XLR PRO™

Radio Frequency (RF) Module

User Guide

Revision history—90001407

Revision	Date	Description
P1	May 2014	Initial release released for certifications.
A	October 2015	Updated to include full descriptions of all XLR PRO RF Module features.
B	October 2016	Added several I/O settings commands. Added a getting started section for the development kit. Added mechanical drawings. Revised the AT command and API frame sections. Added Australian certification information.
C	November 2016	Added information about heat dissipation and using a heat sink.

Trademarks and copyright

Disclaimers

Information in this document is subject to change without notice and does not represent a commitment on the part of Digi International. Digi provides this document “as is,” without warranty of any kind, expressed or implied, including, but not limited to, the implied warranties of fitness or merchantability for a particular purpose. Digi may make improvements and/or changes in this manual or in the product(s) and/or the program(s) described in this manual at any time.

Warranty

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

www.digi.com/howtobuy/terms

Send comments

Documentation feedback: To provide feedback on this document, send your comments to techcomm@digi.com.

Customer support

Digi Technical Support: Digi offers multiple technical support plans and service packages to help our customers get the most out of their Digi product. For information on Technical Support plans and pricing, contact us at +1 952.912.3444 or visit us at www.digi.com/support.

Support portal login: www.digi.com/support/eservice

Contents

About the XLR PRO RF Module

Technical specifications

Getting started with the XLR PRO RF Module

Operations

Networking methods

AT commands

Operate in API mode

Advanced application features

Configure the XLR PRO RF Module using XCTU

Safety notices and certifications

XLR PRO antennas

About the XLR PRO RF Module

The XLR PRO RF Module is a high performance, industrial grade, long-range radio solution that ensures reliable wireless data communications over long distances.

Using patent-pending Punch2™ Technology to maximize range and significantly increase immunity to interference, the XLR PRO 900 MHz radio can connect a variety of devices across many industrial applications.

Punch2 Technology leverages chirp spread spectrum (CSS) modulation to provide better receiver sensitivity, multipath performance, and interference rejection than is available through commonly used frequency hopping spread spectrum (FHSS) or direct sequence spread spectrum (DSSS) systems. The advantages of Punch2 Technology arise from the characteristics of the chirp signal as well as several digital-signal-processing techniques that enhance performance and reliability. With Punch2 Technology, data is spread to a higher bandwidth by multiplying each transmit modulation symbol with a chirp signal. Operating at an expanded bandwidth provides several benefits:

- Greater receiver sensitivity
- Interference immunity
- Improved multipath performance
- Adjustable data rates

The XLR PRO RF Module flexible configuration and management options allow you to quickly set up and deploy one or more XLR PRO RF Modules, as well as apply firmware updates, get device status information, and more.

Data rates and sensitivity

The following table lists the available data rates along with the corresponding receiver sensitivity.

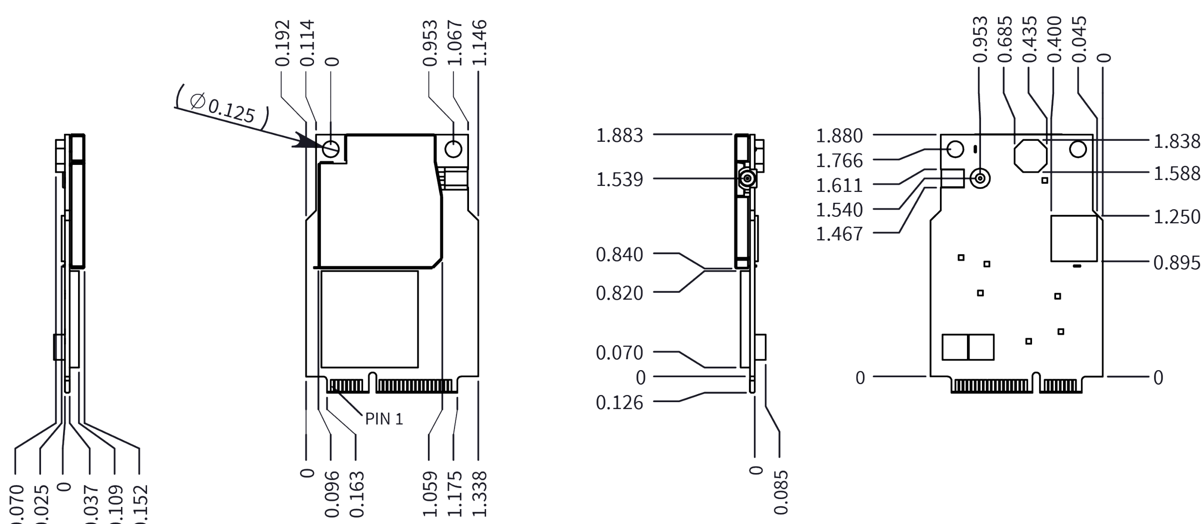
RF data rate setting (BR command)	Data rate	Receiver sensitivity (dBm, 25 °C)
0	9.4 kb/s	-120
1	28 kb/s	-118
2	66 kb/s	-116
3	141 kb/s	-112
4	291 kb/s	-109
5	591 kb/s	-106
6	1.2 Mb/s	-103
7	2.4 Mb/s	-100
8	3.2 Mb/s	-98

Antenna port

The antenna port is a 50 Ω MMCX RF signal connector for connecting to an external antenna.

Mechanical drawings

The following figures show the XLR PRO RF Module mechanical drawings. All dimensions are in inches.



Pinouts

The following table shows the pin assignments for the XLR PRO RF Module. In the table, low-asserted signals have a horizontal line above signal name.

Pin number	Name	Default direction	Notes
1	—	—	do not connect
2	—	—	do not connect
3	—	—	do not connect
4	—	—	do not connect
5	GND	—	
6	VDD_5V	Source input	
7	—	—	do not connect
8	—	—	do not connect
9	—	—	do not connect
10	—	—	do not connect
11	VDD_5V	Source input	
12	GND	—	
13	3.3V_OUT	Source output	Output of the regulator is defaulted to 3.3 V. Intended use is for powering 3.3V_IN. VDD_5V is required even if an external regulator is used to power 3.3V_IN instead of 3.3V_OUT.
14	3.3V_IN	Source input	This powers the IO_RING determining I/O voltage levels.
15	3.3V_ADJUST	Input	3.3V_OUT can be adjusted to a lower voltage level by connecting through a resistor to 3.3V_IN.
16	$\overline{\text{RESET}}$	Input	
17	—	—	do not connect
18	DIO0/AD0/Comm Btn	Both	
19	—	—	do not connect
20	DIO1/AD1	Both	
21	—	—	do not connect
22	DIO2/AD2	Both	
23	—	—	do not connect
24	DIO3/AD3	Both	

Pin number	Name	Default direction	Notes
25	—	—	do not connect
26	DIO4	Both	
27	—	—	do not connect
28	DIO5/ASSOC	Both	
29	DIO19/SPI_ATTN	Both (input)	
30	DIO8/DTR/SLP_RQ	Both	
31	DIO17/SPI_SSEL	Both	
32	DIO9/ON/SLEEP	Output	
33	DIO16/SPI_MOSI	Both	
34	VREF	Both	unused
35	DIO15/SPI_MISO	Both	
36	DIO10/PWM0/RSSI	Both (output)	
37	DIO18/SPI_CLK	Both	
38	DIO11/PWM1	Both	
39	DIO7/CTS	Both (output)	
40	DIO12	Both	
41	DIO6/RTS	Both (input)	
42	GND	—	
43	DIO13/DOUT	Output	
44	—	—	do not connect
45	DIO14/DIN/CONFIG	Input	
46	SHUTDOWN	Input	Controls 3.3V_OUT regulator. Connect to VDD_5V if the module should always be on. Connect to ground if 3.3V_OUT is not used.
47	GND	—	
48	VDD_5V	Source input	

Pin number	Name	Default direction	Notes
49	—	—	do not connect
50	—	—	do not connect
51	—	—	do not connect
52	—	—	do not connect

Electrical specifications

In this table, low-asserted signals have a horizontal line above signal name.

Electrical specifications	Minimum	Typical	Maximum	Units	Status
VDD_5V	3.8	5	5.5	V	Operation below 4.5 V may reduce radio transmit power performance and LNA sensitivity.
$\overline{\text{SHUTDOWN}}$ IL **Note** See the warning below.	-0.3		0.4	V	3.3 V_OUT Disabled
$\overline{\text{SHUTDOWN}}$ IH	1.1		VDD_5V +0.3	V	3.3 V_OUT Enabled
3.3V_OUT voltage	1.8	3.3	3.4	V	3.3 V_OUT = 3.3 V if 3.3 V_ADJUST is not connected
3.3V_OUT current			300	mA	Up to 50 mA may be drawn for other devices if a ferrite is used to keep noise off XLR supply
3.3V_ADJUST voltage		open	3.3V_OUT	V	3.3 V_ADJUST connected by resistor R to 3.3 V_OUT: 3.3 V_OUT = 0.5 V*(340k*(R+340k)/(340k+R+340k)/60.4k+1)
3.3V_IN voltage **Note** See the warning below.	2.4	3.3	3.5	V	
3.3V_IN current			80	mA	
General I/O inputs IL	-0.3		0.3 x (3.3V_IN)	V	Input low
General I/O inputs IH	0.7 x (3.3V_IN)		3.3V_IN + 0.3	V	Input high

Electrical specifications	Minimum	Typical	Maximum	Units	Status
Internal pull-up resistor		40k		Ω	
Internal pull-down resistor		40k		Ω	
General I/O outputs OL			0.05 x (3.3V_IN)	V	Output low
General I/O outputs OH	0.95 x (3.3V_IN)			V	Output high
Output source current			2	mA	
Output sink current			2	mA	
Maximum output current drawn on all GPIOs			48	mA	
VREF range	N/A		3.3V_IN	V	Not used

WARNING! In shutdown modes where 3.3V_IN voltage is turned off, ensure that all I/O lines do not have any voltage applied to them or parasitic powering may occur. I/O lines should never exceed 3.3V_IN + 0.3 V.

Heat dissipation

We recommend that you use a heat sink with the XLR PRO RF Module. The XLR PRO RF Module produces heat and when it transmits, it generates more heat. The device has over-temperature protection and automatically shuts down if it gets too hot. It is best to avoid this temperature and not exceed 90°C so that the application can control transmissions, and is not stopped due to overheating.

The device’s temperature is affected by:

- Transmitter duty-cycle
- Quality of heat sink
- Ambient temperature

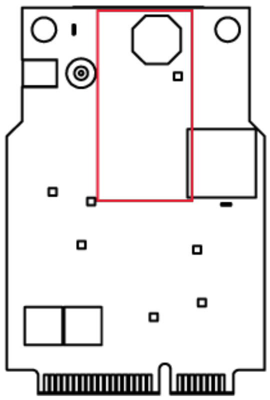
Continuous transmission with a good heat sink is possible up to an ambient temperature of approximately 75°C. The device's heat dissipation has to be tested on an application by application basis.

Use a heat sink

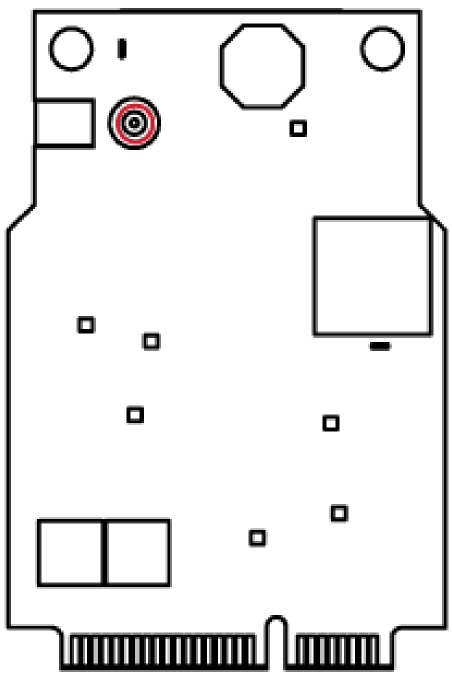
Use a heat sink to prevent the XLR PRO RF Module from overheating. The following mechanical drawing shows the heat sink that we use on the XLR Module Development Board:

<insert picture>

Your application can follow this design, or use a custom one. When implementing a custom design, we recommend using a gasket that covers at least the area covered by the XLR Module Development Board, highlighted in red in the following image:



Place the heat sink on the back side opposite to the shield, including the exposed rectangle and test point. Use non-conductive material whenever contacting the module to prevent shorting. Avoid the bulls-eye of the target, which is highlighted in red in the following image.



Use the TP (Temperature) command to query and measure the temperature on the module. This returns the temperature of the module in 8-bit two's compliment format. If the temperature in your application exceeds 90°C, re-evaluate the application design.

Methods to alleviate the heat are:

- Reduce the transmitter duty-cycle.
- Redesign the heat sink implementation.
- Add a fan.

Operation without a heat sink

Although we do not recommend it, you can operate the XLR PRO RF Module without a heat sink. This can only be done with an ambient temperature lower than 40°C, and with the transmitter being used no more than five seconds in a ten second period.

If you do not use a heat sink:

- Limit transmissions to fifty percent or less.
- The ambient temp must be less than 40°C.

For more information

The XLR PRO Radio Frequency (RF) family of products includes the following publications:

Title	Part number	Description
XLR PRO Radio Frequency (RF) Modem Quick Start Guide	90002204	Provides a brief summary of the XLR PRO and XLR PRO INTL Radio Frequency (RF) Modem kit.
XLR PRO Radio Frequency (RF) Modem Getting Started Guide	90002203	Provides step-by-step instructions for setting up a pair of XLR PRO (or XLR PRO INTL) modems to test over-the-air communications between the radios.
XLR PRO Radio Frequency (RF) Modem User Guide	90002202	Provides complete information on all XLR PRO and XLR PRO INTL Radio Frequency (RF) Modem features; describes how to configure XLR PRO modems using XCTU, the Web configuration interface, and Device Cloud; provides reference information on all supported AT commands and API frames.

Technical specifications

The following tables provide the device's technical specifications.

General

Specification	Value
Dimensions	34 x 51 x 6.1 mm (1.34 x 2.01 x .240 inches)
Weight	11 g (.39 oz)
RoHS	Compliant

Frequency range

Specification	Value
USA and Canada	ISM 902 to 928 MHz
Australia	ISM 915 to 928 MHz
RF data rate	9.4 kb/s to 3.2 Mb/s
Maximum transmit power (software selectable)	+30 dBm (1 W)

Rural range line-of-sight

Speed	Range
1.2 Mb/s	up to 100+ miles[1]

Receiver sensitivity

RF data rate	Sensitivity
9.4 kb/s	-120 dBm
141 kb/s	-112 dBm
3.2 Mb/s	-98 dBm

Receiver selectivity

RF data rate	Selectivity
141 kb/s	70 dB (below 908 MHz, above 922 MHz) 40 dB (908 MHz to 922 MHz)

[1]Based on 100-mile range results. Other data rates scale based on sensitivity levels. Results will vary based on noise levels and line of sight quality.

