# XBee-PRO® 900 RF Modules

XBee-PRO® 900 RF Modules

RF Module Operation

RF Module Configuration
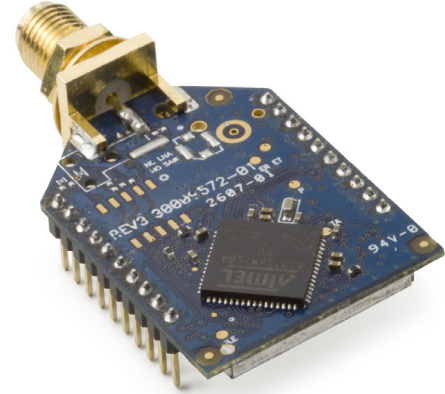
Appendices

**RF Modules by Digi International**
**Firmware version:**

1x5x XBee-PRO 900

21x5x XBee-PRO 900 (International variant)

| 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] |
| | Live Chat: | www.digi.com |
| | Online Support: | http://www.digi.com/support/eservice/login.jsp |
| | Email: | rf-experts@digi.com |

# Contents

# 1. XBee-PRO® RF Modules

The XBee-PRO® 900 RF Modules were engineered to support the unique needs of low-cost, low-power wireless sensor networks. The modules require minimal power and provide reliable delivery of data between remote devices.

The modules operate within the ISM 900 MHz frequency band.

## Key Features

### High Performance, Low Cost

- Indoor/Urban: up to 450ft (140m)
- Outdoor line-of-sight: up to 1.8 miles (3 km)
- Transmit Power Output: 50 mW (+17dBm)
- Receiver Sensitivity: -100dBm
- RF Data Rate: 156.25 kbps

### Advanced Networking & Security

- Retries and Acknowledgements
- Point-to-point, point-to-multipoint and peer-to-peer topologies supported
- (FHSS) Frequency Hopping Spread Spectrum

### Low Power

- TX Current: 210mA (@3.3 V)
- RX Current: 80mA (@3.3 V)
- Sleep current: 48μA (typical @3.3 V)

### Easy-to-Use

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

## Worldwide Acceptance

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

ISM (Industrial, Scientific & Medical) **900 frequency band**

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

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

## Specifications

**Specifications of the XBee-PRO®RF Module**

| Specification | XBee-PRO®900 |
|---|---|
| Performance | |
| Indoor/Urban Range | up to 450ft (140m) |
| Outdoor RF line-of-sight Range | up to 1.8 miles (3km) w/2.1dB dipole antenna<br>up to 6 miles (10km) w/high gain antenna |
| Transmit Power Output | +17 dBm (50 mW) |
| RF Data Rate | 156.25 kbps |
| Data Throughput | Up to 87000 bps |
| Serial Interface Data Rate (software selectable) | 3.3V CMOS Serial UART (5V tolerant UART) |
| Receiver Sensitivity | -100dBm |
| Power Requirements | |
| Supply Voltage | 3.0 to 3.6 VDC |
| Transmit Current | 210 mA,  (180 mA typical) |
| Idle / Receive Current | 80 mA (typical) |
| Sleep Current | 48 $\mu$A typical @ 3.3V |
| General | |
| Operating Frequency Band | Standard Variant: 902-928 MHz (ISM)<br>International Variant: 916-928 MHz |
| Dimensions | 0.962 in x 1.312 in (2.443 cm x 3.332 cm) |
| Operating Temperature | -40 to 85 C (Industrial), 0 to 95% non-condensing |
| Antenna Options | 1/4 wave wire antenna, RPSMA RF connector, U.FI RF connector |
| Digital I/O | 13 I/O lines |
| ADC | 6 10-bit analog inputs |
| Networking & Security | |
| Supported Network Topologies | Point-to-point, point-to-multipoint, peer-to-peer |
| Number of Channels (software selectable) | 8 hopping patterns on 12 channels (standard variant)<br>4 hopping patterns on 5 channels  (International variant) |
| Addressing Options | PAN ID, Channel and 64-bit addresses |
| Encryption | 128 bit AES |
| Agency Approvals | |
| United States (FCC Part 15.247) | FCC ID: MCQ-XBEE09P |
| Australia | C-Tick |
| Industry Canada (IC) | IC: 1846A-XBEE09P |
| Europe (CE) | N/A |

## Mechanical Drawings

**Mechanical drawings of the XBee-PRO®RF Modules (antenna opstions not shown)**
.
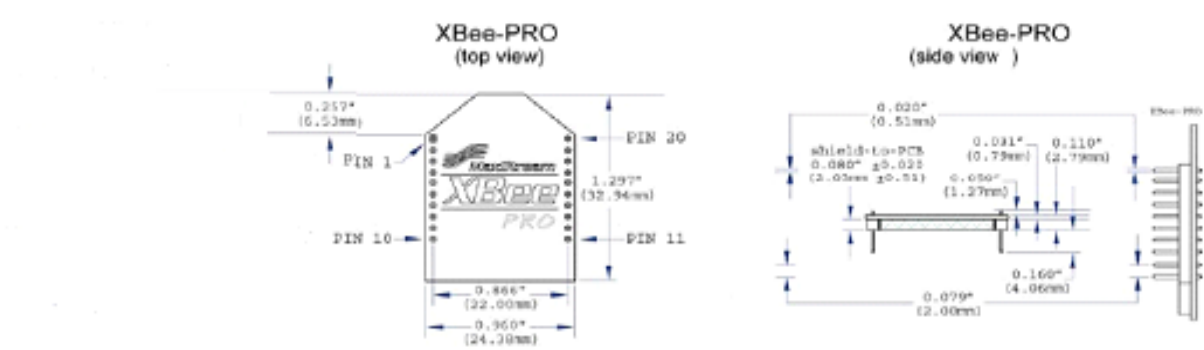


**Mechanical Drawings for the RPSMA Variant**
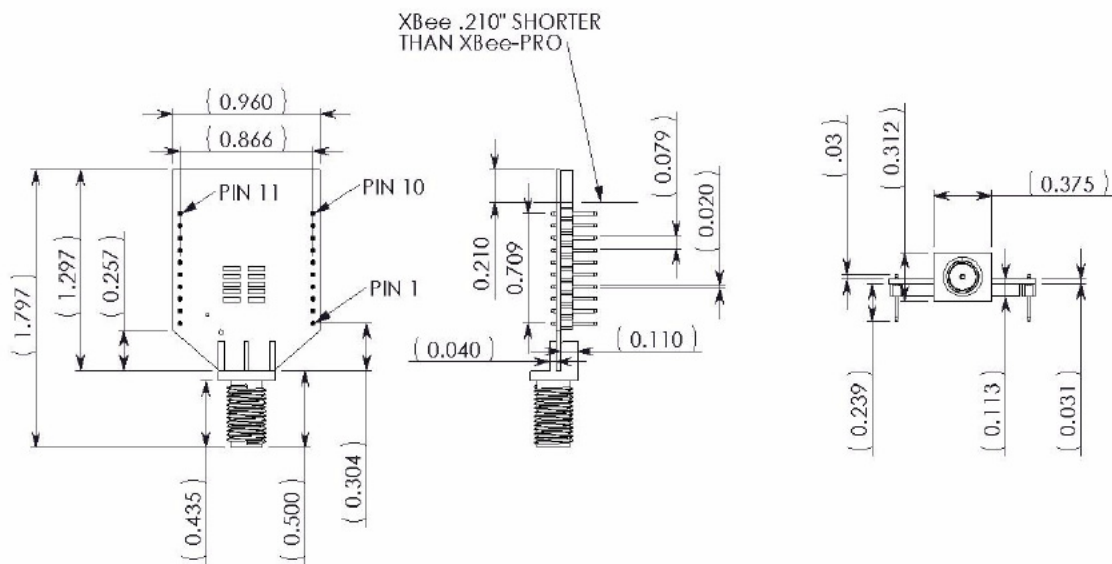


## Mounting Considerations

The XBee-PRO® RF Module (through-hole) was designed to mount into a receptacle (socket) and therefore does not require any soldering when mounting it to a board. The 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 manu-facturers 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)
- 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

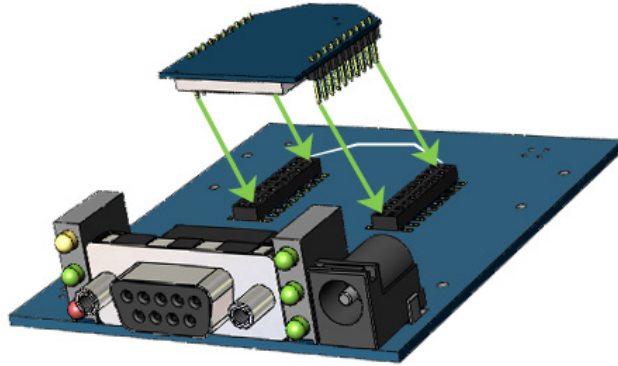Digi also recommends printing an outline of the module on the board to indicate the orientation the module should be mounted.

# Pin Signals

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

| Pin # | Name | Direction | Description | Max Voltage |
|-------|------|-----------|-------------|-------------|
| 1 | Vcc | - | Power supply | 3.6V |
| 2 | DOUT | Output | UART Data Out | Vcc |
| 3 | DIN / **CONFIG** | Input | UART Data In | 5V |
| 4 | DIO12 | Either | Digital I/O 12 | Vcc |
| 5 | $\overline{\text{RESET}}$ | Input/Open drain output | Module Reset (reset pulse must be at least 100 us. This must be driven as an open drain/collector. The module will drive this line low when a reset occurs. This line should never be driven high.) | Vcc |
| 6 | PWM0 / RSSI / DIO10 | Either | PWM Output 0 / RX Signal Strength Indicator / Digital IO | Vcc |
| 7 | PWM / DIO11 | Either | PWM Output 1 / Digital I/O 11 | Vcc |
| 8 | [reserved] | - | Do not connect | n/a |
| 9 | $\overline{\text{DTR}}$ / SLEEP_RQ/ DIO8 | Either | Pin Sleep Control Line or Digital IO 8 | 5V |
| 10 | GND | - | Ground | 0V |
| 11 | AD4/DIO4 | Either | Analog Input 4 or Digital I/O 4 | Vcc |
| 12 | $\overline{\text{CTS}}$ / DIO7 | Either | Clear-to-Send Flow Control or Digital I/O 7 | Vcc |
| 13 | ON / $\overline{\text{SLEEP}}$ | Output | Module Status Indicator or Digital I/O 9 | Vcc |
| 14 | VREF | - | This line must be connected if analog IO  sampling is desired. Must be between 2.6 V and Vcc. | Vcc |
| 15 | Associate / DIO5/AD5 | Either | Associated Indicator, Digital I/O 5 | Vcc |
| 16 | $\overline{\text{RTS}}$ / DIO6 | Either | Request-to-Send Flow Control, Digital I/O 6 | 5V |
| 17 | AD3 / DIO3 | Either | Analog Input 3 or Digital I/O 3 | Vcc |
| 18 | AD2 / DIO2 | Either | Analog Input 2 or Digital I/O 2 | Vcc |
| 19 | AD1 / DIO1 | Either | Analog Input 1 or Digital I/O 1 | Vcc |
| 20 | AD0 / DIO0 / Commissioning Button | Either | Analog Input 0, Digital I/O 0, or Commissioning Button | Vcc |

## 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 1.0 uF and 8.2pF capacitor are recommended to be placed as near to pin1 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 100mV peak to peak. To ensure proper power up, Vcc SLOPE must be superior or equal to 6V/ms.

