming tool (requires some modification to the module and purchase of hardware from www.ember.com)

Example

A certificate consists of four keys, only three of which need to be serially installed on an XBee. The Device Public Key is not needed. Here is an example of a test certificate, for MAC address 0013A200404C15A4:

CA Public Key: 0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b8

Device Implicit Cert:
03061958d95eaf5477be7c89a94a85aabbb08cdd3d0b0013a200404c15a454455354534543410109
0010000000000000

Device Private Key: 03ea7f821cd85f0d4f6a782b2e6994df1cc48be8fd

Device Public Key: 030149359f204a4e010835d69baaddfcd857d395d647

Three AT commands are used for installing certificate keys.

- ZU - 22-byte public key (CA Public key)
- ZT - 48-byte implicit device certificate (Device Implicit Cert)
- ZV - 21-byte private key (Device Private key)

Convert the AT commands and their parameters into AT command API packets.

The ZU command with the CA Public key parameter:

7E 00 1A 08 01 5A 55 02 00 FD E8 A7 F3 D1 08 42 24 96 2A 4E 7C 54 E6 9A C3 F0 4D A6 B8 CB

The ZT command with the Device Implicit Cert parameter:

7E 00 34 08 01 5A 54 03 06 19 58 D9 5E AF 54 77 BE 7C 89 A9 4A 85 AA BB B0 8C DD 3D 0B 00 13 A2 00 40 4C 15 A4 54 45 53 54 53 45 43 41 01 09 00 10 00 00 00 00 00 00 AC

The ZV command with the Device Private key parameter:

7E 00 19 08 01 5A 56 03 EA 7F 82 1C D8 5F 0D 4F 6A 78 2B 2E 69 94 DF 1C C4 8B E8 FD 42

With X-CTU (or by similar means), write the API packets to the serial port of the XBee.

Send a write command (WR) to commit the certificate to non-volatile memory.

7E 00 04 08 01 77 72 0D

Reset the router (FR) so it will restart using the new certificate.

7E 00 04 08 01 46 52 5E

Send a verify certificate (VC) command to verify the presence of a certificate.

A returned parameter of '1' indicates a certificate is present; a '0' indicates one is not present.

7E 00 04 08 01 56 43 5D

To erase a certificate, send AT command API packets with keys set to zero (ZU, ZT, ZV).

7E 00 05 08 01 5A 55 00 47

7E 00 05 08 01 5A 54 00 48

7E 00 05 08 01 5A 56 00 46

Appendix E:Additional Information

1-Year Warranty

XBee Modules from Digi International, Inc. (the "Product") are warranted against defects in materials and workmanship under normal use, for a period of 1-year from the date of purchase. In the event of a product failure due to materials or workmanship, Digi will repair or replace the defective product. For warranty service, return the defective product to Digi International, shipping prepaid, for prompt repair or replacement.

The foregoing sets forth the full extent of Digi International's warranties regarding the Product. Repair or replacement at Digi International's option is the exclusive remedy. THIS WARRANTY IS GIVEN IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, AND DIGI SPECIFICALLY DISCLAIMS ALL WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL DIGI, ITS SUPPLIERS OR LICENSORS BE LIABLE FOR DAMAGES IN EXCESS OF THE PURCHASE PRICE OF THE PRODUCT, FOR ANY LOSS OF USE, LOSS OF TIME, INCONVENIENCE, COMMERCIAL LOSS, LOST PROFITS OR SAVINGS, OR OTHER INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT, TO THE FULL EXTENT SUCH MAY BE DISCLAIMED BY LAW. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES. THEREFORE, THE FOREGOING EXCLUSIONS MAY NOT APPLY IN ALL CASES. This warranty provides specific legal rights. Other rights which vary from state to state may also apply.