Interface data rate (software selectable)

UART interface	Serial data rate
UART	Up to 921.6 kb/s
SPI	Up to 6 Mb/s

Networking and security

Item	Specification
Modulation	Chirp Spread Spectrum
Supported network topologies	Point-to-point/point-to-multipoint
Encryption	128-bit AES

Power requirements

Item	Value	
Supply voltage	3.8 to 5.5 VDC (see Electrical specifications)	
Receive current	@ 5 VDC	295 mA typical
Transmit current	@ 5 VDC	1.58 A typical
Shutdown/power down current	3 µA typical at room temperature	

Environmental

Specification	Description
Operating temperature	-40° C to 85° C

Regulatory approvals

Regulation	Approval
United States	FCC ID: MCQ-XLRP
Canada	IC: 1846A-XLRP
Australia	RCM (pending)

Connectors

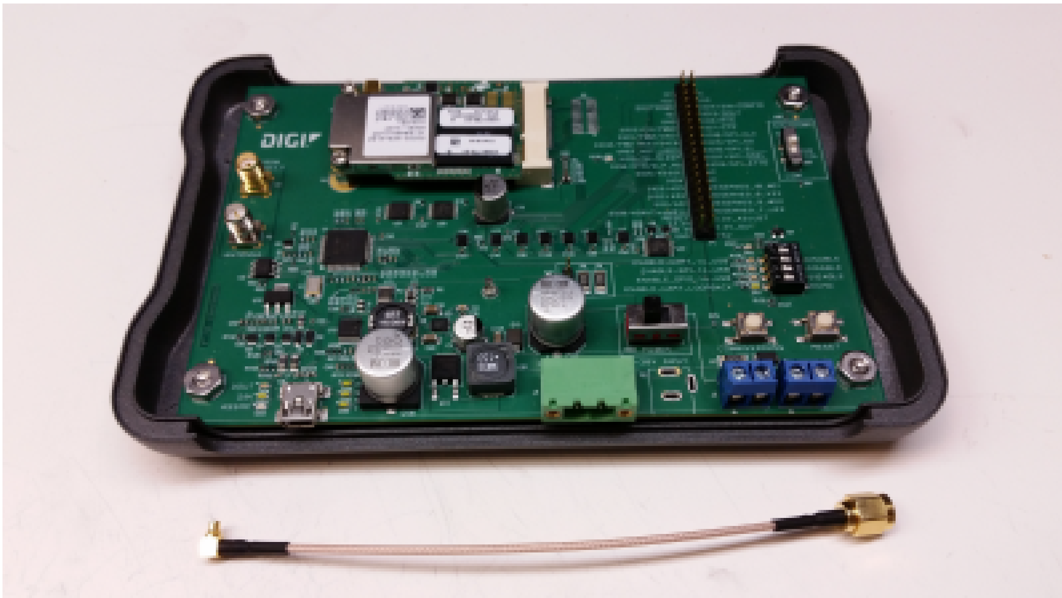
Connector	Description
Antenna	MMCX

Getting started with the XLR PRO RF Module

This section provides information on the Development Board for the XLR PRO RF Module and getting started instructions if you have an XLR PRO RF Module Development Kit.

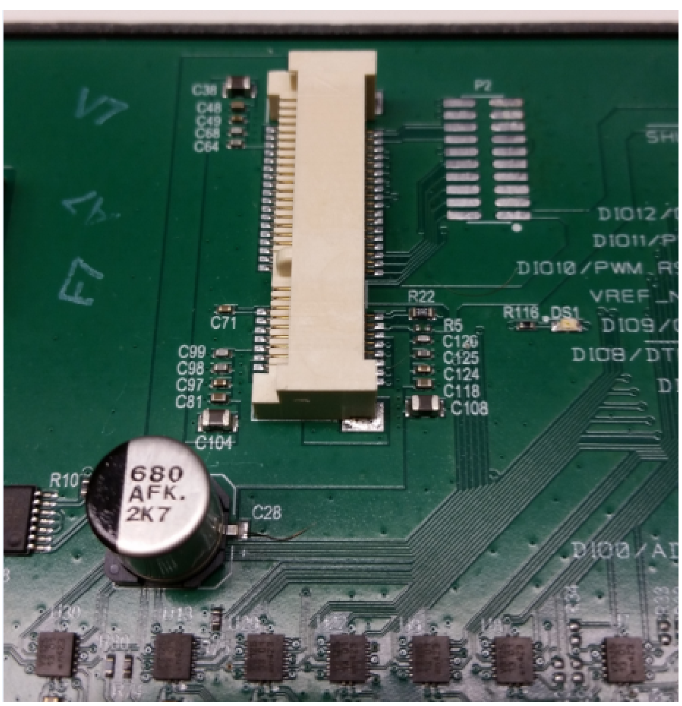
Development Board for the XLR PRO RF Module

The following figure shows the Development Board for the XLR PRO RF Module with an onboard XLR PRO RF Module.



Edge connector

The XLR PRO RF Module is connected to the development board by an edge connector. You must insert the board into the edge connector at a 30 degree angle before pressing down onto the mounting screws. Do not power the board when inserting a module.



CAUTION! The thermal gasket for the XLR PRO RF Module is subject to wear and tear. We recommend that you limit the number of insertions of the XLR PRO RF Module.

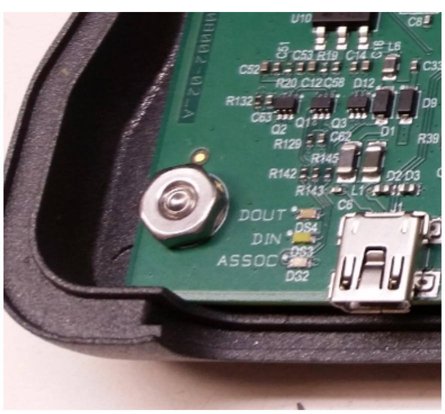
XLR PRO Modular Development Board LEDs

The board contains the following LEDs:

RSSI

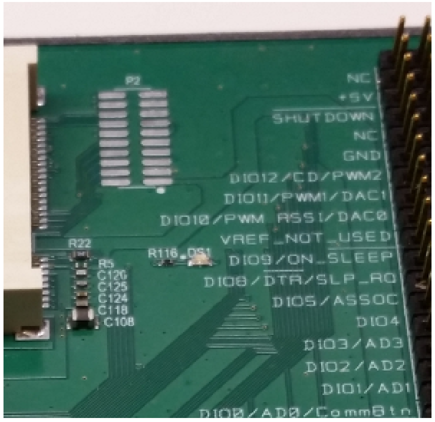
The RSSI LEDs display the fade margin present in an active wireless link. Unilluminated LEDs indicate a very weak or nonexistent signal. Three illuminated LEDs indicate a strong signal. The RSSI LEDs only illuminate for a few seconds after a valid RF packet is received.

DOUT/DIN/ASSOC



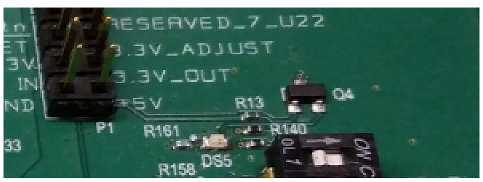
An illuminated yellow LED for DOUT indicates that DIO13/DOUT is low. An illuminated white LED for DIN indicates DIO14/DIN/CONFIG is low. An illuminated blue LED for ASSOC shows that DIO5/ASSOC is high.

DIO9/ON_$\overline{\text{SLEEP}}$



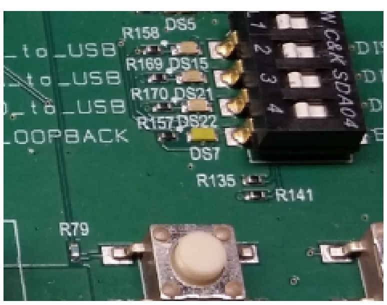
An illuminated blue LED indicates that the DIO9/ON_$\overline{\text{SLEEP}}$ line is high.

5V line indicator



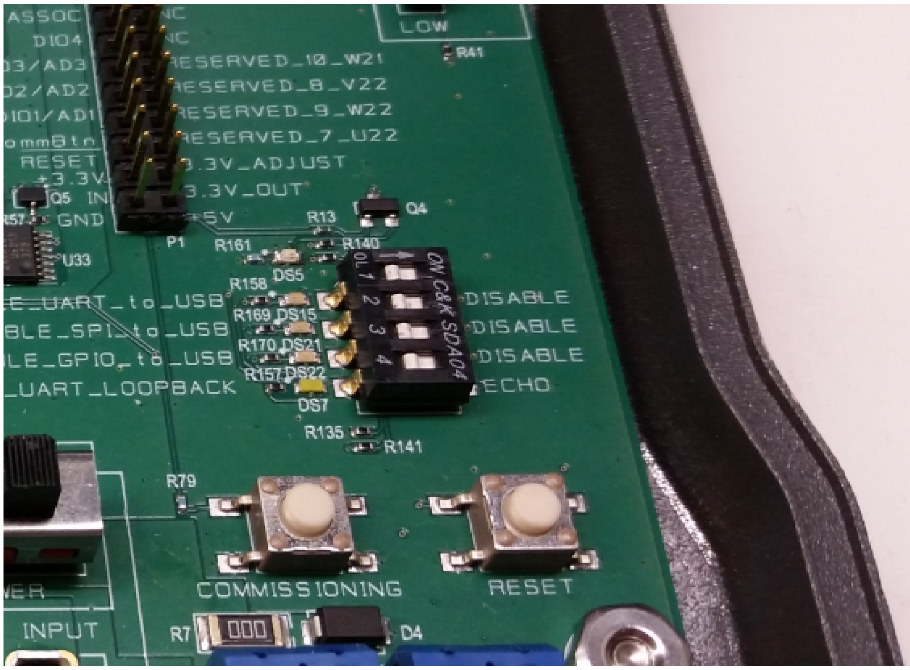
An illuminated red LED indicates that the 5 V line on the XLR PRO Module is powered. This light being illuminated does not ensure the voltage on VDD_5V line is correct, it only indicates that a voltage is present.

USB communication and loopback



Illuminated yellow LEDs indicate that UART, SPI, and GPIO lines are respectively enabled between the USB port and the XLR PRO Module. An illuminated white LED for the UART echo indicates that loopback in enabled.

USB communication enable and loopback switches



The USB communication enable switches are on the right side of the board, just above the COMMISSIONING and RESET buttons. Slide the DIP switches to the left to enable them (ENABLE_UART_LOOPBACK is enabled when pushed to the right). When enabled, they allow communication from the USB port to reach the XLR PRO module. The loopback switch loops back data sent from the USB port back out of the USB port.

ENABLE_UART_to_USB: switch allows UART communication from the USB port.

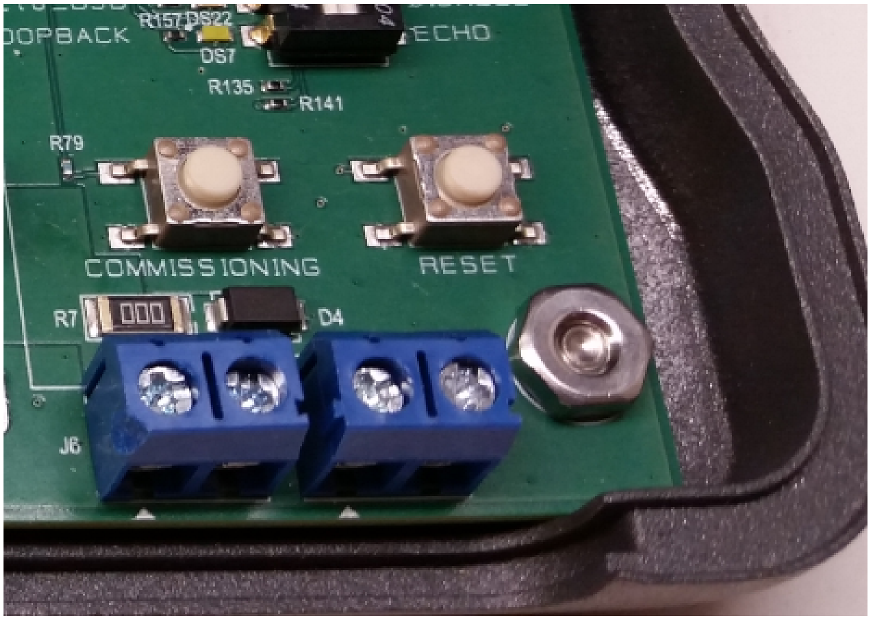
ENABLE_SPI_to_USB: switch allows SPI communication from the USB port.

ENABLE_GPIO_to_USB: switch allows GPIO communication from the USB port.

ENABLE_UART_LOOPBACK: switch allows loopback UART DIN sent from the USB port (UART DIN goes to UART DOUT). ENABLE_UART_LOOPBACK is enabled when the switch is slid to the right. If the switch is enabled and ENABLE_UART_to_USB is low, the loopback is disconnected from the XLR PRO module, but is still looped back to the USB port.

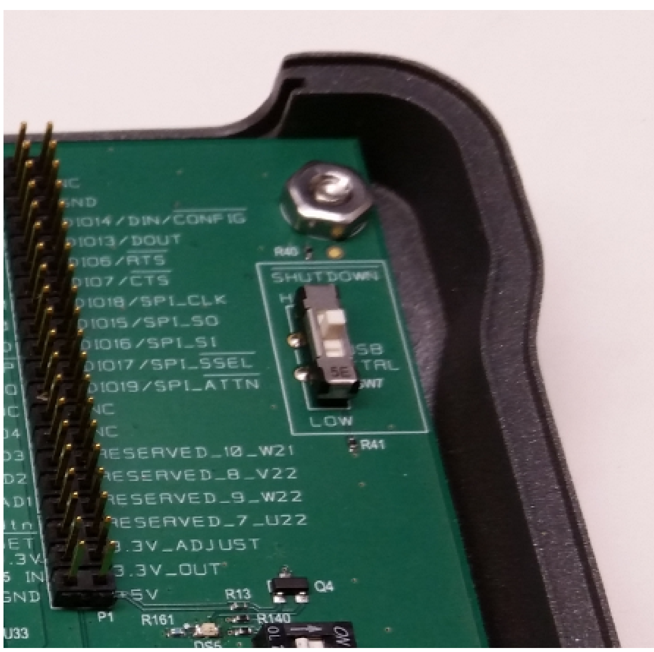
The LEDs next to the switches indicate whether the lines are enabled.

Commissioning and Reset buttons



Use the Commissioning and Reset buttons to toggle the $\overline{\text{RESET}}$ and DIO0/AD0/CommBtn lines to the XLR PRO module.

XLR PRO Shutdown control switch



Use the large three-state switch located near the top right corner of the board to control shutdown of the XLR PRO. It follows this logic table:

Switch position	FTDI_SEL_ON_WRITE	XLR_SHUTDOWN	Module 3.3V_OUT Enabled
High	X	High	Yes
Low	X	Low	No
Middle	Low	High	Yes
	High	Low	No

If the switch is high, the XLR PRO module is enabled. If the switch is low, the XLR PRO module is put into shutdown. The middle position could mean the XLR PRO module is enabled or in shutdown, depending on how a line is pulled on the USB bridging chip on the development board.