### Recommended Pin Connections

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

All unused pins should be left disconnected. All inputs on the radio can be pulled high with internal pull-up resistors using the PR software command. 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 button pin (pin 20). The Associate LED pin will flash differently depending on the state of the module, and a pushbutton attached to pin 20 can enable various deployment and troubleshooting functions without having to send UART commands.

The combined source and sink capabilities of the module are limited to 120mA for all pins on the module. Module pins 11 and 15 can source/sink a maximum of 2mA; pins 9, 6, and 13 can source/sink a maximum of 16mA; and all other pins can source/sink a maximum of 8mA.

If analog sampling is desired the VRef pin (pin 14) should be attached to a voltage reference.

### 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. 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. Large ground planes on a host PCB should not adversely affect maximum range, but they may affect radiation patterns of onboard XBee antennas.

## Electrical Characteristics

DC Characteristics of the XBee-PRO® 900  (VCC =3.0-3.6VDC)

| 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 | $I_{OL}$ = 2 mA, VCC >= 3.0 V | - | - | 0.18*VCC | V |
| $V_{OH}$ | Output High Voltage | $I_{OH}$  = 2 mA, VCC >= 3.0 V | 0.82*VCC | - | - | V |
| $II_{IN}$ | Input Leakage Current | $V_{IN}$ = VCC or GND, all inputs, per pin | - | - | 0.5uA | uA |

# 2. RF Module Operation

## Serial Communications

The XBee-PRO® RF 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 Digi 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.)
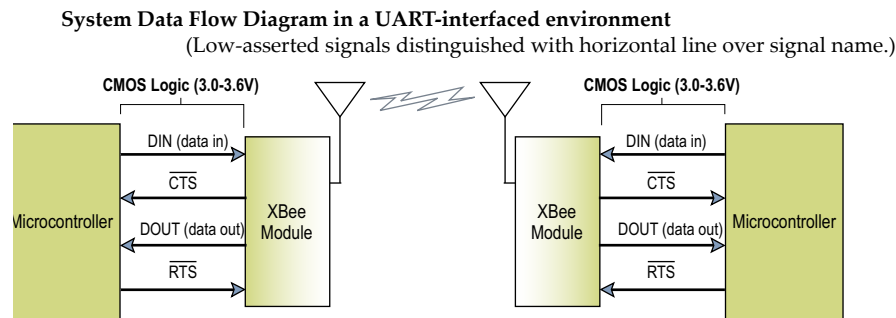


#### Serial Data

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)
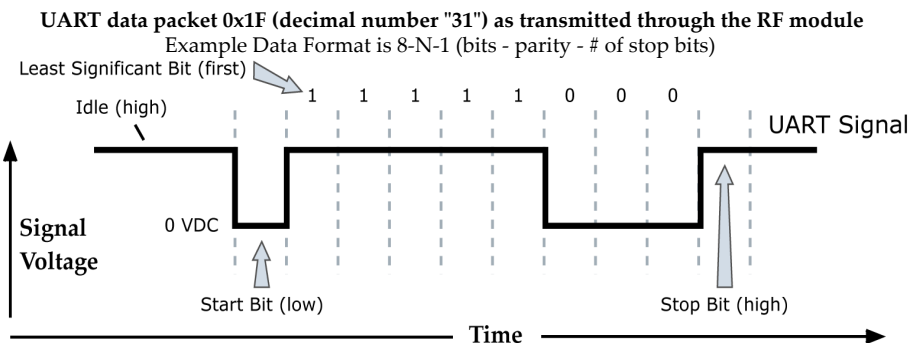


The module UART performs tasks, such as timing and parity checking, that are needed for data communications. Serial communications depend on the two UARTs to be configured with compatible settings (baud rate, parity, start bits, stop bits, data bits).

### Serial Buffers

The XBee-PRO modules maintain 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.

**Internal Data Flow Diagram**
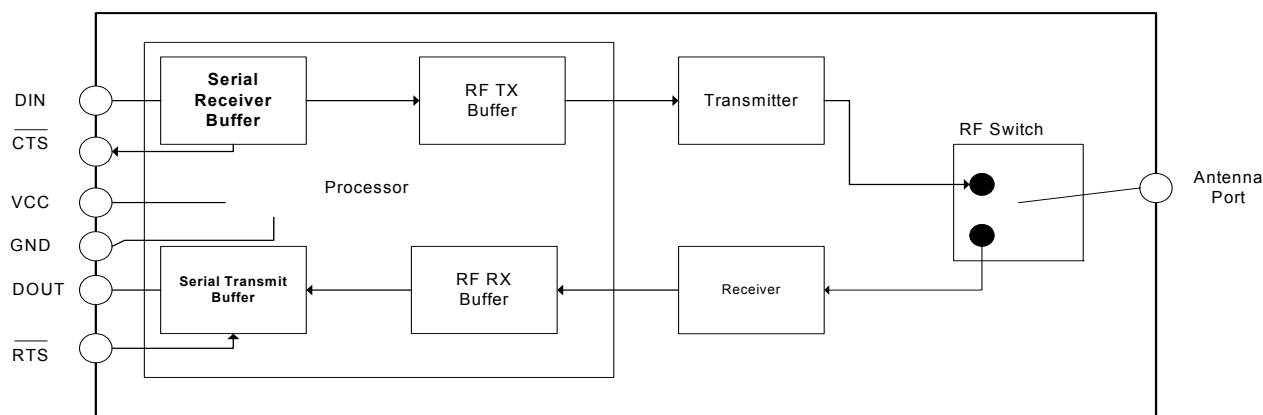


### Serial Receive Buffer

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, 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:**
• 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.

### Serial Transmit Buffer

When RF data is received, the data is moved into the serial transmit buffer and is sent out the serial port. 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**
• 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. Even occasional transmissions from a large number of modules can quickly add up and overflow the transmit buffer.

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

### $\overline{CTS}$ Flow Control

If $\overline{CTS}$ flow control is enabled (D7 command), when the serial receive buffer is filled with FT bytes, the module de-asserts $\overline{CTS}$ (sets it high) to signal to the host device to stop sending serial data. $\overline{CTS}$ is re-asserted when less than FT - 16 bytes are in the UART receive buffer. (See command description for the FT command.)

### $\overline{RTS}$ Flow Control

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

## Serial Interface Protocols

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

### Transparent Operation

When operating in transparent mode, the modules act as a serial line replacement. All UART data received through the DIN pin is queued up for RF transmission. When RF data is received, the data is sent out through the DOUT pin. The module configuration parameters are configured using the AT command mode interface.

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

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

### API Operation

API operation is an alternative to transparent operation. The frame-based API extends the level to which a host application can interact with the networking capabilities of the module. When in API mode, all data entering and leaving the module's UART 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, sync status, etc.

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

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

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

**A Comparison of Transparent and API Operation**

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

| Transparent Operation Features | |
|---|---|
| Simple Interface | All received serial data is transmitted unless the module is in command mode. |
| Easy to support | It is easier for an application to support transparent operation and command mode |
| **API Operation Features** | |
| Easy to manage data transmissions to multiple destinations | Transmitting RF data to multiple remotes only requires changing the address in the API frame. This process is much faster than in transparent operation where the application must enter AT command mode, change the address, exit command mode, and then transmit data. Each API transmission can return a transmit status frame indicating the success or reason for failure. |
| Received data frames indicate the sender's address | All received RF data API frames indicate the source address. |
| Advanced networking diagnostics | API frames can provide indication of IO samples from remote devices, and node identification messages. |
| Remote Configuration | Set / read configuration commands can be sent to remote devices to configure them as needed using the API. |

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

• sends RF data to multiple destinations

• sends remote configuration commands to manage devices in the network

• receives IO samples from remote devices

• receives RF data packets from multiple devices, and the application needs to know which device sent which packet.

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

To implement API operations, refer to the API Operation chapter (Chapter 7).

## Idle Mode

When not receiving or transmitting data, the RF module is in Idle Mode. During Idle Mode, the RF module is checking for valid RF data. 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)

• Command Mode (Command Mode Sequence is issued)

• Sleep Mode (A device is configured for sleep)

## 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.

## Receive Mode

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

### Command Mode

To modify or read RF Module parameters, the module must first enter into Command Mode - a state in which incoming serial characters are interpreted as commands. Refer to the API Mode section for an alternate means of configuring modules.

#### AT Command Mode

**To Enter AT Command Mode:**
Send the 3-character command sequence "+++" and observe guard times before and after the command characters. [Refer to the "Default AT Command Mode Sequence" below.]

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

• No characters sent for one second [GT (Guard Times) parameter = 0x3E8]

• Input three plus characters ("+++") within one second [CC (Command Sequence Character) parameter = 0x2B.]

• No characters sent for one second [GT (Guard Times) parameter = 0x3E8]

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

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

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

> NOTE: Failure to enter AT Command Mode is most commonly due to baud rate mismatch. When using XCTU, ensure that the 'Baud' setting on the "PC Settings" tab matches the interface data rate of the RF module. By default, the BD parameter = 3 (9600 bps).

**To Send AT Commands:**
Send AT commands and parameters using the syntax shown below.

Syntax for sending AT Commands

**"AT" Prefix** + **ASCII Command** + **Space** (Optional) + **Parameter** (Optional, HEX) + **Carriage Return**

Example: ATDL 1F<CR>

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

The preceding example would change the RF module Destination Address (Low) to "0x1F". To store the new value to non-volatile (long term) memory, subsequently send the WR (Write) command.

For modified parameter values to persist in the module's registry after a reset, changes must be saved to non-volatile memory using the WR (Write) Command. Otherwise, parameters are restored to previously saved values after the module is reset.

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

#### Applying Command Changes

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

• The AC (Apply Changes) command is issued.

• AT command mode is exited.

**To Exit AT Command Mode:**
  1. Send the ATCN (Exit Command Mode) command (followed by a carriage return).

[OR]

2.   If no valid AT Commands are received within the time specified by CT (Command Mode Timeout) Command, the RF module automatically returns to Idle Mode.

For an example of programming the RF module using AT Commands and descriptions of each configurable parameter, refer to the Command Reference Tables chapter.

## Sleep Mode

Sleep modes allow the RF module to enter states of low power consumption when not in use. The XBee RF Modules support both pin sleep (sleep mode entered on pin transition) and cyclic sleep (Module sleeps for a fixed time).  XBee sleep modes are discussed in detail in Chapter 4.

# 3. Advanced Application Features

## Remote Configuration Commands

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

### Sending a Remote Command

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

### Applying Changes on Remote Devices

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

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

### Remote Command Responses

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

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

### Device Configuration

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

### Device Placement

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

**AT Configuration (AP=0)**

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

**API Configuration (AP=1 or AP=2)**

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

## RSSI Indicators

It is possible to measure the received signal strength on a device using the DB command. DB returns the RSSI value (measured in –dBm) of the last received packet. However, this number can be misleading. XBee-PRO 900 modules have the limitation that only RSSI values near the sensitivity level of the radio are accurately reported. Signals received which are stronger than approximately -85 dBm will be reported as approximately -85dBm.