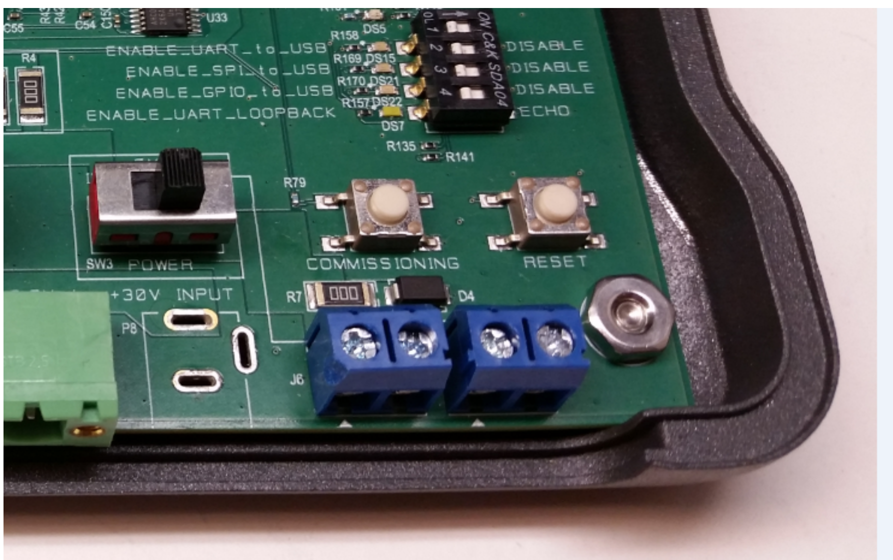
High: the XLR PRO module is enabled (not in shutdown mode).

Low: the XLR PRO module is disabled (in shutdown mode).

Middle: a state where shutdown mode for the XLR PRO module could be software selected.

FTDI pin 53 (signal name FTDI_SEL_ON_WRITE) is the line that determines whether or not shutdown mode is enabled. A pull up resistor makes the default value of this line high (XLR is in shutdown).

XLR PRO direct voltage supply input



CAUTION! When using the direct voltage supply input you risk damaging the XLR PRO module. Any voltage higher than 5.5 V risks damaging the XLR PRO module.

There are two ways to power the XLR PRO module on the development board: through the Phoenix connector or the terminal blocks. The green Phoenix connector powers an on-board regulator which regulates down to 5 V and is passed to the module. You can use the blue terminal blocks labeled J6 on the bottom right of the board to power the XLR PRO module directly. You can select the inputs using SW3. When SW3 is slid to the left, the Phoenix connector powers the XLR PRO module, and when it is slid to the right the terminal blocks power the XLR PRO module.

Powering the XLR PRO module from the terminal blocks allows a varied input voltage level to reach the device, since the Phoenix connector would only provide 5 V because of the regulator. It is important to follow the supply voltage requirements above when powering from the terminal blocks.

All power supplied on the terminal blocks is used to power the XLR PRO module only. Components on the XLR PRO Modular Development Board are not powered through the terminal blocks. Current measurements can be taken if powering through the terminal. Shutdown currents include current through D4 as well as the XLR module.

WARNING! Never exceed VDD_5V specifications.

40-Pin development header

CAUTION! When using the development header, note that GPIO and communication pins are not 5 V tolerant. Ensure that voltages do not exceed electrical specifications.

Use the development header (P1) to communicate directly with the XLR PRO module. When using this development header be sure to set all USB communication enable switches to the disabled state. This means on SW5, you should disable ENABLE_UART_to_USB, ENABLE_SPI_to_USB, and ENABLE_GPIO_to_USB. Also, when the XLR PRO module is in shutdown mode, ensure that no GPIO lines are being driven high.

Set up a Development kit for a range test

CAUTION! Make sure the board is not powered by either the USB or a power supply when you insert the XLR PRO module.

1. If no XLR PRO modules are present on the board, insert the XLR PRO modules into the development boards. Start inserting at a 30 degree angle and then level it out to match hexagonal spacers attached on board. Use a 5/32 inch hexagonal driver to add a socket cap screw. The socket cap screws should lock the module in place.
2. Connect both development boards to power supplies and plug the power supplies into an outlet. Ensure that the large switch labeled **SW3** is flipped to **INT**. Ensure that you flip the $\overline{\text{SHUTDOWN}}$ switch (SW7) to high.
3. Connect the development boards to the USB port on a computer using the mini-USB cables. Separate the development boards by at least 2 m (6 ft).
4. Use the MMCX to SMA cable provided to connect the XLR PRO to the SMA connector (P7) on the development boards .
5. Connect the A09-HASM-675 antenna to the RPSMA connector labeled P5.

The following table shows the Development Board for the XLR PRO RF Module Development Kit contents.

Description	Quantity	Part number
Development Board for the XLR PRO RF Module	2	XL9-DEV-U
Mini USB cable	2	N/A
12 VDC power supply	2	N/A
RPSMA antenna	2	A09-HASM-675

Configure the device using XCTU

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

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

Click **Discover devices** and follow the instructions. XCTU should discover two XLR PRO RF Module modules.

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

Configure the devices for a range test

For devices to communicate with each other, you configure them so they are in the same network. You also set the local device to API mode to obtain all possible data of the remote device.

When you connect the development board to a PC for the first time, the PC automatically installs drivers, which may take two minutes to complete.

1. Add the two devices to XCTU.
2. Select the first XLR PRO RF Module and click the **Load default firmware settings** button.
3. Configure the following parameters:

   ID: 2015

   NI: LOCAL_DEVICE

   AP: API enabled [1]
4. Click the **Write radio settings** button.
5. Select the other XLR PRO RF Module and click the **Default firmware settings** button.
6. Configure the following parameters:

   ID: 2015

   NI: REMOTE_DEVICE

   AP: API disabled [0]
7. Click the **Write radio settings** button.

   After you write the radio settings for each device, their names appear in the **Radio Modules** area. The Port indicates LOCAL_DEVICE is in API mode.
8. Disconnect REMOTE_DEVICE from your computer and remove it from XCTU.
9. Place REMOTE_DEVICE at the testing location and connect its power supply.

Perform a range test

A range test is a simple point to point wireless demonstration that tests the devices' ability to transmit to and receive from each other. Wireless environments vary dramatically depending on many factors and the range test allows you to experiment with the devices in your own environment.

WARNING! Keep the boards 2 m (6 ft) apart when transmitting to protect the device's front end.

Follow these steps to perform the range test:

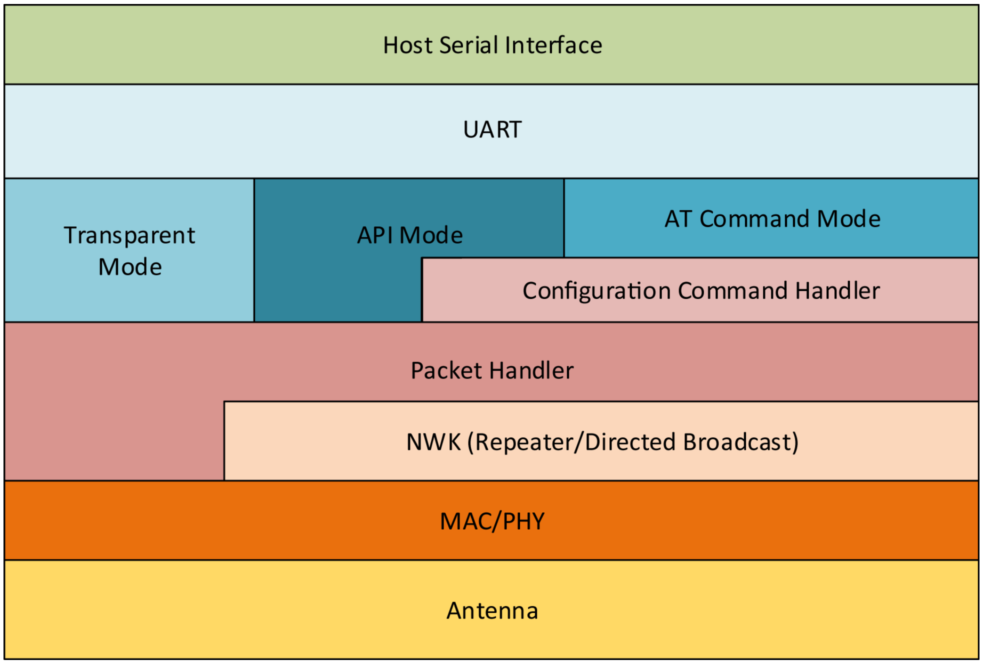
1. Open the **Tools** menu within XCTU and select the **Range Test** option.
2. Your local devices are listed on the left side of the Devices Selection section. Select the local device and click **the Discover remote devices** button .
3. When the discovery process finishes, the remote device is displayed in the **Discovering remote devices...** dialog. Click **Add selected devices**.
4. Select the remote device from the **Discovered device** list located on the right panel inside.
5. Click **Start Range Test**.

6. Range test data is represented in the chart. By default, 100 packets are sent for the test. XCTU displays the instant local and remote RSSI in two separate controls, as well as the number of packets sent and received. Remote devices only report their RSSI value when the local device is operating in API mode (by setting **AP** to 1).
7. Watch the range test status indicators: RSSI will decrease the further away you are, and if you continue to move the setup further away, eventually you will see the percentage of successful packets drop below 100%, indicating you are approaching the limits of the range.
8. Click **Stop Range Test** to stop the process at any time.

Operations

XLR PRO operational design

The XLR PRO RF Module uses a multi-layered firmware base for data flow. The flow of data depends on the hardware and software configuration you choose. The configuration block diagram below shows the host serial interface as the physical starting point and the antenna as the physical endpoint for transferred data. As long as an interface block is able to touch another block above or below, the two interfaces can interact. For example, if the XLR PRO RF Module is using API mode, Transparent Mode is not available.



UART

The following parameters must be configured to match the host device:

- **BD**: Baud rate (See the AT command table for limits)
- **NB**: Parity (None, Even, or Odd)
- **SB**: Stop bits (1 or 2)

UART connections support hardware flow control using $\overline{\text{CTS}}$ and $\overline{\text{RTS}}$ and requires matching parameters on the XLR PRO and the host device. This includes the following:

- **D6**: $\overline{\text{RTS}}$ flow control. If enabled, then XLR PRO will not output data unless $\overline{\text{RTS}}$ is asserted. The host device should not de-assert $\overline{\text{RTS}}$ for long periods of time to avoid filling the serial transmit buffer. If an RF data packet is received, and the serial transmit buffer does not have enough space for all of the data bytes, the entire RF data packet will be discarded.
- **D7**: $\overline{\text{CTS}}$ flow control. If enabled, then XLR PRO will not assert $\overline{\text{CTS}}$ low unless it can handle more data from the host.

- **FT**: Flow control threshold. If $\overline{\text{CTS}}$ flow control is enabled (with the **D7** parameter), the XLR PRO deasserts $\overline{\text{CTS}}$ when the serial receive buffer reaches the threshold defined by the **FT** parameter. Once $\overline{\text{CTS}}$ is de-asserted, it will not be asserted again until the receive buffer has 17 bytes less than the threshold defined by **FT**. By default, **FT** is 65 bytes less than the maximum space available for receive data.

Serial communications

XLR PRO handles all serial traffic the same, regardless of the interface in use.

Serial buffers

- **Serial receive buffer**

  When serial data enters the XLR PRO, the data is stored in the serial receive buffer until it can be processed. Under certain conditions, the XLR PRO may not be able to process data in the serial receive buffer immediately. If large amounts of serial data are sent to the XLR PRO such that the serial receive buffer would overflow, then the new data will be discarded. This can be avoided using hardware or software flow control.
- **Serial transmit buffer**

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

SPI signals

The XLR PRO RF Module supports SPI communications in slave mode. Slave mode receives the clock signal and data from the master and returns data to the master. The SPI port uses the following signals on the device:

Signal	Pin number	Applicable AT command
SPI_MOSI (Master out, Slave in)	33	**P5**
SPI_MISO (Master in, Slave out)	35	**P6**
SPI_CLK (Serial clock)	37	**P7**
SPI_$\overline{\text{SSEL}}$ (Slave select)	31	**P8**
SPI_$\overline{\text{ATTN}}$ (Attention)	29	**P9**

By default, the inputs have pull-up resistors enabled. Use the **PR** command to disable the pull-up resistors. When the SPI pins are not connected but the pins are configured for SPI operation, then the device requires the pull-ups for proper UART operation.

Signal description

SPI_MOSI: When SPI_SSEL is asserted (low) and SPI_CLK is active, the device outputs the data on this line at the SPI_CLK rate. When SPI_SSEL is de-asserted (high), you should tri-state this output such that another slave device can drive the line.

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

SPI_CLK: The SPI master outputs a low signal on this line to select the desired slave. When you configure the device for SPI operations, this pin is an input. This signal clocks data transfers on MOSI and MISO.

SPI_SSEL: The SPI master outputs a clock on this pin, and the rate must not exceed the maximum allowed, 6 Mb/s. When you configure the device for SPI operations, this pin is an input. This signal enables serial communication with the slave.

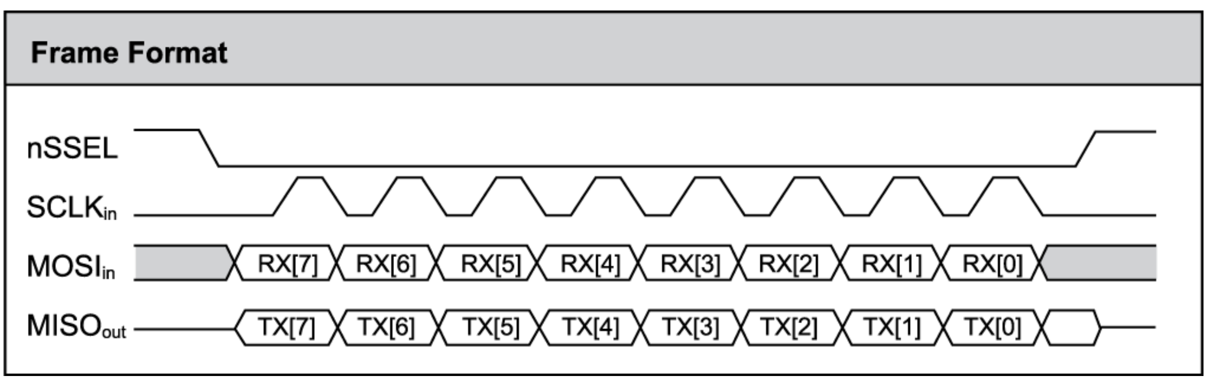
SPI_ATTN: The device asserts this pin low when it has data to send to the SPI master. When you configure this pin for SPI operations, it is an output (not tri-stated). This signal alerts the master that the slave has data queued to send. The device asserts this pin as soon as data is available to send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

Slave mode characteristics

In slave mode, the following apply:

- SPI Clock rates up to 6 MHz (6 Mb/s) are possible.
- Data is MSB first.
- It uses Frame Format Mode 0. This means CPOL= 0 (idle clock is low) and CPHA = 0 (data is sampled on the clock’s leading edge). The picture below diagrams Mode 0.
- The SPI port is setup for API mode and is equivalent to **AP** = 1.

The following picture shows the frame format for SPI communications.

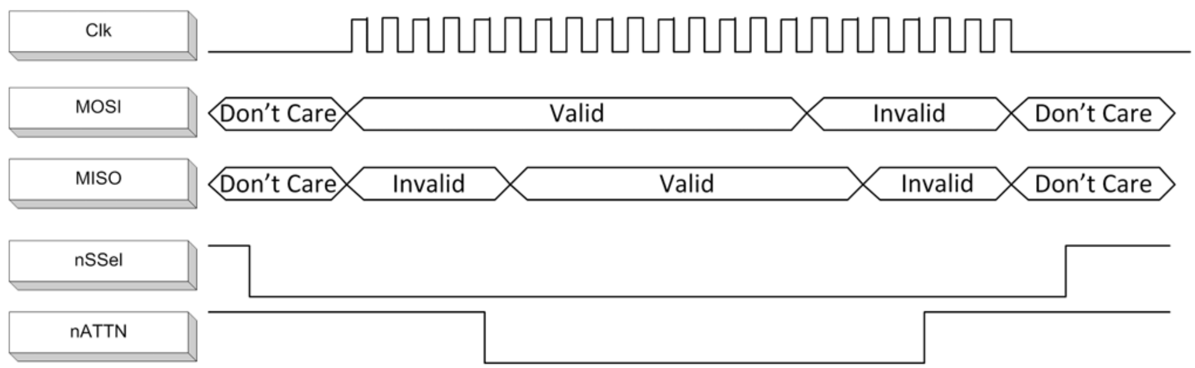


Full duplex operation

When using SPI on the XLR PRO RF Module the device uses API operation (**AP** = 1) without escaped characters to packetize data. SPI is a full duplex protocol, even when data is only available in one direction. This means that whenever a device receives data, it also transmits, and that data is normally invalid. Likewise, whenever a device transmits data, invalid data is probably received. To determine whether or not received data is invalid, the firmware places the data in API packets.

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

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



Configuration considerations

The configuration considerations are:

- How do you select the serial port? For example, should you use the UART or the SPI port?
- If you use the SPI port, what data format should you use in order to avoid processing invalid characters while transmitting?
- What SPI options do you need to configure?

SPI and API mode

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

SPI parameters

Most host processors with SPI hardware allow you to set the bit order, clock phase and polarity. For communication with all XLR PRO RF Modules, the host processor must set these options as follows:

- Bit order: send MSB first
- Clock phase (CPHA): sample data on first (leading) edge
- Clock polarity (CPOL): first (leading) edge rises

All XLR PRO RF Modules use SPI mode 0 and MSB first. Mode 0 means that data is sampled on the leading edge and that the leading edge rises. MSB first means that bit 7 is the first bit of a byte sent over the interface.

Serial port selection

To enable the UART port, configure DIN and DOUT (**P3** and **P4** parameters) as peripherals. To enable the SPI port, enable SPI_MISO, SPI_MOSI, SPI_SSEL , and SPI_CLK (**P5** through **P9**) as peripherals. If you enable both ports then output goes to the UART until the first input on SPI.

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

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

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

Serial receive buffer

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

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

Serial transmit buffer

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

Networking methods

MAC/PHY layers

PHY stands for "physical layer." The PHY layer manages the hardware that modulates and demodulates the RF bits.

MAC stands for "media access control." The MAC layer sends and receives RF frames. Each packet includes a MAC layer data header that contains addressing information, as well as packet options. This layer implements packet acknowledgments (ACKs), packet tracking to eliminate duplicates, and so on.

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

The following table shows the AT commands related to the MAC/PHY layers.

AT command	Description
ID	The **ID** (network identifier) command sets the network identifier. For XLR PRO radios to communicate, you must configure them with the same network identifier.
PL	The **PL** (power level) command sets the transmit (TX) power level. You can reduce the power level from the maximum to reduce current power consumption or to test at short distances. This comes at the expense of reduced radio range.
RR	The **RR** (unicast retries) command specifies the number of times a sending radio attempts to get an ACK from a destination radio when sending a unicast packet.
MT	The **MT** (broadcast multi-transmit) command specifies the number of times a broadcast packet is repeatedly transmitted. This adds redundancy to improve reliability.

Addressing basics

64-bit addresses

We assign each device a unique IEEE 64-bit address at the factory. When a device is in API operating mode and it sends a packet, this is the source address that the receiving device returns.

- Use the **SH** and **SL** commands to read this address.
- The form of the address is: 0x0013A2XXXXXXXXXX.
- The first six digits are the Digi Organizationally Unique Identifier (OUI).
- The broadcast address is 0x000000000000FFFF.

Make a unicast transmission

To transmit to a specific device in Transparent operating mode:

- Set **DH**:**DL** to the **SH**:**SL** of the destination device.

To transmit to a specific device in API operating mode:

- In the 64-bit destination address of the API frame, enter the **SH**:**SL** address of the destination device.

Make a broadcast transmission

To transmit to all devices in Transparent operating mode:

- Set **DH**:**DL** to 0x000000000000FFFF.

To transmit to all devices in API operating mode:

- Set the 64-bit destination address to 0x000000000000FFFF.

The scope of the broadcast changes based on the delivery method you choose.

Delivery method

The XLR PRO RF Module supports two delivery methods:

- Point-to-multipoint (0x40)
- Repeater (directed broadcast) (0x80)

Transparent mode uses the **TO** (transmit options) parameter as the default delivery method. For API transmissions, the TxOptions API field specifies the delivery method. When the TxOptions API field is set to 0, the value in the **TO** parameter will also be used by API transmissions.

Point to Point / Point to Multipoint (P2MP)

This delivery method does not use a network header, only the MAC header.

In P2MP, the sending devices always send all messages directly to the destination. Other nodes do not repeat the packet. The sending device only delivers a P2MP unicast directly to the destination device, which must be in range of the sending device.

The XLR PRO RF Module uses patented technology that allows the destination device to receive unicast transmissions directed to it, even when there is a large amount of traffic. This works best if you keep broadcast transmissions to a minimum.

A sending node repeats a P2MP broadcast transmission **MT**+1 times, but the receiving nodes do not repeat it, so like a unicast transmission, the receiving device must be in range.

All devices that receive a P2MP broadcast transmission will output the data through the UART.

Repeater/directed broadcast

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

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

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

By default, the **CE** (node messaging option) parameter is set to not route broadcasts. Due to the long-range of the XLR PRO, Digi advises you to evaluate on a per-radio basis which nodes should be configured as repeaters. Limiting the amount of congestion and generated RF traffic provides a more reliable network.

Transmission timeouts

When a node receives an API Tx Request while in API mode or an **RO** (packetization timeout) while in transparent mode, the time required to route the data to its destination depends on a number of configured parameters and whether the transmission is a unicast or a broadcast.

Note The timeouts in this section are theoretical timeouts. An application should pad the calculated maximum timeouts by a few hundred milliseconds. When using API mode, Tx Status API packets should be the primary method of determining if a transmission has completed.

Transmitting a broadcast

A directed broadcast transmission must be relayed by all routers in the network. The maximum delay is when the sender and receiver are on opposite ends of the network. The **NH** (network hops) and **%H** (MAC unicast one-hop timeout) parameters define the maximum broadcast delay as follows:

BroadcastTxTime = NH*%8

AT commands

Special commands

The following commands are special commands.

AC (Apply Changes)

Immediately applies new settings without exiting Command mode.

Parameter range

N/A

Default

N/A

FR (Software Reset)

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

Parameter range

N/A

Default

N/A

RE (Restore Defaults)

Restore device parameters to factory defaults.

In order for the default parameters to persist through subsequent resets, send a separate **WR** command after **RE**.

Parameter range

N/A

Default

N/A

WR (Write)

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

When you issue a **WR** command add a 100 millisecond delay or wait for an **OK** response before issuing any subsequent AT commands.

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

Parameter range

N/A

Default

N/A

MAC/PHY commands

The following AT commands are MAC/PHY commands.

ID (Network ID)

Sets or displays the network identifier for the module. To communicate with other modules in a network, the modules must have matching network identifiers. If you are using OEM network identifiers, set **ID** to **FFFF** to use the factory value.

Parameter range

0 - 0x7FFF

Default

0x7FFF

BR (RF Data Rate)

Sets or displays the rate at which RF data is transmitted for all operational modes. Devices within a network do not need to have matching data transmission rates. The **BR** setting does not control the rate at which RF modules receive data.

Range

An integer from **0** through **8**:

Value	Description
0	9.38 kb/s
1	28.14 kb/s
2	65.66 kb/s
3	140.7 kb/s
4	290.8 kb/s
5	590.9 kb/s
6	1.191 Mb/s
7	2.392 Mb/s
8	3.189 Mb/s

Default

Default is **4** which indicates a data transmission rate of 290.8 kb per second.

PL (Power Level)

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

Parameter range

An integer from **0** through **4**:

Value	Description
0	0 dBm, (1 mW)
1	+10 dBm, (10 mW)
2	+20 dBm, (100 mW)
3	+27 dBm, (500 mW)
4	+30 dBm, (1 Watt)

Default

4

RR (Unicast Retries)

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

Parameter range

0 - 0xF

Default

0xA (10 retries)

MT (Broadcast Multi-Transmits)

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

Parameter range

0x0 - 0x8

Default

3

Diagnostic commands

The following commands are diagnostic commands.

DB (Received Signal Strength)

Reports the received signal strength of the last received RF data packet. Because **DB** reports the signal strength of the last hop only, **DB** does not provide an accurate quality measurement for a multihop link.

DB returns a hexadecimal value for the -dBm measurement. For example, if **DB** returns 0x60, then the RSSI of the last packet received is -96 dBm.

Parameter range

0 - 0xFF [Read-only]

Default

N/A

EA (MAC ACK Timeouts)

Reports or resets the total number of MAC-level unicast transmissions that timed out waiting for a MAC ACK. The total can be up to **RR** (unicast retries) + 1 timeouts per unicast, up to a maximum of **0xFFFF**. After **0xFFFF**, additional retries are not counted. You can reset the counter to any 16-bit value within the valid range by appending a hexadecimal value to the **EA** command.

EA is a volatile value—that is, the value does not persist across module resets.

Parameter range

0 - 0xFFFF

Default

N/A

ER (Received Error Count)

Reports or resets the total number of received packets that were rejected because of bit errors in the packet, up to a maximum of **0xFFFF** errors. After **0xFFFF**, additional errors are not counted. (Occasionally, random noise can cause a packet to be rejected.) You can reset the counter to any 16-bit value within the valid range by appending a hexadecimal value to the **ER** command.

ER is a volatile value—that is, the value does not persist across device resets.

Parameter range

0 - 0xFFFF

Default

0

GD (Good Packets Received)

Reports or resets the total number of successfully received packets that contain a valid MAC header, up to a maximum of **0xFFFF** packets. After **0xFFFF**, additional successfully received packets are not counted. You can reset the counter to any 16-bit value within the valid range by appending a hexadecimal value to the **GD** command.

GD is a volatile value—that is, the value does not persist across device resets.

Parameter range

0 - 0xFFFF

Default

N/A (0 after reset)

TR (Transmission Failure Count)

Reports or resets the total number of unicast transmissions for which all retries failed with no MAC ACK from the destination node, up to a maximum of **0xFFFF** transmission failures. After **0xFFFF**, failures are no longer counted. You can reset the counter to any 16-bit value within the valid range by appending a hexadecimal value to the **TR** command.

TR is a volatile value—that is, the value does not persist across module resets.

Parameter range

0 - 0xFFFF

Default

N/A

UA (Unicasts Attempted)

Reports or resets the total number of MAC unicast transmissions for which an ACK is requested, up to a maximum of **0xFFFF** transmissions. After **0xFFFF**, additional transmissions are not counted. You can reset the counter to any 16-bit value within the valid range by appending a hexadecimal value to the **UA** command.

UA is a volatile value—that is, the value does not persist across device resets.

Parameter range

0 - 0xFFFF

Default

0

%H (MAC Unicast One Hop Time)

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

Parameter range

[Read-only]

Default

0x267

%8 (MAC Broadcast One Hop Time)

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

Parameter range

[Read-only]

Default

0x23D

N? (Network Discovery Timeout)

Reports the maximum response time in milliseconds for **ND** (Network Discover) and **DN** (Discover Node) responses. The timeout is based on **NT** (Node Discovery Timeout) and the network propagation time.

Parameter range

[Read-only]

Default

0x3C41

Network commands

The following commands are network commands.

CE (Node Messaging Options)

Sets the routing and messaging mode for the device. A device can be configured to route or not route and configured to multi-hop packets when TO (Transmit Options) is configured for Directed Broadcast (0x80).

Parameter range

An integer from 0 through 2:

Value	Description
0	Standard router node. A standard router repeats directed broadcasts.
1	Not applicable.
2	Non-routing node.

Default

2

BH (Broadcast Hops)

Sets or displays the maximum number of transmission hops for directed broadcast data transmissions when **TO** (Transmit Options) is configured for Directed Broadcast (0x80). For maximum hops, set the value to **0**. If **BH** is set to a value greater than the value for NH (Network Hops), then the **NH** value is used.

Parameter range

An integer from **0** through **4**.

Default

0

NH (Network Hops)

Sets or displays the maximum number of hops expected for a Directed Broadcast network.

Parameter range

An integer from **0** through **4**.

Default

4

NN (Network Delay Slots)

Sets or displays the maximum delay slots before rebroadcasting a Directed Broadcast packet.

Parameter range

An integer from **0** through **8**.

Default

3

RF Addressing commands

The following AT commands are RF addressing commands.

SH (Serial Number High)

Displays the upper 32 bits of the unique IEEE 64-bit RF extended address assigned to the XLR PRO.

Parameter range

0 - 0xFFFFFFFF [read-only]

Default

Set in the factory

SL (Serial Number Low)

Displays the lower 32 bits of the unique IEEE 64-bit RF extended address assigned to the XLR PRO.

Parameter range

0 - 0xFFFFFFFF [Read-only]

Default

Set in the factory

DH (Destination Address High)

Displays the upper 32 bits of the unique IEEE 64-bit RF extended address for the destination module. **DH** and DL (Destination Address Low) together define the destination address used for transmission of transparent data. For broadcast, use the destination address **0x000000000000FFFF**.

Parameter range

0 - 0xFFFFFFFF

Default

0

DL (Destination Address Low)

Displays the lower 32 bits of the unique IEEE 64-bit RF extended address for the destination module. and **DL** together define the destination address used for transmission of transparent data in either serial or IP socket modes. For broadcast, use the destination address **0x000000000000FFFF**.

Parameter range

0 - 0xFFFFFFFF

Default

0xFFFF

TO (Transmit Options)

Sets or displays transmit options for all serial transmissions. **TO** options can be overridden packet-by-packet using the **TxOptions** field of an API **TxRequest** frame.

Parameter range

One of the following hexadecimal values:

Value	Description
0x40	Point-to-point/multipoint, ACK enabled
0x41	Point-to-point/mulitpoint, ACK disabled
0x80	Repeater/Directed broadcast, ACK enabled
0x81	Repeater/Directed broadcast, ACK disabled

Default

0x40

NI (Node Identifier)

Sets or displays a string identifier for the XLR PRO. The **NI** string identifier is returned by the ND (Network Discover) command. The **NI** string identifier can also be used by the DN (Discover Node) command to set the destination address—DL (Destination Address Low) and —to the extended 64-bit address of the XLR PRO with the matching **NI** string identifier.

Parameter range

A string of case-sensitive ASCII printable characters from 0 to 20 bytes in length. The string cannot start with the space character. A carriage return or a comma automatically ends the command.

Default

One ASCII space character (0x20)

NT (Node Discovery Timeout)

Sets or displays the maximum randomized delay time used for sending network discovery responses—ND (Network Discover), DN (Discover Node), and FN (Find Neighbors) command responses. The random delay time is used to stagger the discovery command responses to alleviate network congestion.

Use N? (Network Discovery Timeout) to determine the maximum response time a ND (Network Discover) response requires based on **NT** and network propagation time.

Parameter range

0x20 - 0x2EE0 (x 100 ms)

Default

0x82 (13 seconds)

NO (Node Discovery Options)

Sets or displays network discovery options. Depending on the selected options, **NO** changes the behavior of the ND (Network Discover) command and determines the values returned for received **ND** responses and API node identification frames.