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

## Network Discovery

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

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

## Commissioning Pushbutton and Associate LED

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

**Commissioning Pushbutton and Associate LED Functionalities**
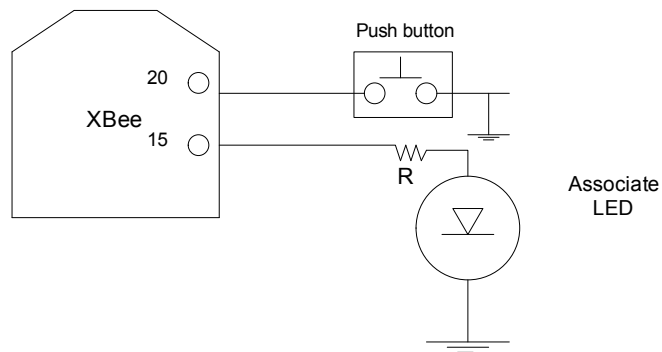


**A pushbutton and an LED can be connected to module pins 20 and 15 respectively  to support the commissioning pushbutton and associated LED functionalities.**

### Commissioning Pushbutton

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

| Button Presses | Sleep Configuration | Action |
|---|---|---|
| 1 | Not configured for sleep | Immediately sends a Node Identification broadcast transmission. All devices that receive this transmission will blink their Associate LED rapidly for 1 second. All API devices that receive this transmission will send a Node Identification frame out their UART (API ID 0x95) |
| 1 | Configured for sleep | Wakes the module for 30 seconds. Immediately sends a Node Identification broadcast transmission. All devices that receive this transmission will blink their Associate LED rapidly for 1 second. All API devices that receive this transmission will send a Node Identification frame out their UART (API ID 0x95). |
| 2 | Not configured for sleep | No effect. |
| 4 | Any | Issues an ATRE to restore module parameters to default values. |

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

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

### Associate LED

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

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

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

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

### Diagnostics Support

The Associate pin works with the commissioning pushbutton to provide additional diagnostics behaviors to aid in deploying and testing a network. If the commissioning push button is pressed once the device transmits a broadcast node identification packet. If the Associate LED functionality is enabled (D5 command), a device that receive this transmission will blink its Associate pin rapidly for 1 second.

## I/O Line Monitoring

### I/O Samples

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

### Queried Sampling

Parameters for the pin configuration commands typically include the following:

| Pin Command Parameter | Description |
|---|---|
| 0 | Unmonitored digital input |
| 1 | Reserved for pin-specific alternate functionalities. |
| 2 | Analog input (A/D pins) or PWM output (PWM pins) |
| 3 | Digital input, monitored. |
| 4 | Digital output, default low. |
| 5 | Digital output, default high. |
| 6-9 | Alternate functionalities, where applicable. |

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

| Module Pin Names | Module Pin Number | Configuration Command |
|---|---|---|
| CD / DIO12 | 4 | P2 |
| PWM0 / RSSI / DIO10 | 6 | P0 |
| PWM1 / DIO11 | 7 | P1 |
| DTR / SLEEP_RQ / DIO8 | 9 | D8 |
| AD4 / DIO4 | 11 | D4 |
| CTS / DIO7 | 12 | D7 |
| ON_SLEEP / DIO9 | 13 | D9 |
| ASSOC / AD5 / DIO5 | 15 | D5 |
| RTS / DIO6 | 16 | D6 |
| AD3 / DIO3 | 17 | D3 |
| AD2 / DIO2 | 18 | D2 |
| AD1 / DIO1 | 19 | D1 |
| AD0 / DIO0 / CommissioningButton | 20 | D0 |

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

| 1 | Sample Sets | Number of sample sets in the packet. (Always set to 1.) |
|---|---|---|

| 2 | Digital Channel Mask | Indicates which digital IO lines have sampling enabled. Each bit corresponds to one digital IO line on the module. <br><br> • bit 0 = AD0/DIO0 <br> • bit 1 = AD1/DIO1 <br> • bit 2 = AD2/DIO2 <br> • bit 3 = AD3/DIO3 <br> • bit 4 = DIO4 <br> • bit 5 = ASSOC/DIO5 <br> • bit 6 = RTS/DIO6 <br> • bit 7 = CTS/GPIO7 <br> • bit 8 = DTR / SLEEP_RQ / DIO8 <br> • bit 9 = ON_SLEEP / DIO9 <br> • bit 10 = RSSI/DIO10 <br> • bit 11 = PWM/DIO11 <br> • bit 12 = CD/DIO12 <br><br> For example, a digital channel mask of 0x002F means DIO0,1,2,3, and 5 are enabled as digital IO. |
|---|---|---|
| 1 | Analog Channel Mask | Indicates which lines have analog inputs enabled for sampling. Each bit in the analog channel mask corresponds to one analog input channel. <br> • bit 0 = AD0/DIO0 <br> • bit 1 = AD1/DIO1 <br> • bit 2 = AD2/DIO2 <br> • bit 3 = AD3/DIO3 <br> • bit 4 = AD4/DIO4 <br> • bit 5 = ASSOC/AD5/DIO5 |
| Variable | Sampled Data Set | If any digital IO lines are enabled, the first two bytes of the data set indicate the state of all enabled digital IO. Only digital channels that are enabled in the Digital Channel Mask bytes have any meaning in the sample set. If no digital IO are enabled on the device, these 2 bytes will be omitted. <br> Following the digital IO data (if any), each enabled analog channel will return 2 bytes. The data starts with AIN0 and continues sequentially for each enabled analog input channel up to AIN5. |

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

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

## Periodic I/O Sampling

Periodic sampling allows an XBee-PRO module to take an I/O sample and transmit it to a remote device at a periodic rate. The periodic sample rate is set by the IR command. If IR is set to 0, periodic sampling is disabled. For all other values of IR, data will be sampled after IR milliseconds have elapsed and transmitted to a remote device. The DH and DL commands determine the

destination address of the IO samples. Only devices with API mode enabled will send IO data samples out their UART. Devices not in API mode will discard received IO data samples.

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

## Digital I/O Change Detection

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



**Enabling Edge Detection will force an immediate sample of all monitored digital IO lines if any digital IO lines change state.**
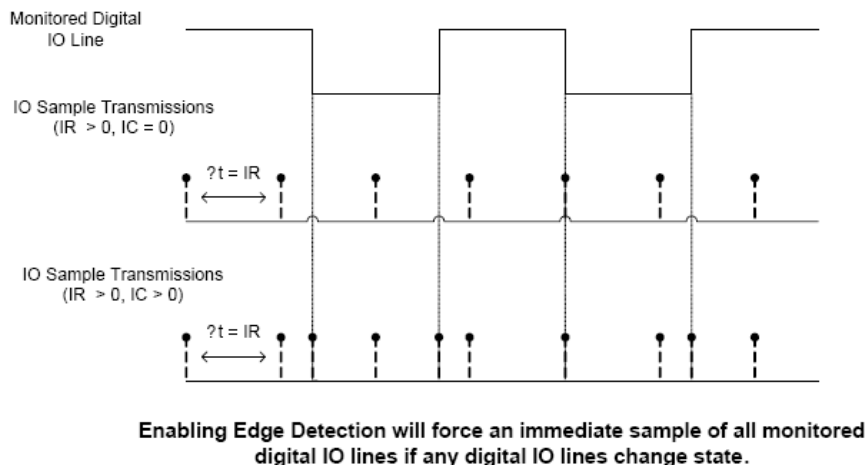
# 4. Sleep Mode

A number of low-power modes exist to enable modules to operate for extended periods of time on battery power. These sleep modes are enabled with the SM command.

## Sleep Modes

### Normal Mode (SM=0)

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

### Asynchronous Pin Sleep Mode (SM=1)

Pin sleep allows the module to sleep and wake according to the state of the **Sleep_RQ** pin (pin 9). Pin sleep mode is enabled by setting the SM command to 1. When **Sleep_RQ** is asserted (high), the module will finish any transmit or receive operations and enter a low-power state. The module will wake from pin sleep when the **Sleep_RQ** pin is de-asserted (low). Upon waking up, a pin sleep module will begin polling its parent as described in the Indirect Messaging and Polling section.

### Asynchronous Cyclic Sleep Mode (SM=4)

Cyclic sleep allows the module to sleep for a specified time and wake for a short time to poll its parent for any buffered data messages before returning to sleep again. Cyclic sleep mode is enabled by setting the SM command to 4. In cyclic sleep, the module sleeps for a specified time. If the XBee receives serial or RF data while awake, it will then extend the time before it returns to sleep by the amount specified by the ST command. Otherwise, it will enter sleep mode immediately. The **On_SLEEP** line is asserted (high) when the module wakes, and is de-asserted (low) when the module sleeps. If hardware flow control is enabled (D7 command), the **CTS** pin will assert (low) when the module wakes and can receive serial data, and de-assert (high) when the module sleeps.

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

(SM=5) is a slight variation on (SM=4) that allows the module to be woken prematurely by asserting the **Sleep_RQ** pin (pin 9). In (SM=5), the XBee can wake after the sleep period expires, or if a high-to-low transition occurs on the **Sleep_RQ** pin.

## Asynchronous Sleep Operation

### Wake Timer

In cyclic sleep mode (SM=4 or SM=5), if serial or RF data is received, the module will start a sleep timer (time until sleep). Any data received serially or by RF link will reset the timer. The timer duration can be set using the ST command. While the module is awake, it will send poll request transmissions every 100 ms to check its parent for buffered data. The module returns to sleep when the sleep timer expires.

## Indirect Messaging and Polling

To enable reliable communication with sleeping devices, the messaging mode command (CE) can be used to enable indirect messaging and polling.

## Indirect Messaging

Indirect messaging is a communication mode designed for communicating with asynchronous sleeping devices.  A module can enable indirect messaging by making itself an indirect messaging coordinator with the CE command.  An indirect messaging coordinator does not immediately transmit a unicast when it is received over the serial port.  Instead, the module holds on to the data until it is requested via a poll.  On receiving a poll, the indirect messaging coordinator will send out any data packets it may have queued up.

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

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

Indirect messaging has no effect on broadcasts.  Broadcasts are sent immediately when received over the serial port and are not put on the indirect messaging queue.

## Polling

Polling is the automatic process by which a node can request data from an indirect messaging coordinator.  Polling can be enabled on a device by configuring it as an end device with the CE command.  When polling is enabled, the module will send a poll request at least once every 100 ms.  When normal data is sent to the destination specified by the DH/DL of an end device module, that data will also function as a poll.

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

# 5. Command Reference Tables

## Special

**Special Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| WR | **Write**. Write parameter values to non-volatile memory so that parameter modifications persist through subsequent resets.<br>Note: Once WR is issued, no additional characters should be sent to the module until after the "OK\r" response is received. | -- | -- |
| RE | **Restore Defaults**. Restore module parameters to factory defaults. | -- | -- |
| FR | **Software Reset**. Reset module. Responds immediately with an "OK" then performs a reset 100ms later. | -- | -- |
| AC | **Apply Changes**. Immediately applies new settings without exiting command mode. | -- | -- |
| R1 | **Restore Compiled**. Restore module parameters to compiled defaults. | -- | -- |
| VL | **Version Long**. Shows detailed version information including application build date and time. | -- | -- |

## Addressing

**Addressing Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| DH | **Destination Address High**. Set/Get the upper 32 bits of the 64-bit destination address. When combined with DL, it defines the destination address used for transmission. | 0 to 0xFFFFFFFF | 0 |
| DL | **Destination Address Low**. Set/Get the lower 32 bits of the 64-bit destination address. When combined with DH, DL defines the destination address used for transmission. | 0 to 0xFFFFFFFF | 0x0000FFFF |
| DD | **Device Type Identifier.** Stores a device type value. This value can be used to differentiate multiple XBee-based products. | 0-0xFFFFFFFF<br>[ read only ] | 0x40000 |
| SH | **Serial Number High**. Read high 32 bits of the RF module's unique IEEE 64-bit address. 64-bit source address is always enabled. This value is read-only and it never changes | 0-0xFFFFFFFF | Factory |
| SL | **Serial Number Low**. Read low 32 bits of the RF module's unique IEEE 64-bit address. 64-bit source address is always enabled. This is read only and it is also the serial number of the node.. | 0-0xFFFFFFFF | Factory |
| HP | **Hopping Channel**.<br>Set/read the spread spectrum channel on which the module communicates. Separate channels minimize interference between multiple networks operating in the same vicinity. | 0-0x7 (standard variant)<br>0-0x3 (International variant) | 0 |
| SE | **Source Endpoint**. Set/read the application layer source endpoint value. This value will be used as the source endpoint for all data transmissions. The default value 0xE8 (Data endpoint) is the Digi data endpoint | 0-0xFF | 0xE8 |
| DE | **Destination Endpoint**.<br>Set/read application layer destination ID value. This value will be used as the destination endpoint for all data transmissions. The default value (0xE8) is the Digi data endpoint. | 0-0xFF | 0xE8 |
| CI | **Cluster Identifier**.<br>Set/read application layer cluster ID value. This value will be used as the cluster ID for all data transmissions. The default value 0x11 (Transparent data cluster ID) | 0-0xFFFF | 0x11 |
| NP | **Maximum RF Payload Bytes**.<br>This value returns the maximum number of RF payload bytes that can be sent in a unicast transmission based on the current configurations. | 0-0xFFFF | n/a |
| CE | **Coordinator/End Device.**<br>Set/read the messaging mode of the module.<br>0 - Normal mode<br>1 - Indirect messaging coordinator<br>2 - Polling device. | 0-2 | 0 |

## Serial Interfacing (I/O)

**Serial Interfacing Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| AP | **API mode**. Set or read the API mode of the radio. The following settings are allowed:<br>**0** API mode is off. All UART input and output is raw data and packets are delineated using the RO and RB parameters.<br>**1** API mode is on. All UART input and output data is packetized in the API format, without escape sequences.<br>**2** API mode is on with escaped sequences inserted to allow for control characters (XON, XOFF, escape, and the 0x7e delimiter to be passed as data.) | 0, 1, or 2 | 0 |
| AO | **API Output Format**. Enables different API output frames. Options include:<br>**0** Standard Data Frames (0x90 for RF RX)<br>**1** Explicit Addressing Data Frames (0x91 for RF RX) | 0, 1 | 0 |
| BD | **Baud rate**. Set or read serial interface rate (speed for data transfer between radio modem and host). Values from 0-8 select preset standard rates. Values at 0x39 and above select the actual baud rate. Providing the host supports it. Baud rates can go as high as 1.875Mbps. The values from 0 to 8 are interpreted as follows:<br>**0** - 1,200bps   **3** - 9,600bps   **6** - 57,600bps<br>**1** - 2,400bps   **4** - 19,200bps   **7** - 115,200bps<br>**2** - 4,800bps   **5** - 38,400bps   **8** - 230,400bps | 0 to 8, and 0x39 to 0x1c9c38 | 0x03 (9600 bps) |
| RO | **Packetization Timeout**. Set/Read number of character times of inter-character silence required before packetization. Set (RO=0) to transmit characters as they arrive instead of buffering them into one RF packet. | 0 - 0xFF [x character times] | 3 |
| FT | **Flow Control Threshhold**. Set or read flow control threshhold. De-assert CTS and/or send XOFF when FT bytes are in the UART receive buffer. Re-assert CTS when less than FT - 16 bytes are in the UART receive buffer. | 17 to 319 | 0x13F =319 |
| NB | **Parity**. Set or read parity settings for UART communications. The values from 0 to 4 are interpreted as follows:<br>**0** No parity    **3** Forced high parity<br>**1** Even parity  **4** Forced low parity<br>**2** Odd parity | 0 to 4 | 0 (No parity) |
| SB | **Stop Bits**. Set or read number of stop bits used for UART communications.<br>  0 - 1 stop bit    1 - 2 stop bits | 0 to 1 | 0 (1 stop bit) |
| D7 | **DIO7 Configuration**. Configure options for the DIO7 line of the module. Options include:<br>0 = Input, unmonitored<br>1 = CTS flow control<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high<br>6 = RS-485 Tx enable, low TX (0V on transmit, high when idle)<br>7 = RS-485 Tx enable, high TX (high on transmit, 0V when idle) | 0-1, 3-7 | 0 |
| D6 | **DIO6 Configuration**. Configure options for the DIO6 line of the module. Options include:<br>0 = Input, unmonitored<br>1 = RTS flow control<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0-1, 3-5 | 0 |

### I/O Commands

**I/O Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| P0 | **DIO10/PWM0 Configuration**. Configure options for the DIO10/PWM0 line of the module. Options include:<br>0 = Input, unmonitored<br>1 = RSSI<br>2 = PWM0<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0-5 | 1 |

**I/O Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| P1 | **DIO11/PWM1 Configuration**. Configure options for the DIO11/PWM1 line of the module. Options include:<br>0 = Input, unmonitored<br>2 = PWM1<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0, 2-5 | 0 |
| P2 | **DIO12 Configuration**. Configure options for the DIO12 line of the module. Options include:<br>0 = Input, unmonitored<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0, 3-5 | 0 |
| RP | **RSSI PWM Timer.** Time RSSI signal will be output after last transmission. When RP = 0xFF, output will always be on. | 0 - 0xFF [x 100 ms] | 2032 3.2 seconds) |
| D0 | **AD0/DIO0 Configuration**. Configure options for the AD0/DIO0 line of the module. Options include:<br>0 = Input, unmonitored<br>2 = Analog Input<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0,2-5 | 0 |
| D1 | **AD1/DIO1 Configuration**. Configure options for the AD1/DIO1 line of the module. Options include:<br>0 = Input, unmonitored<br>2 = Analog Input<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0, 2-5 | 0 |
| D2 | **AD2/DIO2 Configuration**. Configure options for the AD2/DIO2 line of the module. Options include:<br>0 = Input, unmonitored<br>2 = Analog Input<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0, 2-5 | 0 |
| D3 | **AD3/DIO3 Configuration**. Configure options for the AD3/DIO3 line of the module. Options include:<br>0 = Input, unmonitored<br>2 = Analog Input<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0, 2-5 | 0 |
| D4 | **AD4/DIO4 Configuration**. Configure options for the AD4/DIO4 line of the module. Options include:<br>0 = Input, unmonitored<br>2 = Analog Input<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0, 2-5 | 0 |
| D5 | **AD5/DIO5 Configuration**. Configure options for the AD5/DIO5 line of the module. Options include:<br>0 = Input, unmonitored<br>1 = Associate LED<br>2 = Analog Input<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0-5 | 1 |
| D8 | **DIO8/SLEEP_RQ Configuration**.<br>Configure options for the DIO8/SLEEP_RQ line of the module. Options include:<br>0 = Input, unmonitored<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high<br>When used as SLEEP_RQ, the D8 parameter should be configured in mode 0 or 3. | 0,3-5 | 0 |

**I/O Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| D9 | **DIO9 / ON/SLEEP.**<br>Configuration. Configure options for the DIO9/ON/SLEEP line of the module. Options include:<br>0 = Input, unmonitored<br>1 = ON/SLEEP<br>3 = Digital input, monitored<br>4 = Digital output low<br>5 = Digital output high | 0,1,3-5 | |
| PR | **Pull-up Resistor.** Set/read the bit field that configures the internal pull-up resistor status for the I/O lines. "1" specifies the pull-up resistor is enabled. "0" specifies no pullup.<br><br>Bit  I/O Line          Module pin  Range       Notes<br>0    DIO4/AD4        11          55k - 330k     1<br>1    DIO3/AD3        17          5k - 23k       2<br>2    DIO2/AD2        18          5k - 23k       2<br>3    DIO1/AD1        19          5k - 23k       2<br>4    DIO0/AD0        20          5k - 23k       2<br>5    DIO6/RTS        16          5k - 23k       2<br>6    DIO8/SLEEP_RQ/DTR  9         5k - 23k       2<br>7    DIN/Config       3           5k - 23k       2<br>8    DIO5/Associate    15          55k - 330k     1<br>9    DIO9/On/Sleep    13          5k - 23k       2<br>10   DIO12           4           5k - 23k       2<br>11   DIO10/PWM0/RSSI  6         5k - 23k       2<br>12   DIO11/PWM1     7          5k - 23k       2<br>13   DIO7/CTS       12          5k - 23k       2<br>14   DOUT            2           5k - 23k       2<br>**Note 1**: When set as a digital input with pull-up disabled, the leakage can be 9 uA in the worst case and 90nA in typical case when the line is set externally at a low level.<br>**Note 2**: When set as a digital input with pull-up enabled, the voltage of line will stabilize between Vcc-0.65V and Vcc-0.45V | 0 - 7FFF | 0x3F7F |
| M0 | **PWM0 Output Level**.<br>Set/read the output level of the PWM0 line. The line should be configured as a PWM output using the P0 command. | 0-0x03FF | 0 |
| M1 | **PWM1 Output Level**.<br>Set/read the output level of the PWM1 line. The line should be configured as a PWM output using the P1 command. | 0-0x03FF | 0 |
| LT | **Assoc LED Blink Time**.<br>Set/Read the Associate LED blink time. If the Associate LED functionality is enabled (D5 command), this value determines the on and off blink times for the LED. If LT=0, the default blink rate will be used (250ms). For all other LT values, LT is measured in 10ms | 0x14-0xFF (x10ms) | 0 |
| IS | **Force Sample**.<br>Forces a read of all enabled digital and analog input lines. | n/a | n/a |
| IC | **I/O Digital Change Detection**.<br>Set/Read the digital I/O pins to monitor for changes in the I/O state. IC works with the individual pin configuration commands (D0-D9, P0-P2). If a pin is enabled as a digital input/output, the IC command can be used to force an immediate I/O sample transmission when the DIO state changes. IC is a bitmask that can be used to enable or disable edge detection on individual channels. Unused bits should be set to 0.<br>Bit (I/O pin):<br> 0 (DIO0)<br>1 (DIO1)<br>2 (DIO2)<br>3 (DIO3)<br>4 (DIO4)<br>5 (DIO5)<br>6 (DIO6)<br>7 (DIO7)<br>8 (DIO8)<br>9 (DIO9)<br>10 (DIO10)<br>11 (DIO11)<br>12 (DIO12) | 0-0xFFFF | 0 |