Parameter range

0x0 - 0x7 (bit field)

Hex value	Bitfield	Description
0x01	0000 0001	Appends DD (Device Type Identifier) value to ND (Network Discover) responses and API node identification frames.
0x02	0000 0010	Sends **ND** or FN (Find Neighbors) response frame when **ND** is issued.
0x03	0000 0011	Selects both 01 and 02 options
0x04	0000 0100	Appends RSSI of the last hop for the repeater networks to **ND** or **FN** responses and API node identification frames.
0x05	0000 0101	Selects both 01 and 04 options.
0x06	0000 0110	Both 02 and 04 options.
0x07	0000 0111	Select all options: 01, 02, and 04.

Default

0x0

CI (Cluster ID)

Sets or displays the default application layer cluster identifier used for all data transmissions.

Parameter range

0 - 0xFFFF

Value	Description
0x11	Transparent data
0x12	Loopback (the destination node echoes transmitted packets back to the originator)
0x14	Link test
0x23	Memory Access (GPM)

Default

0x11

DE (Destination Endpoint)

Sets or displays the application layer destination ID value. The value is used as the destination endpoint for all data transmissions. The default value (0xE8) is the Digi data endpoint.

Parameter range

Value	Description
0xE6	Digi device endpoint
0xE8	Digi data endpoint

Default

0xE8

SE (Source Endpoint)

Sets or displays the application layer source endpoint value. The value is used as the source endpoint for all data transmissions. The default value (0xE8) is the Digi data endpoint.

Parameter range

Value	Description
0xE6	Digi device endpoint
0xE8	Digi data endpoint

Default

0xE8

Addressing discovery and configuration commands

DN (Discover Node)

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

DN behavior in Command mode

When a destination address is discovered, the device:

1. Sets DL (Destination Address Low) and to the extended 64-bit address of the device with the matching string.
2. Returns **OK<CR>**.
3. Exits command mode to allow immediate communications.

For API mode (AP = 1 or 2):

When a destination address is discovered:

- Receiving device returns **0xFFFE** and 64-bit extended addresses in an API command response frame.

Errors

If there is no response after the number of milliseconds set by the N? (Network Discovery Timeout) parameter or a parameter is not specified (left blank), the command is terminated and an **ERROR** message is returned. When an **ERROR** is returned, command mode is not exited.

Parameter range

A string of case-sensitive ASCII printable characters from 1 to 20 bytes in length. The string cannot start with the space character. A carriage return or a comma automatically ends the command.

Default

N/A

ND (Network Discover)

Discovers and reports all devices found in the network.

For each discovered device, the following information is returned:

RESERVED<CR> (always **0xFFFE**)
SH<CR> (4 bytes)
SL<CR> (4 bytes)
NI<CR> (Variable length, up to 20 bytes)
PARENT_NETWORK ADDRESS<CR> (always **0xFFFE**)
DEVICE_TYPE<CR> (1 Byte: **0**=Coord, **1**=Router, **2**=End Device)
STATUS<CR> (1 Byte: Reserved)
PROFILE_ID<CR> (2 Bytes)
MANUFACTURER_ID<CR> (2 Bytes)
DIGI DEVICE TYPE<CR> (4 Bytes. Optionally included based on settings.)
RSSI OF LAST HOP<CR> (1 Byte. Optionally included based on settings.)
<CR>

After the number of milliseconds set by the N? (Network Discovery Timeout) parameter, the command ends by returning a carriage return (CR). Optionally, **ND** also accepts a as a parameter and only a device that matches the identifier is returned.

If the **ND** command is sent through a local API frame, each response is returned as a separate Local or Remote AT Command Response API packet, respectively. The data returned is the same without carriage return delimiters. The string ends with a **0x00** (null) character.

Parameter range

N/A

Default

N/A

FN (Find Neighbors)

Discovers and reports all devices found within immediate RF range.

For each discovered device, the following information is reported:

RESERVED<CR> (always **0xFFFE**)

SH<CR> (4 bytes)

SL<CR> (4 bytes)

NI<CR> (Variable length, up to 20 bytes)

PARENT_NETWORK ADDRESS<CR> (always **0xFFFE**)

DEVICE_TYPE<CR> (1 Byte: **0**=Coord, **1**=Router, **2**=End Device)

STATUS<CR> (1 Byte: Reserved)

PROFILE_ID<CR> (2 Bytes)

MANUFACTURER_ID<CR> (2 Bytes)

DIGI DEVICE TYPE<CR> (4 Bytes. Optionally included based on settings.)

RSSI OF LAST HOP<CR> (1 Byte. Optionally included based on settings.)

<CR>

If the **FN** command is issued in command mode, after the number of milliseconds set by the N? (Network Discovery Timeout) parameter + overhead time, the command ends by returning a carriage return (CR).

If the **FN** command is sent through a local API frame, each response is returned as a separate Local or Remote AT Command Response API packet, respectively. The data returned is the same without carriage return delimiters. The string ends with a **0x00** (null) character.

Parameter range

N/A

Default

N/A

Security commands

The following AT commands are security commands.

KY (AES Encryption Key)

Sets the 16-byte network security key used for encryption and decryption of transmitted data. This command is write-only. If you attempt to read **KY**, an **OK** status is returned. You must set the encryption key to the same value for all devices for successful communication.

Parameter range

128-bit value

Default

N/A

Serial interfacing commands

The following commands are serial interfacing commands.

BD (Baud Rate)

Sets or displays the serial baud rate for the XLR PRO.

Parameter range

A **BD** value of 1 through 0xA selects a standard baud rate preset.

A hexadecimal value from 0x5B9 through 0x5B8D80 specifies a specific baud rate.

Preset values include:

Value	Description
0x1	2,400 b/s
0x2	4,800 b/s
0x3	9,600 b/s
0x4	19,200 b/s
0x5	38,400 b/s
0x6	57,600 b/s
0x7	115,200 b/s
0x8	230,400 b/s
0x9	460,800 b/s
0xA	921,600 b/s

To set a non-standard baud rate, enter a value above **0x5B9**. **BD** adjusts the value to the closest supported baud rate. After entering a specific baud rate, query **BD** to read the actual baud rate. Baud rates can be set as high as 6 Mb/s, but the host and serial switching circuitry may not support it.

Default

3 (9600 b/s)

NB (Parity)

Set or display the parity settingsfor serial communications.

Parameter range

0x00 - 0x02

Parameter	Description
0x00	No parity
0x01	Even parity
0x02	Odd parity

Default

0x00

SB (Stop Bits)

Sets or displays the number of stop bits for the UART.

Range

One of the following values:

Value	Description
0	One (1) stop bit.
1	Two (2) stop bits.

Default

0

RO (Packetization Timeout)

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

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

Parameter range

0 - 0xFF (x character times)

Default

3

FT (Flow Control Threshold)

Set or display the flow control threshold.

De-assert $\overline{\text{CTS}}$ and/or send XOFF when **FT** bytes are in the UART receive buffer. Re-assert $\overline{\text{CTS}}$ when less than **FT**-16 bytes are in the UART receive buffer.

Parameter range

0x11 - 0x94F

Default

0x91F

AP (API Mode)

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

The device ignores this command when using SPI, where API mode is always enabled.

Parameter range

0 - 2

Value	Description
0	Transparent mode. API mode is off. All serial input and output is raw data and packets are delineated using the **RO** and **RB** parameters.
1	API mode without escapes. The device packetizes all UART input and output data in API format, without escape sequences.
2	API mode with escapes. The device is in API mode and inserts escaped sequences to allow for control characters.

Default

0

AO (API Options)

Sets or displays the API data frame output format for received frames. Applies to both UART and SPI interfaces.

Parameter range

0, 1

Value	Description
0	API RX indicator (0x90)
1	API Explicit RX indicator (0x91)

Default

0

I/O settings commands

The following AT commands are I/O settings commands.

D0 (DIO0/AD0)

The DIO0/AD0 pin configuration (pin 18).

Parameter range

0 - 5

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

Default

1

D1 (DIO1/AD1)

The DIO1/AD1 pin configuration (pin 20).

Parameter range

0, 2 - 5

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

Default

0

D2 (DIO2/AD2)

The DIO2/AD2 pin configuration (pin 22).

Parameter range

0, 2 - 5

Parameter	Description
0	Disabled
1	N/A

Parameter	Description
2	ADC
3	Digital input
4	Digital output, low
5	Digital output, high

Default

0

D3 (DIO3/AD3)

The DIO3/AD3 pin configuration (pin 24).

Parameter range

0, 2 - 5

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

Default

0

D4 (DIO4)

The DIO4 pin configuration (pin 26).

Parameter range

0, 3 - 5

Parameter	Description
0	Disabled
1	N/A
2	N/A
3	Digital input

Parameter	Description
4	Digital output, low
5	Digital output, high

Default

0

D5 (DIO5/ASSOCIATED_INDICATOR)

The ASSOC/DIO5 pin configuration (pin 28).

Parameter range

Parameter	Description
0	Disabled
1	Associate LED indicator
2	N/A
3	Digital input
4	Digital output, default low
5	Digital output, default high

Default

1

D6 (DIO6/$\overline{\text{RTS}}$)

The DIO6/$\overline{\text{RTS}}$ pin configuration (pin 41).

Parameter range

0, 1, 3 - 5

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

Default

0

D7 (DIO7/$\overline{\text{CTS}}$)

The DIO7/$\overline{\text{CTS}}$ pin configuration (pin 39).

Parameter range

0, 1, 3 - 7

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

Default

1

D8 (DIO8/$\overline{\text{DTR}}$/SLP_RQ)

The DIO8/$\overline{\text{DTR}}$/SLP_RQ pin configuration (pin 30). The XLR PRO RF Module does not support sleep. The SLEEP_REQUEST option is provided for compatibility purposes and does not affect the module.

Parameter range

0, 1, 3 - 5

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

Default

1

D9 (DIO9/ON_SLEEP)

The DIO9/ON_SLEEP) pin configuration (pin 32).

Parameter range

0, 1, 3 - 5

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

Default

1

P0 (DIO10/RSSI/PWM0 Configuration)

The DIO10/RSSI/PWM0 pin configuration (pin 36).

Parameter range

0 - 5

Parameter	Description
0	Disabled
1	RSSI PWM0 output
2	PWM0 output
3	Digital input
4	Digital output, low
5	Digital output, high

Default

1

P1 (DIO11/PWM1 Configuration)

The DIO11/PWM1 pin configuration (pin 38).

Parameter range

0 - 5

Parameter	Description
0	Disabled
1	32.768 kHz clock output
2	PWM1 output
3	Digital input
4	Digital output, low
5	Digital output, high

Default

0

P2 (DIO12 Configuration)

The DIO12 pin configuration (pin 40).

Parameter range

0, 3 - 5

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

Default

0

P3 (DIO13/DOUT)

The DIO13/DOUT pin configuration (pin 43).

Parameter range

0, 1

Parameter	Description
0	Unmonitored digital input
1	Data out for UART

Default

1

P4 (DIO14/DIN/$\overline{\text{CONFIG}}$)

The DIO14/DIN/$\overline{\text{CONFIG}}$ pin configuration (pin 45).

Parameter range

0 - 1

Parameter	Description
0	Disabled
1	UART DIN/$\overline{\text{CONFIG}}$ enabled

Default

1

P5 (DIO15/SPI_MISO)

The DIO15/SPI_MISO pin configuration (pin 35).

Parameter range

0, 1, 4, 5

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

Default

1

P6 (DIO16/SPI_MOSI)

The DIO16/SPI_MOSI pin configuration (pin 33).

Parameter range

0, 1, 4, 5

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

Default

1

P7 (DIO17/SPI_$\overline{\text{SSEL}}$)

The DIO17/SPI_$\overline{\text{SSEL}}$ pin configuration (pin 31).

Parameter range

0, 1, 4, 5

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

Default

1

P8 (DIO18/SPI_SCLK)

The DIO18/SPI_CLK pin configuration (pin 37).

Parameter range

0, 1, 4, 5

Parameter	Description
0	Disabled
1	SPI_CLK

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

Default

1

P9 (DIO19/SPI_$\overline{\text{ATTN}}$)

The DIO19/SPI_$\overline{\text{ATTN}}$ pin configuration (pin 29).

Parameter range

0, 1, 4- 6

Parameter	Description
0	Disabled
1	SPI_$\overline{\text{ATTN}}$
2	N/A
3	N/A
4	Digital output, low
5	Digital output, high
6	UART data present indicator

Default

1

PD (Pull Up/Down Direction)

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

Parameter range

0 - 0xFFFFF

Default

0xFFFFF

PR (Pull-up/Down Resistor Enable)

The bit field that configures internal pull-up/down resistors status for I/O lines. If you set a **PR** bit to 1, it enables the internal pull-up/down resistor, 0 specifies no internal pull-up/down. The following table defines the bit-field map for both the **PR** and **PD** commands.

Bit	I/O line	Module pin
0	DIO4/AD4	26
1	DIO3/AD3	24
2	DIO2/AD2	22
3	DIO1/AD1	20
4	DIO0/AD0	18
5	DIO6/$\overline{\text{RTS}}$	41
6	DIO8/$\overline{\text{DTR}}$/SLEEP_REQUEST	30
7	DIO14/DIN/$\overline{\text{CONFIG}}$	45
8	DIO5/ASSOCIATE	28
9	DIO9/On/$\overline{\text{SLEEP}}$	32
10	DIO12	40
11	DIO10/RSSI/PWM0	36
12	DIO11/PWM1	38
13	DIO7/$\overline{\text{CTS}}$	39
14	DIO13/DOUT	43
15	DIO15/SPI_MISO	35
16	DIO16/SPI_MOSI	33
17	DIO17/SPI_$\overline{\text{SSEL}}$	31
18	DIO18/SPI_SCLK	37
19	DIO19/SPI_$\overline{\text{ATTN}}$	29

Parameter range

0 - 0xFFFFF (bit field)

Default

0xFFFFF

Example

If **PR** is set to 0x41F, and **PD** is set to 0xFF8, then DIO4, DIO3, and DIO2 have a pull-down resistor enabled, while DIO1, DIO0, and DIO12 have a pull-up resistor enabled. These pins are only affected if they are configured as digital inputs.

M0 (PWM0 Duty Cycle)

The duty cycle of the PWM0 line. Use the **P0** command to configure the line as PWM output.

Parameter range

0 - 0x3FF

Default

0

M1 (PWM1 Duty Cycle)

The duty cycle of the PWM1 line. Use the **P1** command to configure the line as PWM output.

Parameter range

0 - 0x3FF

Default

0

LT (Associate LED Blink Time)

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

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

Parameter range

0, 0x14 - 0xFF (x 10 ms)

Default

0

RP (RSSI PWM Timer)

Sets or displays the amount of time in 100 ms that the pulse width modulation (PWM) output on the RSSI pin is active after a valid RF packet is received. When **RP** is set to **0xFF**, output is always on.

Parameter range

0 - 0xFF (x 100 ms)

Default

0x28 (four seconds)

I/O sampling commands

The following AT commands configure I/O sampling parameters.

AV (Analog Voltage Reference)

The analog voltage reference used for A/D sampling.

Parameter range

0 - 2

Parameter	Description
0	1.25 V reference
1	2.5 V reference

Default

1

IC (Analog Voltage Reference)

Set or read the digital I/O pins to monitor for changes in the I/O state.