**I/O Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| IR | **IO Sample Rate**. Set/Read the I/O sample rate to enable periodic sampling. For periodic sampling to be enabled, IR must be set to a non-zero value, and at least one module pin must have analog or digital I/O functionality enabled (see D0-D9, P0-P2 commands). The sample rate is measured in milliseconds. | 0 - 0xFFFF (ms) | 0 |
| CB | **Commissioning Pushbutton**. This command can be used to simulate commissioning button presses in software. The parameter value should be set to the number of button presses to be simulated. For example, sending the ATCB1 command will execute the action associated with 1 commissioning button press. | 0-4 | n/a |

## Diagnostics

**Diagnostics Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| VR | **Firmware Version**. Read firmware version of the module. | 0 - 0xFFFFFFFF [read-only] | Firmware-set |
| HV | **Hardware Version**. Read hardware version of the module. | 0 - 0xFFFF [read-only] | Factory-set |
| CK | **Configuration Code**. Read the configuration code associated with the current AT command configuration.The code returned can be used as a quick check to determine if a node has been configured as desired. | 0-0xFFFFFFFF | n/a |
| ER | **RF Errors**. Read the number of times a packet was received which contained integrity errors of some sort.  When the value reaches 0xFFFF, it stays there. | n/a | n/a |
| GD | **Good packets**. Read the number of good frames with valid MAC headers that are received on the RF interface. When the value reaches 0xFFFF, it stays there. | n/a | n/a |
| RP | **RSSI PWM timer**. Set or read the time that the RSSI output (indicating signal strength) will remain active after the last reception. Time units are measured in tenths of seconds | 1 to 0xff | 0x20 = 3.2 seconds |
| TR | **Transmission Errors**. Read the number of MAC frames that exhaust MAC retries without ever receiving a MAC acknowledgement message from the adjacent node. When the value reaches 0xffff, it stays there. | n/a | n/a |
| TP | **Temperature.** Read module temperature in Celsius. Negatives temperatures can be returned. | 0xff74 to 0x0258 | n/a |
| DB | **Received Signal Strength.** This command reports the received signal strength of the last received RF data packet. The DB command only indicates the signal strength of the last hop. It does not provide an accurate quality measurement for a multihop link. The DB command value is measured in -dBm. For example if DB returns 0x60, then the RSSI of the last packet received was -96dBm. An XBee-PRO 900 module will only report RSSI values within approximately 15 dBm of the sensitivity level of the module. Signals which exceed approximately -85dBm will be reported as approximately -85dBm. | n/a | n/a |
| %V | **Supply Voltage.** Reads the voltage on the Vcc pin in mV. Read module voltage in millivolts. | 0-0x/F00 | n/a |

## AT Command Options

**AT Command Options Commands**

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

## Node Identification

**Node Identification Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| ID | **Network ID**. Set or read the user network identifier. Nodes must have the same network identifier to communicate. Changes to ID can be written to non-volatile memory using the WR command. | 0x0000 to 0x7FFF | 0x7FFF |
| NT | **Node Discover Timeout.** Set/Read the amount of time a node will spend discovering other nodes when ND or DN is issued. | 0 - 0xFC [x 100 msec] | 0x82 (130d) |
| NI | **Node Identifier**. Stores a string identifier. The string accepts only printable ASCII data In AT Command Mode, the string can not start with a space. A carriage return or comma ends the command. Command will automatically end when maximum bytes for the string have been entered. This string is returned as part of the ATND (Network Discover) command. This identifier is also used with the ATDN (Destination Node) command. | up to 20 byte ASCII string | a space character |
| DN | **Discover Node - Destination Node**. Resolves an NI (Node Identifier) string to a physical address (case sensitive). The following events occur after the destination node is discovered:<br><br>\<AT Firmware\><br>1. DL & DH are set to the extended (64-bit) address of the module with the matching NI (Node Identifier) string.<br>2. OK (or ERROR)\r is returned.<br>3. Command Mode is exited to allow immediate communication<br>\<API Firmware\><br>0xFFFE and 64-bit extended addresses are returned in an API Command Response frame.<br>If there is no response from a module within (NT * 100) milliseconds or a parameter is not specified (left blank), the command is terminated and an "ERROR" message is returned. In the case of an ERROR, Command Mode is not exited. | 20 byte ascii string | |
| ND | Network Discover - Network Discover. Discovers and reports all RF modules found. The following information is reported for each module discovered.<br> MY\<CR\><br> SH\<CR\><br> SL\<CR\><br> NI\<CR\> (Variable length)<br> PARENT_NETWORK ADDRESS (2 Bytes)\<CR\><br> DEVICE_TYPE\<CR\> (1 Byte: 0=Coord, 1=Router, 2=End Device)<br> STATUS\<CR\> (1 Byte: Reserved)<br> PROFILE_ID\<CR\> (2 Bytes)<br> MANUFACTURER_ID\<CR\> (2 Bytes)<br> \<CR\><br>After (NT * 100) milliseconds, the command ends by returning a \<CR\>. ND also accepts a Node Identifier (NI) as a parameter (optional). In this case, only a module that matches the supplied identifier will respond. If ND is sent through the API, each response is returned as a separate AT_CMD_Response packet. The data consists of the above listed bytes without the carriage return delimiters. The NI string will end in a "0x00" null character. | | |

**Node Identification Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| NO | **Network Discovery Options**.<br>Set/Read the options value for the network discovery command. The options bitfield value can change the behavior of the ND (network discovery) command and/or change what optional values are returned in any received ND responses or API node identification frames.<br>Options include:<br>0x01 = Append DD value (to ND responses or API node identification frames)<br>0x02 = Local device sends ND response frame when ND is issued. | 0-0x03 [bitfield] | 0 |

## Security

**Security Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| EE | **Security Enable** Enables or disables 128-bit AES encryption. This command parameter should be set the same on all devices. | 0 to 1 | 0 |
| KY | **Security Key** Sets the 16 byte network security key value. This command is write-only. Attempts to read KY will return an OK status. This command parameter should be set the same on all devices. | 128-bit value | n/a |

## MAC Level

**MAC-level Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| MT | **Broadcast Multi-Transmit**. Set/Read the number of additional MAC-level broadcast transmissions. All broadcast packets are transmitted MT+1 times to ensure it is received. | 0-0xF | 3 |
| RR | **Unicast Mac Retries**. Set/Read the maximum number of MAC level packet delivery attempts for unicasts. If RR is non-zero packets sent from the radio will request an acknowledgement, and can be resent up to RR times if no acknowledgements are received. | 0-0xF | 10 |

## Sleep

**Sleep Commands**

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| SM | **Sleep Mode**.<br>Set/read the sleep mode of the module.<br>0 - No sleep mode enabled<br>1 - Pin sleep. In this mode, the sleep/wake state of the module is controlled by the SLEEP_RQ line.<br>4 - Asynchronous cyclic sleep. In this mode, the module periodically sleeps and wakes based on the SP and ST commands.<br>5 - Asynchronous cyclic sleep with pin wake-up. In this mode, the module acts in the same way as asynchronous cyclic sleep with the exception that the module will prematurely terminate a sleep period when a falling edge of the SLEEP_RQ line is detected. | 0, 1, 4, 5, | 0 |
| SO | **Sleep Options**.<br>Set/read the sleep options of the module. This command is a bitmask.<br>The following sleep options are defined:<br>bit 8 = Always wake for ST time | Any of the available sleep option bits can be set or cleared. | 0x00 |
| ST | **Wake Time**.<br>Set/read the wake period of the module.<br>This command defines the amount of time that the module will stay awake after receiving RF or serial data. | 0x45-0x36EE80 | 0x7D0 (2 seconds) |
| SP | **Sleep Period**.<br>Set/read the sleep period of the module. This command defines the amount of time the module will sleep per cycle.<br>For a node operating as an Indirect Messaging Coordinator, this command defines the amount of time that it will hold an indirect message for an end device. The coordinator will hold the message for (2.5 * SP). | 1 - 1440000 (x 10 ms) | 2 seconds |

| AT Command | Name and Description | Parameter Range | Default |
|---|---|---|---|
| SN | **Number of Sleep Periods.**<br>Set/read the number of sleep periods value. This command controls the number of sleep periods that must elapse between assertions of the ON_SLEEP line during the wake time of asynchronous cyclic sleep. During cycles when the ON_SLEEP line is not asserted, the module will wake up and check for any serial or RF data. If any such data is recieved, then the ON_SLEEP line will be asserted and the module will fully wake up. Otherwise, the module will return to sleep after checking. This command does not work with synchronous sleep modules. | 1 - 0xFFFF | 1 |
| WH | **Wake Host**.<br>Set/Read the wake host timer value.<br>If the wake host timer is set to a non-zero value, this timer specifies a time (in millisecond units) that the device should allow after waking from sleep before sending data out the UART or transmitting an I/O sample. If serial characters are received, the WH timer is stopped immediately. | 0-0xFFFF (x 1ms) | 0 |

# 6. API Operation

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

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

## API Frame 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:

**Figure 8-01. UART Data Frame Structure:**

| Start Delimiter (Byte 1) | Length (Bytes 2-3) | | Frame Data (Bytes 4-$n$) | Checksum (Byte n + 1) |
|---|---|---|---|---|
| 0x7E | MSB | LSB | API-specific Structure | 1 Byte |
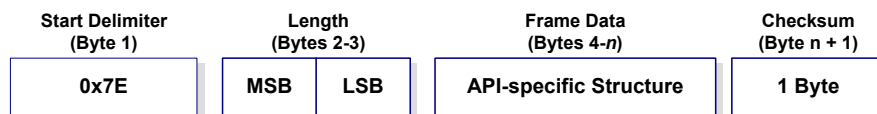
MSB = Most Significant Byte, LSB = Least Significant Byte

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:

**Figure 8-02. UART Data Frame Structure - with escape control characters:**

| Start Delimiter (Byte 1) | Length (Bytes 2-3) | | Frame Data (Bytes 4-n) | Checksum (Byte n + 1) |
|---|---|---|---|---|
| 0x7E | MSB | LSB | API-specific Structure | 1 Byte |

Characters Escaped If Needed

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 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:

**Figure 8-03. UART Data Frame & API-specific Structure:**
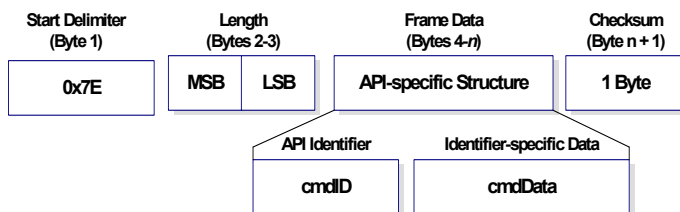


The cmdID frame (API-identifier) indicates which API messages will be contained in the cmdData frame (Identifier-specific data). Note that multi-byte values are sent big endian.The XBee modules support the following API frames:

**API Frame Names and Values**

| API Frame Names | API ID |
|---|---|
| AT Command | 0x08 |
| AT Command - Queue Parameter Value | 0x09 |
| Transmit Request | 0x10 |
| Explicit Addressing Command Frame | 0x11 |
| Remote Command Request | 0x17 |
| AT Command Response | 0x88 |
| Modem Status | 0x8A |
| Transmit Status | 0x8B |
| Receive Packet (AO=0) | 0x90 |
| Explicit Rx Indicator (AO=1) | 0x91 |
| ZigBee IO Data Sample RX Indicator | 0x92 |
| Node Identification Indicator (AO=0) | 0x95 |
| Remote Command Response | 0x97 |

**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 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.
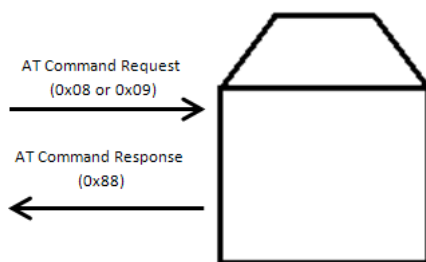


## Transmitting and Receiving RF Data

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. The received data frame (0x90 or 0x91) is set by the AP command.
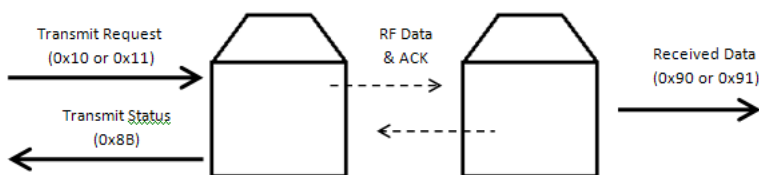


## Remote AT Commands

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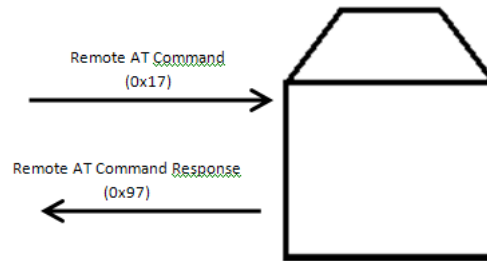
## Supporting the API

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

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

      case RX_IO_SAMPLE_FRAME:
         //process IO sample frame
         break;

      case NODE_IDENTIFICATION_FRAME:
         //process node identification frame
         break;

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

## Frame Data

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  value of the module

| | Frame Fields | | Offset | Example | Description |
|---|---|---|---|---|---|
| **A P I  P a c k e t** | **Start Delimiter** | | 0 | 0x7E | |
| | **Length** | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | LSB 2 | 0x04 | |
| | **Frame-specific Data** | **Frame Type** | 3 | 0x08 | |
| | | **Frame ID** | 4 | 0x52 (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 | | Command Name - Two ASCII characters that identify the AT Command. |
| | | | 6 | | |
| | | **Parameter Value (optional)** | | | If present, indicates the requested parameter value to set the given register. If no characters present, register is queried. |
| | **Checksum** | | 8 | 0x0F | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

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

## AT Command - Queue Parameter Value

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

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

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

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

## Transmit Request

Frame Type: 0x10

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

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

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

| Frame Fields | | Offset | Example | Description |
|---|---|---|---|---|
| Start Delimiter | | 0 | 0x7E | |
| Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | LSB 2 | 0x16 | |
| Frame-specific Data | Frame Type | 3 | 0x10 | |
| | 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 address is also supported: 0x000000000000FFFF - Broadcast address |
| | | 6 | 0x13 | |
| | | 7 | 0xA2 | |
| | | 8 | 0x00 | |
| | | 9 | 0x40 | |
| | | 10 | 0x0A | |
| | | 11 | 0x01 | |
| | | LSB 12 | 0x27 | |
| | Reserved | 13 | 0xFF | Set to 0xFFFE. |
| | | 14 | 0xFE | |
| | Broadcast Radius | 15 | 0x00 | Sets maximum number of hops a broadcast transmission can occur. If set to 0, the broadcast radius will be set to the maximum hops value. |
| | Transmit Options | 16 | 0x00 | Bitfield: bit 0: Disable ACK bit 1: Don't attempt route Discovery. All other bits must be set to 0. |
| | RF Data | 17 | 0x54 | Data that is sent to the destination device. The maximum number of payload bytes can be read with the NP command. |
| | | 18 | 0x78 | |
| | | 19 | 0x44 | |
| | | 20 | 0x61 | |
| | | 21 | 0x74 | |
| | | 22 | 0x61 | |
| | | 23 | 0x30 | |
| | | 24 | 0x41 | |
| Checksum | | 25 | 0x13 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

(The leftmost vertical label reads "API Packet")

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

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

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

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

## Explicit Addressing Command Frame

Frame Type: 0x11

Allows application layer fields (endpoint and cluster ID) to be specified for a data transmission. Similar to the Transmit Request, but also requires 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). For unicast transmissions the 64 bit address field should be set to the address of the desired destination node. The reserved field should be set to 0xFFFE.

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

| Frame Fields | | | Offset | Example | Description |
|---|---|---|---|---|---|
| Start Delimiter | | | 0 | 0x7E | |
| Length | | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x1A | |
| API Packet | 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 address is also supported:

0x000000000000FFFF - Broadcast address |
| | | | 6 | 0x13 | |
| | | | 7 | 0xA2 | |
| | | | 8 | 0x00 | |
| | | | 9 | 0x01 | |
| | | | 10 | 0x23 | |
| | | | 11 | 0x84 | |
| | | | LSB12 | 0x00 | |
| | | Reserved | 13 | 0xFF | Set to 0xFFFE. |
| | | | 14 | 0xFE | |
| | | Source Endpoint | 15 | 0xA0 | Source endpoint for the transmission. |
| | | Destination Endpoint | 16 | 0xA1 | Destination endpoint for the transmission. |
| | | Cluster ID | 17 | 0x15 | Cluster ID used in the transmission |
| | | | 18 | 0x54 | |
| | | Profile ID | 19 | 0xC1 | Profile ID used in the transmission |
| | | | 20 | 0x05 | |
| | | Broadcast Radius | 21 | 0x00 | Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value. |
| | | Transmit Options | 22 | 0x00 | Bitfield:
bit 0: Disable ACK
bit 1: Don't attempt route Discovery.
All other bits must be set to 0. |
| | | Data Payload | 23 | 0x54 | |
| | | | 24 | 0x78 | |
| | | | 25 | 0x44 | |
| | | | 26 | 0x61 | |
| | | | 27 | 0x74 | |
| | | | 28 | 0x61 | |
| Checksum | | | 29 | 0xDD | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

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

### Remote AT Command Request

Frame Type: 0x17

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

| | Frame Fields | | Offset | Example | Description |
|---|---|---|---|---|---|
| **A P I  P a c k e t** | **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 | 0x17 | |
| | | **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 address is also supported:  0x000000000000FFFF - Broadcast address |
| | | | 6 | 0x13 | |
| | | | 7 | 0xA2 | |
| | | | 8 | 0x00 | |
| | | | 9 | 0x40 | |
| | | | 10 | 0x40 | |
| | | | 11 | 0x11 | |
| | | | LSB 12 | 0x22 | |
| | | **Reserved** | 13 | 0xFF | Set to 0xFFFE. |
| | | | 14 | 0xFE | |
| | | **Remote Command Options** | 15 | 0x02 (apply changes) | 0x02 - Apply changes on remote. (If not set, AC command must be sent before changes will take effect.) All other bits must be set to 0. |
| | | **AT Command** | 16 | 0x42 (B) | Name of the command |
| | | | 17 | 0x48 (H) | |
| | | **Command Parameter** | 18 | 0x01 | If present, indicates the requested parameter value to set the given register. If no characters present, the register is queried. |
| | **Checksum** | | 18 | 0xF5 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

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

## AT Command Response

Frame Type: 0x88
In response to an AT Command message, the module will send an AT Command Response message. Some commands will send back multiple frames (for example, the ND (Node Discover) command).

| | Frame Fields | | Offset | Example | Description |
|---|---|---|---|---|---|
| **A P I P a c k e t** | **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 AT Command Mode, no AT Command Response will be given. |
| | | **AT Command** | 5 | 'B' = 0x42 | Command Name - Two ASCII characters that identify the AT Command. |
| | | | 6 | 'D' = 0x44 | |
| | | **Command Status** | 7 | 0x00 | 0 = OK<br>1 = ERROR<br>2 = Invalid Command<br>3 = Invalid Parameter |
| | | **Command Data** | | | Register data in binary format. If the register was set, then this field is not returned, as in this example. |
| | **Checksum** | | 8 | 0xF0 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

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

## Modem Status

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

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

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

## Transmit Status

Frame Type: 0x8B

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

| | Frame Fields | | Offset | Example | Description |
|---|---|---|---|---|---|
| **A P I  P a c k e t** | **Start Delimiter** | | 0 | 0x7E | |
| | **Length** | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x07 | |
| | **Frame-specific Data** | **Frame Type** | 3 | 0x8B | |
| | | **Frame ID** | 4 | 0x47 | Identifies the UART data frame being reported. Note: If Frame ID = 0 in AT Command Mode, no AT Command Response will be given. |
| | | **Reserved** | 5 | 0xFF | Reserved. |
| | | | 6 | 0xFE | |
| | | **Transmit Retry Count** | 7 | 0x00 | The number of application transmission retries that took place. |
| | | **Delivery Status** | 8 | 0x00 | 0x00 = Success<br>0x01 = MAC ACK Failure<br>0x15 = Invalid destination endpoint<br>0x21 = Network ACK Failure<br>0x25 = Route Not Found |
| | | **Discovery Status** | 9 | 0x02 | 0x00 = No Discovery Overhead<br>0x02 = Route Discovery |
| | **Checksum** | | 10 | 0x2E | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

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

## Receive Packet

Frame Type: (0x90)

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

| | Frame Fields | | Offset | Example | Description |
|---|---|---|---|---|---|
| **A P I  P a c k e t** | **Start Delimiter** | | 0 | 0x7E | |
| | **Length** | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x12 | |
| | **Frame-specific Data** | **Frame Type** | 3 | 0x90 | |
| | | **64-bit Source Address** | MSB 4 | 0x00 | 64-bit address of sender |
| | | | 5 | 0x13 | |
| | | | 6 | 0xA2 | |
| | | | 7 | 0x00 | |
| | | | 8 | 0x40 | |
| | | | 9 | 0x52 | |
| | | | 10 | 0x2B | |
| | | | LSB 11 | 0xAA | |
| | | **Reserved** | 12 | 0xFF | Reserved |
| | | | 13 | 0xFE | |
| | | **Receive Options** | 14 | 0x01 | 0x01 - Packet Acknowledged  0x02 - Packet was a broadcast packet |
| | | **Received Data** | 15 | 0x52 | Received RF data |
| | | | 16 | 0x78 | |
| | | | 17 | 0x44 | |
| | | | 18 | 0x61 | |
| | | | 19 | 0x74 | |
| | | | 20 | 0x61 | |
| | **Checksum** | | 21 | 0x11 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

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

### Explicit Rx Indicator

Frame Type:0x91

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

| Frame Fields | | | Offset | Example | Description |
|---|---|---|---|---|---|
| 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 |
| | | | 5 | 0x13 | |
| | | | 6 | 0xA2 | |
| | | | 7 | 0x00 | |
| | | | 8 | 0x40 | |
| | | | 9 | 0x52 | |
| | | | 10 | 0x2B | |
| | | | LSB 11 | 0xAA | |
| | Reserved | | 12 | 0xFF | Reserved. |
| | | | 13 | 0xFE | |
| | Source Endpoint | | 14 | 0xE0 | Endpoint of the source that initiated the transmission |
| | Destination Endpoint | | 15 | 0xE0 | Endpoint of the destination the message is addressed to. |
| | Cluster ID | | 16 | 0x22 | Cluster ID the packet was addressed to. |
| | | | 17 | 0x11 | |
| | Profile ID | | 18 | 0xC1 | Profile ID the packet was addressed to. |
| | | | 19 | 0x05 | |
| | Receive Options | | 20 | 0x02 | 0x01 – Packet Acknowledged<br>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 | 0x56 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

The API Packet spans the Frame Type through Checksum fields.