IC works with the individual pin configuration commands (**D0** - **D9**, **P0** - **P2**). If you enable a pin as a digital I/O, use the **IC** command to force an immediate I/O sample transmission when the DIO state changes. If sleep is enabled, the edge transition must occur during a wake period to trigger a change detect.

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

Set unused bits to 0.

Bit	I/O line	Module pin
0	DIO0	18
1	DIO1	20
2	DIO2	22
3	DIO3	24
4	DIO4	26
5	DIO5	28
6	DIO6	41
7	DIO7	39
8	DIO8	30
9	DIO9	32
10	DIO10	36
11	DIO11	38
12	DIO12	40

Parameter range

0 - 0xFFFF (bit field)

Default

0

IR (Sample Rate)

Set or read the I/O sample rate to enable periodic sampling. When set, this parameter causes the device to sample all enabled DIO and ADC at a specified interval. Samples will be sent to the address specified by the **DH** and **DL** commands. The target device must be operating in API mode in order to output the received sample data.

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

Parameter range

0 - 0xFFFF (x 1 ms)

Default

0

TP (Temperature)

Displays the temperature of the XLR PRO in degrees Celsius. The temperature value is displayed in 8-bit two’s compliment format. For example, **0x1A** = 26°C, and **0xF6** = -10°C.

Because the XLR PRO RF module produces heat, this temperature reading is usually above the ambient temperature.

Range

0 - 0xFF which indicates degrees Celsius displayed in 8-bit two's compliment format.

Default

N/A

%V (Voltage Supply Monitoring)

The device's supply voltage in mV.

Parameter range

0 – 0xFFFF [read-only]

Default

N/A

I/O line passing commands

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

IU (I/O Output Enable)

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

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

Parameter range

0 – 0xFFFF [read-only]

Default

N/A

IA (I/O Input Address)

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

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

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

Parameter range

0 - 0xFFFFFFFFFFFFFFFF

Default

0xFFFFFFFFFFFFFFFF (I/O line passing disabled)

T0 (D0 Timeout)

Specifies how long pin D0 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T1 (D1 Timeout)

Specifies how long pin D1 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T2 (D2 Timeout)

Specifies how long pin D2 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T3 (D3 Timeout)

Specifies how long pin D3 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T4 (D4 Timeout)

Specifies how long pin D4 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T5 (D5 Timeout)

Specifies how long pin D5 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T6 (D6 Timeout)

Specifies how long pin D6 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T7 (D7 Timeout)

Specifies how long pin D7 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T8 (D8 Timeout)

Specifies how long pin D8 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

T9 (D9 Timeout)

Specifies how long pin D9 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

Q0 (P0 Timeout)

Specifies how long pin P0 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

Q1 (P1 Timeout)

Specifies how long pin P1 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

Q2 (P2 Timeout)

Specifies how long pin P2 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

Q3 (P3 Timeout)

Specifies how long pin P3 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

Q4 (P4 Timeout)

Specifies how long pin P4 holds a given value before it reverts to configured value. If set to 0, there is no timeout.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0

PT (PWM Output Timeout)

Specifies how long both PWM outputs (**P0**, **P1**) output a given PWM signal before it reverts to zero. If set to 0, there is no timeout. This timeout only affects these pins when they are configured as PWM output.

Parameter range

0 - 0x1770 (x 100 ms)

Default

0xFF

Command mode options

The following commands are Command mode option commands.

CC (Command Sequence Character)

Sets or displays the ASCII sequence character to use for entering AT command mode. Repeating the **CC** character three times causes the XLR PRO to enter AT command mode. The device responds with **OK\r** when Command mode is successfully entered. The following commands are related to **CC**:

- Use GT (Guard Times) to define a guard time—the amount of time before and after entering a command sequence—to guard against inadvertently entering Command mode.
- Use CT (Command Mode Timeout) to define the timeout for Command mode.
- Use CN (Exit Command mode) to immediately exit Command mode and return to idle mode.

Parameter range

0 - 0xFF

Default

0x2B (+ ASCII)

CN (Exit Command mode)

Exits Command mode and returns the XLR PRO to Idle mode.

Parameter range

N/A

Default

N/A

CT (Command Mode Timeout)

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

Parameter range

2 - 0x1770 (x 100 ms)

Default

0x64 (10 seconds)

GT (Guard Times)

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

Parameter range

0 - 0xFFFF

Default

0x3E8 (one second)

Firmware version/information commands

The following AT commands are firmware version/information commands.

VR (XLR PRO Firmware Version)

Displays the XLR PRO module firmware.

Range

N/A

Default

N/A

HV (Hardware Version)

Display the device's hardware version number.

Parameter range

N/A

Default

N/A

DD (Device Type Identifier)

The Digi device type identifier value. Use this value to differentiate between multiple devices.

The XLR PRO RF Module product code upper word is **0x000E**.

Parameter range

0 - 0xFFFFFFFF [Read-only]

Default

0xE0000

NP (Maximum Packet Payload Bytes)

Reads the maximum number of payload bytes that you can send in a unicast RF transmission based on the device's current configuration.

Parameter range

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

Default

N/A

CK (Configuration CRC)

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

This command allows you to detect an unexpected configuration change on a device.

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

Parameter range

0 - 0xFFFF

Default

N/A

Operate in API mode

API mode overview

By default, the XLR PRO RF Module acts as a serial line replacement (Transparent operation), it queues all UART data that it receive through the DI pin for RF transmission. When the device receives an RF packet, it sends the data out the DO pin with no additional information.

The following behaviors are inherent to Transparent operation:

- If device parameter registers are to be set or queried, a special operation is required for transitioning the device into Command Mode.

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

Use the AP command to set the operation mode

Use the **AP** (API enable) command to specify the operation mode:

AP=0: Transparent operating mode, UART serial line replacement with API modes disabled. This is the default option.

AP=1: Enables API operation.

AP=2: Enables API operation with escaped characters (not available if using SPI).

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

API frame format

An API frame consists of the following:

- Start delimeter
- Length
- Frame data
- Checksum

API operation (AP parameter = 1)

This is the recommended API mode for most applications. The following table shows the data frame structure when you enable this mode:

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

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

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

Setting API to 2 allows escaped control characters in the API frame. Due to its increased complexity, this API mode is only recommended in specific circumstances. API2 may help improve reliability if the serial interface to the module is unstable or malformed frames are frequently being generated.

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

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

API escaped operating mode works similarly to API mode. The only difference is that when working in API escaped mode, the software must escape any payload bytes that match API frame specific data, such as the start-of-frame byte (0x7E). The following table shows the structure of an API frame with escaped characters:

<table>
<tr><th>Frame fields</th><th>Byte</th><th colspan="2">Description</th></tr>
<tr><td>Start delimiter</td><td>1</td><td colspan="2">0x7E</td></tr>
<tr><td>Length</td><td>2 - 3</td><td>Most Significant Byte, Least Significant Byte</td><td rowspan="3">Characters escaped if needed</td></tr>
<tr><td>Frame data</td><td>4 - n</td><td>API-specific structure</td></tr>
<tr><td>Checksum</td><td>n + 1</td><td>1 byte</td></tr>
</table>

Escape characters

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

The following data bytes need to be escaped:

0x7E (start delimiter)

0x7D (escape)

0x11 (XON)

0x13 (XOFF)

To escape a character:

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

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

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

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

0x11 must be escaped which results in the following frame:

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

Note 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 field

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

Frame data

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

Checksum field

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.

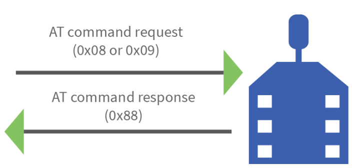
To verify: Add all bytes (include checksum, but exclude the delimiter and length). If the checksum is correct, the sum will equal 0xFF.

API serial exchanges

You can use the Frame ID field to assign an identifier to each outgoing API frame. This Frame ID, if non-zero, can correlate between the outgoing frames and the associated responses.

AT commands

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

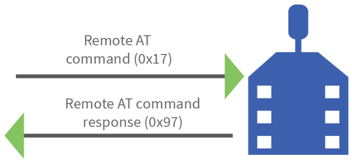


Transmit and receive RF data

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

Remote AT commands

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



Code to support future API frames

If your software application supports the API, you should make provisions that allow for new API frames in future firmware releases. For example, you can include the following section of code on a host microprocessor that handles serial API frames that are sent out the device's DOUT pin:

```
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 usec
                       break;
       }
}
```

API frame types

The following sections document API frame types.

AT Command frame - 0x08

Description

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

A 0x88 response frame is populated with the parameter value that is currently set on the device.

Format

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

Frame data fields	Offset	Description
Frame type	3	0x08
Frame ID	4	
AT command	5-6	Command name: two ASCII characters that identify the AT command.
Parameter value	7-n	If present, indicates the requested parameter value to set the given register. If no characters are present, it queries the register.

Example

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

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	
Frame type	3	0x08
Frame ID	4	0x52
AT command	5	0x4E (N)
	6	0x48 (H)
Parameter value		0x02
Checksum	8	0x0D

AT Command - Queue Parameter Value frame - 0x09

Description

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

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

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

Format

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

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

Transmit Request frame - 0x10

Description

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

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

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

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

Note Using source routing reduces the RF payload by two bytes per intermediate hop in the source route.

Format

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

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

Example

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

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x16
Frame type	3	0x10
Frame ID	4	0x01
64-bit destination address	MSB 5	0x00
	6	0x13
	7	0xA2
	8	0x00
	9	0x40
	10	0x0A
	11	0x01
	LSB 12	0x27
Broadcast radius	15	0x00
Options	16	0x00
RF data	17	0x54
	18	0x78
	19	0x44
	20	0x61
	21	0x74
	22	0x61
	23	0x30
	24	0x41
Checksum	25	0x13

If you enable escaping (**AP** = 2), the frame should look like:

0x7E 0x00 0x16 0x10 0x01 0x00 0x7D 0x33 0xA2 0x00 0x40 0x0A 0x01 0x27 0xFF 0xFE 0x00 0x00 0x54 0x78 0x44 0x61 0x74 0x61 0x30 0x41 0x7D 0x33

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

Explicit Addressing Command frame - 0x11

Description

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

This frame causes the device to send payload data as an RF packet to a specific destination, using specific source and destination endpoints, cluster ID, and profile ID.

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

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

You can set the broadcast radius from 0 up to **NH** to 0xFF. If set to 0, the value of **NH** specifies the broadcast radius (recommended). This parameter is only used for directed broadcast transmissions (transmit options = 0x80).

Format

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

Frame data fields	Offset	Description
Frame type	3	0x11
Frame ID	4	Identifies the data frame for the host to correlate with a subsequent ACK (0x8B). If set to **0**, the device does not send a response.
64-bit destination address	5-12	MSB first, LSB last. Set to the 64-bit address of the destination device. Broadcast = 0x000000000000FFFF
Reserved	13-14	Set to 0xFFFE.
Source endpoint	15	Source endpoint for the transmission.
Destination endpoint	16	Destination endpoint for the transmission.
Cluster ID	17-18	The Cluster ID that the host uses in the transmission.
Profile ID	19-20	The Profile ID that the host uses in the transmission.
Broadcast radius	21	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius set to the network maximum hops value.
Transmission options	22	
Data payload	23-n	Up to **NP** bytes per packet. Sent to the destination device.

Set all other bits to 0.

Transmit Options bit field

Bit field:

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

Set all other bits to 0.

Example

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

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

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x1A
Frame type	3	0x11
Frame ID	4	0x01

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

Remote AT Command Request frame - 0x17

Description

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

Format

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

Frame data fields	Offset	Description
Frame type	3	0x17
Frame ID	4	Identifies the data frame for the host to correlate with a subsequent ACK (0x97). If set to **0**, the device does not send a response.
64-bit destination address	5-12	MSB first, LSB last. Set to the 64-bit address of the destination device.
Reserved	13-14	
Remote command options	15	0x02 = Apply changes on remote. If you do not set this, you must send the **AC** command for changes to take effect. Set all other bits to 0.
AT command	16-17	Command name: two ASCII characters that identify the command.
Command parameter	18-n	If present, indicates the parameter value you request for a given register. If no characters are present, it queries the register.

Example

The following example sends a remote command to:

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

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x10
Frame type	3	0x17
Frame ID	4	0x01

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

AT Command Response frame - 0x88

Description

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

Format

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

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

Example

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

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x05
Frame type	3	0x88
Frame ID	4	0x01
AT command	5	0x42 (B)
	6	0x44 (D)
Command status	7	0x00

Frame data fields	Offset	Example
Command data		
Checksum	8	0xF0

Modem Status frame - 0x8A

Description

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

Format

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

Frame data fields	Offset	Description
Frame type	3	0x8A
Status	4	0x00 Hardware reset 0x01 Watchdog timer reset

Example

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

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

Transmit Status frame - 0x8B

Description

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

Format

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

Frame data fields	Offset	Description
Frame type	3	0x8B
Frame ID	4	Identifies the serial interface data frame being reported. If Frame ID = 0 in the associated request frame, no response frame is delivered.
Reserved	5-6	Set to 0xFFFE.
Transmit retry count	7	The number of application transmission retries that occur.
Delivery status	8	
Discovery status	9	0x00 = No discovery overhead 0x02 = Route discovery

Example

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

Frame Fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x07
Frame type	3	0x8B
Frame ID	4	0x47
Reserved	5	0xFF
	6	0xFE
Transmit retry count	7	0x00

Frame Fields	Offset	Example
Delivery status	8	0x00
Discovery status	9	0x02
Checksum	10	0x2E

Receive Packet frame - 0x90

Description

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

Format

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

Frame data fields	Offset	Description
Frame type	3	0x90
64-bit source address	4-11	The sender's 64-bit address. MSB first, LSB last.
Reserved	12-13	Reserved.
Receive options	14	Bit field: bit 0 = Packet acknowledged bit 1 = Packet was a broadcast packet bits 6 and 7: b’01 = Point-Multipoint b’10 = Repeater mode (directed broadcast) Ignore all other bits.
Received data	15-n	The RF data the device receives.

Example

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

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

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

Explicit Rx Indicator frame - 0x91

Description

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

Format

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

Frame data fields	Offset	Description
Frame type	3	0x91
64-bit source address	4-11	MSB first, LSB last. The sender's 64-bit address.
Reserved	12-13	Reserved.
Source endpoint	14	Endpoint of the source that initiates transmission.
Destination endpoint	15	Endpoint of the destination where the message is addressed.
Cluster ID	16-17	The Cluster ID where the frame is addressed.
Profile ID	18-19	The Profile ID where the fame is addressed.
Receive options	14	Bit field: bit 0 = Packet acknowledged bit 1 = Packet was a broadcast packet bits 6 and 7: b'01 = Point-Multipoint b'10 = Repeater mode (directed broadcast) Ignore all other bits.
Received data	21-n	Received RF data.

Example

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

If a device sends the transmission:

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

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

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x18
Frame type	3	0x91
64-bit source address	MSB 4	0x00
	5	0x13
	6	0xA2
	7	0x00
	8	0x40
	9	0x52
	10	0x2B
	LSB 11	0xAA
Reserved	12	0xFF
	13	0xFE
Source endpoint	14	0xE0
Destination endpoint	15	0xE0
Cluster ID	16	0x22
	17	0x11
Profile ID	18	0xC1
	19	0x05
Receive options	20	0x02
Received data	21	0x52
	22	0x78
	23	0x44
	24	0x61
	25	0x74
	26	0x61

Remote Command Response frame - 0x97

Description

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

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

Format

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

Frame data fields	Offset	Description
Frame type	3	0x97
Frame ID	4	This is the same value passed in to the request. If Frame ID = 0 in the associated request frame the device does not deliver a response frame.
64-bit source (remote) address	5-12	The address of the remote device returning this response.
Reserved	13-14	Reserved.
AT commands	15-16	The name of the command.
Command status	17	0 = OK 1 = ERROR 2 = Invalid Command 3 = Invalid Parameter
Command data	18-n	The value of the requested register.

Example

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

Frame data fields	Offset	Example
Start delimiter	0	0x7E
Length	MSB 1	0x00
	LSB 2	0x13
Frame type	3	0x97
Frame ID	4	0x55

Frame data fields	Offset	Example
64-bit source (remote) address	MSB 5	0x00
	6	0x13
	7	0xA2
	8	0x00
	9	0x40
	10	0x52
	11	0x2B
	LSB 12	0xAA
Reserved	13	0xFF
	14	0xFE
AT commands	15	0x53
	16	0x4C
Command status	17	0x00
Command data	18	0x40
	19	0x52
	20	0x2B
	21	0xAA
Checksum	22	0xF4

Advanced application features

Monitor I/O lines

Pin configurations

Devices support both analog input and digital I/O line modes on several configurable pins.

Pin command parameter	Description
0	Unmonitored digital input (disabled)
1	Reserved for pin-specific alternate functionality
2	Analog input (A/D pins) or PWM output (PWM pins)
3	Digital input, monitored
4	Digital output, low
5	Digital output, high
6-9	Alternate functionality, where applicable

The following table provides the pin configurations when you set the configuration command for a particular pin.

Device pin name	Device pin number	Configuration command
DIO12	40	**P2**
PWM0 / RSSI / DIO10	36	**P0**
PWM1 / DIO11	38	**P1**
$\overline{\text{DTR}}$ / SLEEP_RQ / DIO8	30	**D8**
DIO4	26	**D4**
$\overline{\text{CTS}}$ / DIO7	39	**D7**
ON/$\overline{\text{SLEEP}}$ / DIO9	32	**D9**
ASSOC / DIO5	28	**D5**
$\overline{\text{RTS}}$ / DIO6	41	**D6**
AD3 / DIO3	24	**D3**
AD2 / DIO2	22	**D2**
AD1 / DIO1	20	**D1**
AD0 / DIO0 / Commissioning Pushbutton	18	**D0**

Use the **PR** command to enable internal pull up/down resistors for each digital input. Use the **PD** command to determine the direction of the internal pull up/down resistor.

Queried sampling

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

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

Field	Name	Description
1	Sample sets	Number of sample sets in the packet. Always set to 1.
2	Digital channel mask	Indicates which digital I/O lines have sampling enabled. Each bit corresponds to one digital I/O line on the device. bit 0 = AD0/DIO0 bit 1 = AD1/DIO1 bit 2 = AD2/DIO2 bit 3 = AD3/DIO3 bit 4 = DIO4 bit 5 = ASSOC/DIO5 bit 6 = RTS/DIO6 bit 7 = CTS/GPIO7 bit 8 = DTR / SLEEP_RQ / DIO8 bit 9 = ON_SLEEP / DIO9 bit 10 = RSSI/DIO10 bit 11 = PWM/DIO11 bit 12 = CD/DIO12 For example, a digital channel mask of 0x002F means DIO0,1,2,3, and 5 are enabled as digital I/O.
1	Analog channel mask	Indicates which lines have analog inputs enabled for sampling. Each bit in the analog channel mask corresponds to one analog input channel. bit 0 = AD0/DIO0 bit 1 = AD1/DIO1 bit 2 = AD2/DIO2 bit 3 = AD3/DIO3 bit 4 = AD4/DIO4 bit 5 = ASSOC/AD5/DIO5
Variable	Sampled data set	If you enable any digital I/O lines, the first two bytes of the data set indicate the state of all enabled digital I/O. Only digital channels that you enable in the Digital channel mask bytes have any meaning in the sample set. If do not enable any digital I/O on the device, it omits these two bytes. Following the digital I/O data (if there is any), each enabled analog channel returns two bytes. The data starts with AIN0 and continues sequentially for each enabled analog input channel up to AIN5.

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

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

Periodic I/O sampling

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

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

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

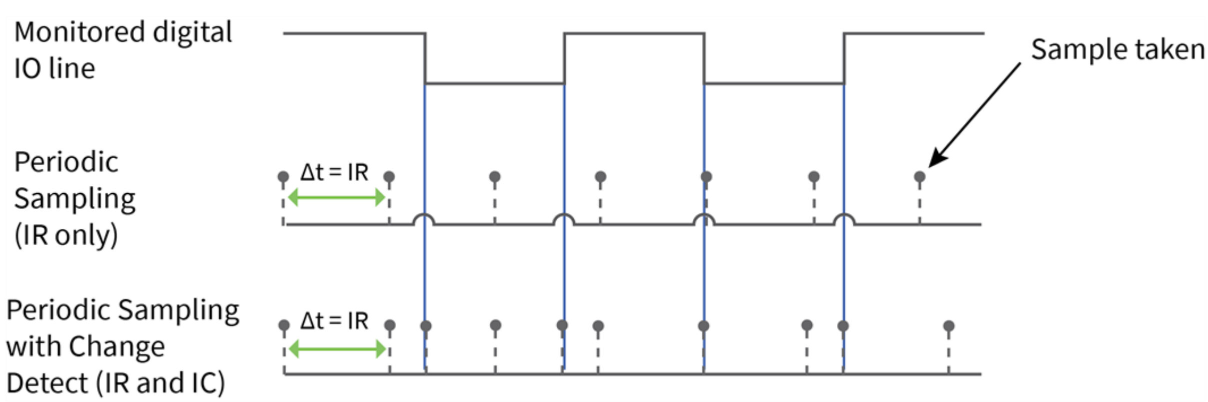
Only devices with API operating mode enabled send I/O data samples out their serial interface. Devices that are in Transparent mode (**AP** = 0) discard the I/O data samples they receive. You must configure at least one pin as a digital or ADC input to generate sample data.

A device with sleep enabled transmits periodic I/O samples at the **IR** rate until the **ST** time expires and the device can resume sleeping.

Detect digital I/O changes

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

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



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

I/O line passing

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

Digital I/O lines are mapped in pairs; pins configured as digital input on the transmitting device affect the corresponding digital output pin on the receiving device. For example: DI5 (pin 28) can only update DO5 (pin 28).

For Analog Line Passing, the XLR PRO RF Module has two PWM output pins that simulate the voltage measured by the ADC lines AD0 and AD1. For example, when configured as an ADC, AD0 (pin 18) updates PWM0 (pin 36); AD1 (pin 20) updates PWM1 (pin 38).

The default setup is for outputs to not be updated. Instead, a device sends I/O sample data out the serial interface if the device is configured for API mode (**AP** = 1 or 2). You can use the **IU** command to disable sample data output.

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

For line passing to function, the device configured with inputs must generate sample data. Refer to Pin configurations for information on how to configure digital and analog sampling.

When outputs are changed from their non-active state, the device can be setup to return the output level to its non-active state. The timers are set using the **T*n*** (**D*n*** Output Timer) and **PT** (PWM Output Timeout) commands. The timers are reset every time the device receives a valid I/O sample packet with a matching **IA** address. You can adjust the **IC** (Change Detect) and **IR** (Sample Rate) parameters on the transmitting device to keep the outputs set to their active output if the system needs more time than the timers can handle.

Configuration example

As an example for a simple digital and analog link, you could set a pair of RF devices as follows:

Command	Description	Device A	Device B
SH	Serial Number High	0x0013A200	0x0013A200
SL	Serial Number Low	0x12345678	0xABCDABCD
DH	Destination High	0x0013A200	0x00000000
DL	Destination Low	0xABCDABCD	0x0000FFFF (broadcast)
IA	I/O Input Address	0x0013A200ABCDABCD	0x0013A20012345678

Command	Description	Device A	Device B
IR	Sample Rate	0x7D0 (2 seconds)	0 (disabled)
IC	DIO Change Detect	0 (disabled)	0x1000 (DIO3 only)
D1	DIO1/AD1	2 : ADC input	N/A
P1	DIO11/PWM1	N/A	2: PWM1 output
PT	PWM Output Timeout	N/A	0x1E (3 seconds)
D2	DIO2/AD2	3: Digital input	5: Digital output, HIGH
D3	DIO3/AD3	5: Digital output, HIGH	3: Digital input
T3	DIO3 Timeout	0x64 (10 seconds)	N/A

In the example, both devices have I/O Line Passing enabled with appropriate inputs and outputs configured. The **IA** parameter determines which device on the network is allowed to affect the device's outputs.

Device A takes a periodic sample of all I/O lines every two seconds and transmits it as a unicast transmission to the address defined by **DH** and **DL** (in this case, Device B). Device B does not periodically sample, instead it monitors DIO3 for a binary change. When it detects a change on that pin, it generates a sample and transmits it as a broadcast to all devices on the network.

When Device B receives a sample packet from Device A:

- DIO2 on Device B outputs the state of DIO2 from Device A.
- PWM1 outputs a duty cycle equivalent to the analog voltage read on AD1 of Device A.
- A PWM timeout has been set to three seconds; if no sample is received, PWM1 returns to 0 V after this period.

When Device A receives a sample packet from Device B:

- DIO3 on Device A outputs the state of DIO3 from Device B.
- A DIO3 timeout has been set to 10 seconds; if no sample is received, DIO3 reverts to a HIGH state after this period.

Note By default, all Digital I/O lines have internal pull-up resistors enabled with the **PR** command. This causes inputs to float high. You can use the **PD** command to change the direction of the internal pull-up/down resistors. The XLR PRO RF Module uses an internal reference voltage of 2.5 V for ADC lines, but you can use the AV command to set it to 1.25 VDC.

Work with networked devices

Network commissioning and diagnostics

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

Local configuration

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

Remote configuration

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

Send a remote command

To send a remote command, populate the Remote Command Request (0x17) API frame with:

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

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

XCTU has a Frames Generator tool that can assist you with building and sending a remote AT frame; see: http://www.digi.com/resources/documentation/digidocs/90001458-13/default.htm#reference/r_frames_generator_tool.htm

Apply changes on remote devices

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

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

Remote command response

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

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

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

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

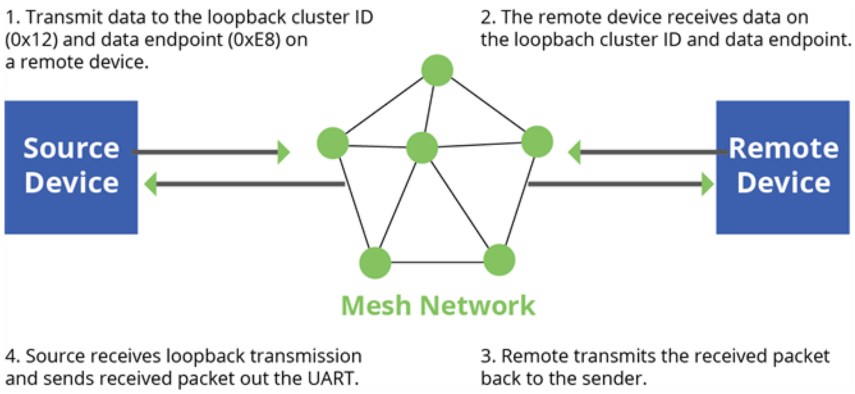
1. It could not reach the destination device.
2. You set the frame ID to 0 in the remote command request.

Test links in a network

For a network installation to be successful, you must determine where to place individual devices in order to establish reliable links throughout a network.

To measure the performance of a network, you can send unicast data through the network from one device to another to determine the success rate of several transmissions. To simplify link testing, the devices support a loopback cluster ID (0x12) on the data endpoint (0xE8). The cluster ID on the data endpoint sends any data transmitted to it back to the sender.

The following figure demonstrates how you can use the loopback cluster ID and data endpoint to measure the link quality in a mesh network.



The configuration steps for sending data to the loopback cluster ID depend on what mode the device is in. For details on setting the mode, see AP (API Mode). The following sections list the steps based on the device's mode.

Transparent operating mode configuration (AP = 0)

To send data to the loopback cluster ID on the data endpoint of a remote device:

1. Set the **CI** command to 0x12.
2. Set the **DH** and **DL** commands to the address of the remote device.

After you exit Command mode, the device transmits any serial characters it received to the remote device, which returns those characters to the sending device.

API operating mode configuration (AP = 1 or AP = 2)

Send an Explicit Addressing Command frame (0x11) using 0x12 as the cluster ID and 0xE8 as both the source and destination endpoint.

The remote device echoes back the data packets it receives to the sending device.

Test links between adjacent devices

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

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

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

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

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

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

Example

Suppose that you want to test the link between device A (**SH**/**SL** = 0x0013A200 40521234) and device B (**SH**/**SL**=0x0013A 200 4052ABCD) by transmitting 1000 40-byte packets:

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

In the following example packet, whitespace marks fields, bold text is the payload portion of the packet:

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

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

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

This means:

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

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

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

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

RSSI indicators

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

You can use the **DB** command to measure the RSSI on a device. **DB** returns the RSSI value measured in -dBm of the last packet the device received. This number can be misleading in multi-hop DigiMesh networks. The **DB** value only indicates the received signal strength of the last hop. If a transmission spans multiple hops, the **DB** value provides no indication of the overall transmission path, or the quality of the worst link, it only indicates the quality of the last link.

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

1. Use the RSSI module pin (pin 36). When the device receives data, it sets the RSSI PWM duty cycle to a value based on the RSSI of the packet it receives.

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

Discover devices

Discover all the devices on a network

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

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

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

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

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

Discover devices within RF range

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

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

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

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

General Purpose Flash Memory

XLR PRO RF Modules provide 119 512-byte blocks of flash memory that an application can read and write to. This memory provides a non-volatile data storage area that an application uses for many purposes. Some common uses of this data storage include:

- Storing logged sensor data
- Buffering firmware update data for a host microcontroller
- Storing and retrieving data tables needed for calculations performed by a host microcontroller.

The General Purpose Memory (GPM) is also used to store a firmware update file for over-the-air firmware updates of the device itself.

Access General Purpose Flash Memory

To access the GPM of a target node locally or over-the-air, send commands to the MEMORY_ACCESS cluster ID (0x23) on the DIGI_DEVICE endpoint (0xE6) of the target node using explicit API frames. For a description of Explicit API frames, see Operate in API mode.

To issue a GPM command, format the payload of an explicit API frame as follows:

Byte offset in payload	Number of bytes	Field name	General field description
0	1	GPM_CMD_ID	Specific GPM commands are described in detail in the topics that follow
1	1	GPM_OPTIONS	Command-specific options
2	2*	GPM_BLOCK_NUM	The block number addressed in the GPM
4	2*	GPM_START_INDEX	The byte index within the addressed GPM block
6	2*	GPM_NUM_BYTES	The number of bytes in the GPM_DATA field, or in the case of a READ, the number of bytes requested
8	varies	GPM_DATA	
* Specify multi-byte parameters with big-endian byte ordering.			

When a device sends a GPM command to another device via a unicast, the receiving device sends a unicast response back to the requesting device's source endpoint specified in the request packet. It does not send a response for broadcast requests. If the source endpoint is set to the DIGI_DEVICE endpoint (0xE6) or Explicit API mode is enabled on the requesting device, then the requesting node outputs a GPM response as an explicit API RX indicator frame (assuming it has API mode enabled).