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

## IO Data Sample Rx Indicator

Frame Type:0x92

When the module receives an IO sample frame from a remote device, it sends the sample out the UART using this frame type (when AO=0). Only modules running API firmware will send IO samples out the UART.

| | Frame Fields | | Offset | Example | Description |
|---|---|---|---|---|---|
| **A P I  P a c k e t** | **Start Delimiter** | | 0 | 0x7E | |
| | **Length** | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x14 | |
| | **Frame-specific Data** | **Frame Type** | 3 | 0x92 | |
| | | **64-bit Source Address** | MSB 4 | 0x00 | 64-bit address of sender |
| | | | 5 | 0x13 | |
| | | | 6 | 0xA2 | |
| | | | 7 | 0x00 | |
| | | | 8 | 0x40 | |
| | | | 9 | 0x52 | |
| | | | 10 | 0x2B | |
| | | | LSB 11 | 0xAA | |
| | | **Reserved** | 12 | 0xFF | Reserved |
| | | | 13 | 0xFE | |
| | | **Receive Options** | 14 | 0x01 | 0x01 - Packet acknowledged<br>0x02 - Packet was a broadcast packet |
| | | **Number of Samples** | 15 | 0x01 | Number of sample sets included in the payload (Always set to 1) |
| | | **Digital Channel Mask** | 16 | 0x00 | Bitmask field that indicates which digital IO lines on the remote have sampling enabled (if any). |
| | | | 17 | 0x1C | |
| | | **Analog Channel Mask** | 18 | 0x02 | Bitmask field that indicates which analog IO lines on the remote have sampling enabled (if any). |
| | | **Digital Samples (if included)** | 19 | 0x00 | If the sample set includes any digital IO lines (Digital Channel Mask > 0), these two bytes contain samples for all enabled digital IO lines. DIO lines that do not have sampling enabled return 0. Bits in these two bytes map the same as they do in the Digital Channel Mask field. |
| | | | 20 | 0x14 | |
| | | **Analog Sample** | 21 | 0x02 | If the sample set includes any analog input lines (Analog Channel Mask > 0), each enabled analog input returns a 2-byte value indicating the A/D measurement of that input. Analog samples are ordered sequentially from AD0/DIO0 to AD3/DIO3, to the supply voltage. |
| | | | 22 | 0x25 | |
| | **Checksum** | | 23 | 0xF9 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

**Example:** Suppose an IO sample is received with analog and digital IO, from a remote with a 64-bit address of 0x0013A200 40522BAA. If pin AD1/DIO1 is enabled as an analog input, AD2/DIO2 and DIO4 are enabled as a digital inputs (currently high), and AD3/DIO3 is enabled as a digital output (low) the IO sample is shown in the API example in the table above.

## Node Identification Indicator

Frame Type:0x95

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

| | Frame Fields | | Offset | Example | Description |
|---|---|---|---|---|---|
| | Start Delimiter | | 0 | 0x7E | |
| | Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | | LSB 2 | 0x20 | |
| | | Frame Type | 3 | 0x95 | |
| | | 64-bit Source Address | MSB 4 | 0x00 | 64-bit address of sender |
| | | | 5 | 0x13 | |
| | | | 6 | 0xA2 | |
| | | | 7 | 0x00 | |
| | | | 8 | 0x40 | |
| | | | 9 | 0x52 | |
| | | | 10 | 0x2B | |
| | | | LSB 11 | 0xAA | |
| A P I  P a c k e t | Frame-specific Data | Reserved | MSB 12 | 0xFF | Reserved |
| | | | LSB 13 | 0xFE | |
| | | Receive Options | 14 | 0x02 | 0x01 - Packet Acknowledged<br>0x02 - Packet was a broadcast packet |
| | | Reserved | 15 | 0xFF | Reserved. |
| | | | 16 | 0xFE | |
| | | 64-bit Network address | 17 | 0x00 | Indicates the 64-bit address of the remote module that transmitted the node identification frame. |
| | | | 18 | 0x13 | |
| | | | 19 | 0xA2 | |
| | | | 20 | 0x00 | |
| | | | 21 | 0x40 | |
| | | | 22 | 0x52 | |
| | | | 23 | 0x2B | |
| | | | 24 | 0xAA | |
| | | NI String | 25 | 0x20 | Node identifier string on the remote device. The NI-String is terminated with a NULL byte (0x00). |
| | | | 26 | 0x00 | |
| | | Reserved | 27 | 0xFF | Reserved. |
| | | | 28 | 0xFE | |
| | | Device Type | 29 | 0x01 | 0 = Coordinator<br>1 = Normal Node<br>2 = End Device |
| | | Source Event | 30 | 0x01 | 1 = Frame sent by node identification pushbutton event (see D0 command) |
| | | Digi Profile ID | 31 | 0xC1 | Set to Digi's application profile ID. |
| | | | 32 | 0x05 | |
| | | Digi Manufacturer ID | 33 | 0x10 | Set to Digi's manufacturer ID. |
| | | | 34 | 0x1E | |
| | Checksum | | 35 | 0x23 | 0xFF - the 8-bit sum of bytes from offset 3 to this byte |

**Example:** If the commissioning push button is pressed on a remote router device with 64-bit address 0x0013A200 40522BAA and default NI string, the following node identification indicator would be received.

## Remote Command Response

Frame Type: 0x97

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

| Frame Fields | | Offset | Example | Description |
|---|---|---|---|---|
| Start Delimiter | | 0 | 0x7E | |
| Length | | MSB 1 | 0x00 | Number of bytes between the length and the checksum |
| | | LSB 2 | 0x13 | |
| Frame-specific Data | Frame Type | 3 | 0x97 | |
| | Frame ID | 4 | 0x55 | This is the same value passed in to the request. |
| | 64-bit Source (remote) Address | MSB 5 | 0x00 | The address of the remote radio returning this response. |
| | | 6 | 0x13 | |
| | | 7 | 0xA2 | |
| | | 8 | 0x00 | |
| | | 9 | 0x40 | |
| | | 10 | 0x52 | |
| | | 11 | 0x2B | |
| | | LSB 12 | 0xAA | |
| | Reserved | 13 | 0xFF | Reserved |
| | | 14 | 0xFE | |
| | AT Commands | 15 | 0x53 | Name of the command |
| | | 16 | 0x4C | |
| | Command Status | 17 | 0x00 | 0 = OK<br>1 = ERROR<br>2 = Invalid Command<br>3 = Invalid Parameter |
| | Command Data | 18 | 0x40 | The value of the required register |
| | | 19 | 0x52 | |
| | | 20 | 0x2B | |
| | | 21 | 0xAA | |
| Checksum | | 22 | 0xF4 | 0xFF - the 8 bit sum of bytes from offset 3 to this byte. |

(The left side of the table spans "API Packet" for rows from Frame Type onward.)

**Example:** If a remote command is sent to a remote device with 64-bit address 0x0013A200 40522BAA to query the SL command, and if the frame ID=0x55, the response would look like the above example.

# Appendix A: Definitions

**Definitions**

**Terms and Definitions**

| | |
|---|---|
| PAN | Personal Area Network - A data communication network that includes a coordinator and one or more routers/end devices. Synonymous with Network ID |
| Network Address | The 16-bit address is used for compatibility and has an address of 0xFFFE |

# Appendix B:  Agency Certifications

### United States FCC

The XBee-PRO® 900  RF Module complies with Part 15 of the FCC rules and regulations. Compliance with the labeling requirements, FCC notices and antenna usage guidelines is required.

To fufill 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-PRO® 900  RF 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-PRO® 900  RF Module

Contains FCC ID:MCQ-XBEE09P

The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (*i.*) this device may not cause harmful interference and (*ii.*) this device must accept any interference received, including interference that may cause undesired operation.

### FCC Notices

**IMPORTANT:** The XBee-PRO® 900 RF Module has been certified by the FCC for use with other products without any further certification (as per FCC section 2.1091). Modifications not expressly approved by Digi could void the user's authority to operate the equipment.

**IMPORTANT:** OEMs must test final product to comply with unintentional radiators (FCC section 15.107 & 15.109) before declaring compliance of their final product to Part 15 of the FCC Rules.

**IMPORTANT:** The RF module has been certified for remote and base radio applications. If the module will be used for portable applications, the device must undergo SAR testing.

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures: Re-orient or relocate the receiving antenna, Increase the separation between the equipment and receiver, Connect equipment and receiver to outlets on different circuits, or consult the dealer or an experienced radio/TV technician for help.

### FCC-Approved Antennas (900 MHz)

The XBee-PRO® 900 RF 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. 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-PRO® 900 have been tested and approved for use with all the antennas listed in chapter five.

**\* 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.

### Canada (IC)

Labeling requirements for Industry Canada are similar to those of the FCC. A clearly visible label on the outside of the final product enclosure must display the following text:

**Contains Model: XBEE09P, IC: 1846A-XBEE09P**

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.

**Transmitter Antennas**

This device has been designed to operate with the antennas listed in Appendix C, and having a maximum gain of 15.1 dB. Antennas not included in this list or having a gain greater than 15.1 dB are strictly prohibited for use with this device. The required antenna impedance is 50 ohms.

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.

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 permitted for successful communication.

### 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 import of the end product.

Contact Digi for questions related to locating a contact in Australia.

# Appendix C:  Antennas: 900 MHz

The following antennas have been approved for use with the XBee-PRO 900 RF module. Digi does not carry all of these antenna variants. Contact Digi Sales for available antennas.

| 900 MHz Antenna Listings | | | | |
|---|---|---|---|---|
| | | | | |
| | | | | |
| A09-F0 | Fiberglass Base | RPN | 0 dBi | Fixed |
| A09-F1 | Fiberglass Base | RPN | 1.0 dBi | Fixed |
| A09-F2 | Fiberglass Base | RPN | 2.1 dBi | Fixed |
| A09-F3 | Fiberglass Base | RPN | 3.1 dBi | Fixed |
| A09-F4 | Fiberglass Base | RPN | 4.1 dBi | Fixed |
| A09-F5 | Fiberglass Base | RPN | 5.1 dBi | Fixed |
| A09-F6 | Fiberglass Base | RPN | 6.1 dBi | Fixed |
| A09-F7 | Fiberglass Base | RPN | 7.1 dBi | Fixed |
| A09-F8 | Fiberglass Base | RPN | 8.1 dBi | Fixed |
| A09-F9 | Base Station | RPSMAF | 9.2 dBi | Fixed |
| A09-W7 | Wire Base Station | RPN | 7.1 dBi | Fixed |
| A09-F0 | Fiberglass Base | RPSMA | 0 dBi | Fixed |
| A09-F1 | Fiberglass Base | RPSMA | 1.0 dBi | Fixed |
| A09-F2 | Fiberglass Base | RPSMA | 2.1 dBi | Fixed |
| A09-F3 | Fiberglass Base | RPSMA | 3.1 dBi | Fixed |
| A09-F4 | Fiberglass Base | RPSMA | 4.1 dBi | Fixed |
| A09-F5 | Fiberglass Base | RPSMA | 5.1 dBi | Fixed |
| A09-F6 | Fiberglass Base | RPSMA | 6.1 dBi | Fixed |
| A09-F7 | Fiberglass Base | RPSMA | 7.1 dBi | Fixed |
| A09-F8 | Fiberglass Base | RPSMA | 8.1 dBi | Fixed |
| A09-M7 | Base Station | RPSMAF | 7.2 dBi | Fixed |
| A09-W7SM | Wire Base Station | RPSMA | 7.1 dBi | Fixed |
| A09-F0TM | Fiberglass Base | RPTNC | 0 dBi | Fixed |
| A09-F1TM | Fiberglass Base | RPTNC | 1.0 dBi | Fixed |
| A09-F2TM | Fiberglass Base | RPTNC | 2.1 dBi | Fixed |
| A09-F3TM | Fiberglass Base | RPTNC | 3.1 dBi | Fixed |
| A09-F4TM | Fiberglass Base | RPTNC | 4.1 dBi | Fixed |
| A09-F5TM | Fiberglass Base | RPTNC | 5.1 dBi | Fixed |
| A09-F6TM | Fiberglass Base | RPTNC | 6.1 dBi | Fixed |
| A09-F7TM | Fiberglass Base | RPTNC | 7.1 dBi | Fixed |
| A09-F8TM | Fiberglass Base | RPTNC | 8.1 dBi | Fixed |
| A09-W7TM | Wire Base Station | RPTNC | 7.1 dBi | Fixed |
| A09-HSM-7 | Straight half-wave | RPSMA | 3.0 dBi | Fixed / Mobile |
| A09-HASM-675 | Articulated half-wave | RPSMA | 2.1 dBi | Fixed / Mobile |
| A09-HABMM-P6I | Articulated half-wave | MMCX | 2.1 dBi | Fixed / Mobile |
| A09-HABMM-6-P6I | Articulated half-wave | MMCX | 2.1 dBi | Fixed / Mobile |
| A09-HBMM-P6I | Straight half-wave w/ | MMCX | 2.1 dBi | Fixed / Mobile |
| A09-HRSM | Right angle half-wave | RPSMA | 2.1 dBi | Fixed |
| A09-HASM-7 | Articulated half-wave | RPSMA | 2.1 dBi | Fixed |

| | | | | |
|---|---|---|---|---|
| A09-HG | Glass mounted half- | RPSMA | 2.1 dBi | Fixed |
| A09-HATM | Articulated half-wave | RPTNC | 2.1 dBi | Fixed |
| A09-H | Half-wave dipole | RPSMA | 2.1 dBi | Fixed |
| A09-HBMMP6I | 1/2 wave antenna | MMCX | 2.1dBi | Mobile |
| A09-QBMMP6I | 1/4 wave antenna | MMCX | 1.9 dBi | Mobile |
| A09-QI | 1/4 wave integrated wire antenna | Integrated | 1.9 dBi | Mobile |
| 29000187 | Helical | Integrated | -2.0 dBi | Fixed/Mobile |
| A09-QW | Quarter-wave wire | Permanent | 1.9 dBi | Fixed / Mobile |
| A09-QRAMM | 3 " Quarter-wave wire | MMCX | 2.1 dBi | Fixed / Mobile |
| A09-QSM-3 | Quarter-wave straight | RPSMA | 1.9 dBi | Fixed / Mobile |
| A09-QSM-3H | Heavy duty quarter- | RPSMA | 1.9 dBi | Fixed / Mobile |
| A09-QBMM-P6I | Quarter-wave w/ 6" | MMCX | 1.9 dBi | Fixed / Mobile |
| A09-QHRN | Miniature Helical Right | Permanent | -1 dBi | Fixed / Mobile |
| A09-QHSN | Miniature Helical Right | Permanent | -1 dBi | Fixed / Mobile |
| A09-QHSM-2 | 2" Straight | RPSMA | 1.9 dBi | Fixed / Mobile |
| A09-QHRSM-2 | 2" Right angle | RPSMA | 1.9 dBi | Fixed / Mobile |
| A09-QHRSM-170 | 1.7" Right angle | RPSMA | 1.9 dBi | Fixed / Mobile |
| A09-QRSM-380 | 3.8" Right angle | RPSMA | 1.9 dBi | Fixed / Mobile |
| A09-QAPM-520 | 5.2" Articulated Screw | Permanent | 1.9 dBi | Fixed / Mobile |
| A09-QSPM-3 | 3" Straight screw | Permanent | 1.9 dBi | Fixed / Mobile |
| A09-QAPM-3 | 3" Articulated screw | Permanent | 1.9 dBi | Fixed / Mobile |
| A09-QAPM-3H | 3" Articulated screw | Permanent | 1.9 dBi | Fixed / Mobile |
| A09-DPSM-P12F | omni directional | RPSMA | 3.0 dBi | Fixed |
| A09-D3NF-P12F | omni directional | RPN | 3.0 dBi | Fixed |
| A09-D3SM-P12F | omni directional w/ 12ft | RPSMA | 3.0 dBi | Fixed |
| A09-D3PNF | omni directional | RPN | 3.0 dBi | Fixed |
| A09-D3TM-P12F | omni directional w/ 12ft | RPTNC | 3.0 dBi | Fixed |
| A09-D3PTM | omni directional | RPTNC | 3.0 dBi | Fixed |
| A09-M0SM | Mag Mount | RPSMA | 0 dBi | Fixed |
| A09-M2SM | Mag Mount | RPSMA | 2.1 dBi | Fixed |
| A09-M3SM | Mag Mount | RPSMA | 3.1 dBi | Fixed |
| A09-M5SM | Mag Mount | RPSMA | 5.1 dBi | Fixed |
| A09-M7SM | Mag Mount | RPSMA | 7.1 dBi | Fixed |
| A09-M8SM | Mag Mount | RPSMA | 8.1 dBi | Fixed |
| A09-M0TM | Mag Mount | RPTNC | 0 dBi | Fixed |
| A09-M2TM | Mag Mount | RPTNC | 2.1 dBi | Fixed |
| A09-M3TM | Mag Mount | RPTNC | 3.1 dBi | Fixed |
| A09-M5TM | Mag Mount | RPTNC | 5.1 dBi | Fixed |
| A09-M7TM | Mag Mount | RPTNC | 7.1 dBi | Fixed |
| A09-M8TM | Mag Mount | RPTNC | 8.1 dBi | Fixed |
| | | | | |
| | | | | |
| A09-Y6 | 2 Element Yagi | RPN | 6.1 dBi | Fixed / Mobile |
| A09-Y7 | 3 Element Yagi | RPN | 7.1 dBi | Fixed / Mobile |
| A09-Y8 | 4 Element Yagi | RPN | 8.1 dBi | Fixed / Mobile |
| A09-Y9 | 4 Element Yagi | RPN | 9.1 dBi | Fixed / Mobile |
| A09-Y10 | 5 Element Yagi | RPN | 10.1 dBi | Fixed / Mobile |

| A09-Y11 | 6 Element Yagi | RPN | 11.1 dBi | Fixed / Mobile |
|---|---|---|---|---|
| A09-Y12 | 7 Element Yagi | RPN | 12.1 dBi | Fixed / Mobile |
| A09-Y13 | 9 Element Yagi | RPN | 13.1 dBi | Fixed / Mobile |
| A09-Y14 | 10 Element Yagi | RPN | 14.1 dBi | Fixed / Mobile |
| A09-Y14 | 12 Element Yagi | RPN | 14.1 dBi | Fixed / Mobile |
| A09-Y15 | 13 Element Yagi | RPN | 15.1 dBi | Fixed / Mobile |
| A09-Y15 | 15 Element Yagi | RPN | 15.1 dBi | Fixed / Mobile |
| A09-Y6TM | 2 Element Yagi | RPTNC | 6.1 dBi | Fixed / Mobile |
| A09-Y7TM | 3 Element Yagi | RPTNC | 7.1 dBi | Fixed / Mobile |
| A09-Y8TM | 4 Element Yagi | RPTNC | 8.1 dBi | Fixed / Mobile |
| A09-Y9TM | 4 Element Yagi | RPTNC | 9.1 dBi | Fixed / Mobile |
| A09-Y10TM | 5 Element Yagi | RPTNC | 10.1 dBi | Fixed / Mobile |
| A09-Y11TM | 6 Element Yagi | RPTNC | 11.1 dBi | Fixed / Mobile |
| A09-Y12TM | 7 Element Yagi | RPTNC | 12.1 dBi | Fixed / Mobile |
| A09-Y13TM | 9 Element Yagi | RPTNC | 13.1 dBi | Fixed / Mobile |
| A09-Y14TM | 10 Element Yagi | RPTNC | 14.1 dBi | Fixed / Mobile |
| A09-Y14TM | 12 Element Yagi | RPTNC | 14.1 dBi | Fixed / Mobile |
| A09-Y15TM | 13 Element Yagi | RPTNC | 15.1 dBi | Fixed / Mobile |
| A09-Y15TM | 15 Element Yagi | RPTNC | 15.1 dBi | Fixed / Mobile |

# Appendix D: Warranty Information

XBee RF Modules from Digi International, Inc. (the "Product") are warranted against defects in materials and workmanship under normal use, for a period of 1-year from the date of purchase. In the event of a product failure due to materials or workmanship, Digi will repair or replace the defective product. For warranty service, return the defective product to Digi International, shipping prepaid, for prompt repair or replacement.

The foregoing sets forth the full extent of Digi International's warranties regarding the Product. Repair or replacement at Digi International's option is the exclusive remedy. THIS WARRANTY IS GIVEN IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, AND DIGI SPECIFICALLY DISCLAIMS ALL WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL DIGI, ITS SUPPLIERS OR LICENSORS BE LIABLE FOR DAMAGES IN EXCESS OF THE PURCHASE PRICE OF THE PRODUCT, FOR ANY LOSS OF USE, LOSS OF TIME, INCONVENIENCE, COMMERCIAL LOSS, LOST PROFITS OR SAVINGS, OR OTHER INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT, TO THE FULL EXTENT SUCH MAY BE DISCLAIMED BY LAW. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES. THEREFORE, THE FOREGOING EXCLUSIONS MAY NOT APPLY IN ALL CASES. This warranty provides specific legal rights. Other rights which vary from state to state may also apply.