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

Byte offset in payload	Number of bytes	Field name	General field description
0	1	GPM_CMD_ID	This field will be the same as the request field
1	1	GPM_STATUS	Status indicating whether the command was successful
2	2*	GPM_BLOCK_NUM	The block number addressed in the GPM
4	2*	GPM_START_INDEX	The byte index within the addressed GPM block
6	2*	GPM_NUM_BYTES	The number of bytes in the GPM_DATA field
8	varies	GPM_DATA	
* Specify multi-byte parameters with big-endian byte ordering.			

PLATFORM_INFO_REQUEST (0x00)

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

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

PLATFORM_INFO (0x80)

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

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

Example

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

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

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

7E 0007 8B 01 FFFE 00 00 00 76

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

ERASE (0x01)

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

Field name	Command-specific description
GPM_CMD_ID	Should be set to ERASE (0x01).

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

ERASE_RESPONSE (0x81)

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

Field name	Command-specific description
GPM_CMD_ID	Should be set to ERASE_RESPONSE (0x81).
GPM_STATUS	A 1 in the least significant bit indicates an error occurred. All other bits are reserved at this time.
GPM_BLOCK_NUM	Matches the parameter passed in the request frame.
GPM_START_INDEX	Matches the parameter passed in the request frame.
GPM_NUM_BYTES	The number of bytes in the GPM_DATA field. For this command, this field will be set to 0.
GPM_DATA	No data bytes are specified for this command.

Example

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

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

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

7E 0007 8B 01 FFFE 00 00 00 76

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

WRITE (0x02) and ERASE_THEN_WRITE (0x03)

The WRITE command writes the specified bytes to the specified GPM location. Before writing bytes to a GPM block, make sure all the bytes have first been erased. The ERASE_THEN_WRITE command performs an ERASE of the entire GPM block specified with the GPM_BLOCK_NUM field prior to doing a WRITE.

Field name	Description
GPM_CMD_ID	Set to WRITE (0x02) or ERASE_THEN_WRITE (0x03).
GPM_OPTIONS	At present, there are no defined options for this command. Set this field to 0.
GPM_BLOCK_NUM	Set to the index of the GPM block to be written.
GPM_START_INDEX	Set to the byte index within the GPM block where the data should be written.
GPM_NUM_BYTES	Set to the number of bytes specified in the GPM_DATA field. Only one GPM block can be operated on per command. For this reason, the GPM_START_INDEX plus the GPM_NUM_BYTES cannot be greater than the GPM block size. **Note** The number of bytes sent in an explicit API frame (including the GPM command fields) cannot exceed the maximum payload size of the radio. Use the ATNP command to query the maximum payload size.
GPM_DATA	Data to be written.

WRITE_RESPONSE (0x82) and ERASE_THEN_WRITE_RESPONSE (0x83)

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

Field name	Description
GPM_CMD_ID	Set to WRITE_RESPONSE (0x82) or ERASE_THEN_WRITE_ RESPONSE (0x83).
GPM_OPTIONS	A one (1) in the least-significant bit indicates an error occurred. All other bits are reserved.
GPM_BLOCK_NUM	Matches the parameter passed in the request frame.
GPM_START_INDEX	Matches the parameter passed in the request frame.
GPM_NUM_BYTES	Number of bytes in the GPM_DATA field. For this command, set to 0.
GPM_DATA	No data bytes should be specified for these commands.

Example:

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

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

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

7E 0007 8B 01 FFFE 00 00 00 76

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

READ (0x04)

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

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

READ_RESPONSE (0x84)

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

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

Example

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

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

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

7E 0007 8B 01 FFFE 00 00 00 76

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

FIRMWARE_VERIFY (0x05) and FIRMWARE_VERIFY_AND_INSTALL(0x06)

Use the FIRMWARE_VERIFY and FIRMWARE_VERIFY_AND_INSTALL commands when remotely updating firmware on a device. For more inoformation about firmware updates. These commands check if the GPM contains a valid over-the-air update file. For the FIRMWARE_VERIFY_AND_INSTALL command, if the GPM contains a valid firmware image then the device resets and begins using the new firmware.

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

FIRMWARE_VERIFY_RESPONSE (0x85)

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

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

Work with flash memory

When working with the General Purpose Memory, observe the following limitations:

- Flash memory write operations are only capable of changing binary 1s to binary 0s. Only the erase operation can change binary 0s to binary 1s. For this reason, you should erase a flash block before performing a write operation.
- When performing an erase operation, you must erase the entire flash memory block—you cannot erase parts of a flash memory block.
- Flash memory has a limited lifetime. The flash memory on which the GPM is based is rated at 20,000 erase cycles before failure. Take care to ensure that the frequency of erase/write operations allows for the desired product lifetime. Digi's warranty does not cover products that have exceeded the allowed number of erase cycles.
- Over-the-air firmware upgrades erase the entire GPM. Any user data stored in the GPM will be lost during an over-the-air upgrade.

Over-the-air (OTA) firmware updates

XLR PROs provide two methods of firmware update:

- Local firmware update via XCTU using the serial interface.
- Over-the-air firmware update using the RF interface.

The over-the-air firmware update method provides a robust and versatile technique which can be tailored to many different networks and applications, with minimum disruption of normal network operations.

Over-the-air firmware updates can be sent to a remote node using a local node and XCTU, or an external application can be programmed to follow this process. There are three phases of the over-the-air update process: distributing the new application, verifying the new application, and installing the new application. In the following section, the node to be updated is referred to as the target node. The node providing the update information is referred to as the source node. In most applications, the source node is locally attached to a PC running update software.

Distribute the new application

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

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

Example:

XLR PRO firmware version 1003 has an .ebin file of 1,048,576 bytes in length. Based on using a recommended packet size of 1024 bytes, sending a packet every 30 seconds minimized network disruption. For this reason, the .ebin should be divided and addressed as follows:

GPM_BLOCK_NUM	BPM_START_INDEX	BPM_NUM_BYTES	.ebin bytes
0	0	102	0 to 1023
0	1024	1024	1024 to 2047
0	2048	1024	2048 to 3071
0	3072	1024	3072 to 4095
1	0	1024	4096 to 5119
1	1024	1024	5120 to 6143
1	2048	1024	6144 to 7167
1	3072	1024	7168 to 8191
-	-	-	-
-	-	-	-
-	-	-	-
255	0	1024	1044480 to 1045503
255	1024	1024	1045504 to 1046527
255	2048	1024	1046528 to 1047551
255	3072	1024	1047552 to 1048575

Verify the new application

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

Install the application

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

Keep in mind

The firmware upgrade process requires that the XLR PRO resets itself and parameters which have not been written to flash will be lost after the reset. To avoid this, write all parameters with the **WR** (write) command before doing a firmware upgrade.

Because explicit API Tx frames can be addressed to a local node or a remote node (accessible over the RF port) the same process can be used to update firmware on an XLR PRO in either case.

Configure the XLR PRO RF Module using XCTU

Download and install XCTU

For XLR PRO module support, maker sure you install XCTU version 6.3.0 or later.

To download and install XCTU:

1. Go to www.digi.com/xctu.
2. Launch the XCTU installer and follow the prompts on the installation screens.

Connect XLR PRO to your PC

To connect XLR PRO to your PC:

1. Connect the XLR PRO serial interface to your PC.
2. Power on the XLR PRO.

Launch XCTU and add the XLR PRO

To launch XCTU and add the XLR PRO:

1. Double-click on the XCTU program icon. The XCTU main menu appears.
2. Click the Add a radio icon . The Add a radio device dialog appears.
3. Provide connection information for the XLR PRO:
   - **Select the Serial/USB port:** Select the USB port connected to the XLR PRO.
   - **Baud Rate:** Select the default, 9600.
   - **Data Bits:** Select the default, 8.
   - **Parity:** Select the default, None.
   - **Stop Bits:** Select the default 1.
   - **Flow Control:** Select the default, None.
   - **The radio module is programmable:** Keep the default, unselected.
4. Click **Finish**. XCTU connects to the XLR PRO and displays the device in the list of radios.
5. Click the XLR PRO to display current properties and configure parameters in the right-hand pane.

Configure parameters using XCTU

All of the XLR PRO parameter values are displayed in the XCTU configuration pane. For a complete list of all parameters, see AT commands.

To change a configuration parameter:

1. Locate the parameter in the XCTU configuration display.
2. Use the **Search** function in the upper right corner to quickly locate a parameter.
3. Select a new value for the parameter.

4. If you want to permanently change the parameter value:
   - To save an individual parameter value, click on the **Write** icon to the right of the parameter.
   - To save all parameter settings, click on the **Write** icon at the top of the XCTU configuration pane.

Update firmware with XCTU

To update XLR PRO firmware, you need XCTU version 6.3.0 (or above).

To update firmware using XCTU:

1. Launch XCTU.
2. Click **Add devices** or **Discover devices** to add the XLR PRO to the list of radios.
   - Select the COM port to which the XLR PRO serial interface is connected.
   - Select the baud rate (9600 8-N-1 by default).
   - Switch baud rate to 115200b/s to reduce the time required to update the firmware.
   - Close the COM port and reopen it at the new baud rate.
3. Select the radio configuration tab.
4. Click the icon to download the firmware.
5. Select the desired firmware, function set, and firmware version.
6. Click **Finish** and then the **Yes**.

Safety notices and certifications

Before installing and powering on the XLR PRO, read all instructions and keep these instructions in a safe place for future reference.

Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

Do not attempt to repair the product. Any attempt to service or repair the unit by the user will void the product warranty.

The XLR PRO must be maintained by Digi or a Digi qualified technician only.

RF exposure statement

The XLR PRO, when used with approved antennas, complies with the FCC and IC certifications detailed in this section. For a list of antennas approved for use with XLR PRO, see XLR PRO antennas.

To comply with RF exposure limits established in the ANSI C95.1 standards, the distance between the antenna or antennas and the user should not be less than 25 cm (10 inches) for USA and 34 cm for Canada.

FCC (United States) certification

The XLR PRO 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 operate under Digi International FCC Certification, RF modules/integrators must comply with the following regulations:

1. The system integrator must ensure that the text provided with this device (see FCC required label text) is placed on the outside of the final product and within the final product operation manual.
2. The XLR PRO RF module may be used only with antennas that have been tested and approved for use with this module. See XLR PRO antennas.

FCC labeling requirements

The Original Equipment Manufacturer (OEM) must ensure that FCC labeling requirements are met. This includes a clearly visible label on the outside of the final product enclosure that displays the text shown in FCC required label text.

FCC required label text

Contains FCC ID: MCQ-XLRP

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 XLR PRO OEM RF Module has been certified by the FCC for use with other products without any further certification (as per FCC section 2.1091). Modifications not expressly approved by Digi International could void the user's authority to operate the equipment.

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 limited modular approval

This is an RF module approved for Limited Modular use operating as a mobile transmitting device with respect to section 2.1091 and is limited to OEM installation for Mobile and Fixed applications only. During final installation, end-users are prohibited from access to any programming parameters. Professional installation adjustment is required for setting module power and antenna gain to meet EIRP compliance for high gain antenna(s).

Final antenna installation and operating configurations of this transmitter including antenna gain and cable loss must not exceed the EIRP of the configuration used for calculating MPE. Grantee (Digi) must coordinate with OEM integrators to ensure the end-users and installers of products operating with the module are provided with operating instructions to satisfy RF exposure requirements.

The FCC grant is valid only when the device is sold to OEM integrators. Integrators are instructed to ensure the end-user has no manual instructions to remove, adjust or install the device.

FCC-approved antennas

This device has been tested with MMCX connectors with the antennas listed in XLR PRO antennas. When integrated into OEM products, fixed antennas require installation preventing end-users from replacing them with non-approved antennas. Antennas not listed in the XLR PRO antennas must be tested to comply with FCC Section 15.203 (unique antenna connectors) and Section 15.247 (emissions).

The FCC requires that all spread spectrum devices operating within the Unlicensed radio frequency bands must limit themselves to a maximum radiated power of 4 Watts EIRP. Failure to observe this limit is a violation of our warranty terms, and shall void the user’s authority to operate the equipment. This can be stated as follows:

RF power - cable loss + antenna gain <= 36 dBm eirp

Fixed base station and mobile applications

Digi RF Modules are pre-FCC approved for use in fixed base station and mobile applications. When the antenna is mounted at least 25 cm (10 inches) from nearby persons, the application is considered a mobile application.

Portable applications and SAR testing

If the module will be used at distances closer than 25 cm (10 inches) to all persons, the device may be required to undergo SAR testing. Co-location with other transmitting antennas closer than 25 cm should be avoided.

RF exposure

The following statement must be included as a CAUTION statement in OEM product manuals.

This equipment is approved for mobile and base station transmitting devices only. Antenna(s) used for this transmitter must be installed to provide a separation distance of at least 25 cm (or 10 inches) from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter

IC (Industry Canada) certification

This device complies with Industry Canada licence-exempt RSS standard(s). Operation is subject to the following two conditions: (1) this device may not cause interference, and (2) this device must accept any interference, including interference that may cause undesired operation of the device.

Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes: (1) l'appareil ne doit pas produire de brouillage, et (2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

RF exposure

CAUTION: This equipment is approved for mobile and base station transmitting devices only. Antenna(s) used for this transmitter must be installed to provide a separation distance of at least 25 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter.

ATTENTION: Cet équipement est approuvé pour la mobile et la station base dispositifs d'émission seulement. Antenne(s) utilisé pour cet émetteur doit être installé pour fournir une distance de séparation d'au moins 25 cm à partir de toutes les personnes et ne doit pas être situé ou fonctionner en conjonction avec tout autre antenne ou émetteur.

IC labeling requirements

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

IC required text

Contains IC:1846A-XLRP

The integrator is responsible for its product to comply with IC ICES-003 & FCC Part 15, Sub. B-Unintentional Radiators. ICES-003 is the same as FCC Part 15 Sub. B and Industry Canada accepts FCC test report or CISPR 22 test report for compliance with ICES-003.

Transmitters with detachable antennas

This radio transmitter (IC: 1846A-XLRP) has been approved by Industry Canada to operate with the antenna types listed in XLR PRO antennas with the maximum permissible gain and required antenna impedance for each antenna type indicated. Antenna types not included in this list, having a gain greater than the maximum gain indicated for that type, are strictly prohibited for use with this device.

Le présent émetteur radio (IC: 1846A-XLRP) a été approuvé par Industrie Canada pour fonctionner avec les types d'antenne énumérés ci?dessous et ayant un gain admissible maximal et l'impédance requise pour chaque type d'antenne. Les types d'antenne non inclus dans cette liste, ou dont le gain est supérieur au gain maximal indiqué, sont strictement interdits pour l'exploitation de l'émetteur.

Detachable antenna

Under Industry Canada regulations, this radio transmitter may operate using only an antenna of a type and maximum (or lesser) gain approved for the transmitter by Industry Canada. To reduce potential radio interference to other users, the antenna type and its gain should be so chosen that the equivalent isotropically radiated power (e.i.r.p.) is not more than that necessary for successful communication.

Conformément à la réglementation d'Industrie Canada, le présent émetteur radio peutfonctionner avec une antenne d'un type et d'un gain maximal (ou inférieur) approuvépour l'émetteur par Industrie Canada. Dans le but de réduire les risques de brouillageradioélectrique à l'intention des autres utilisateurs, il faut choisir le type d'antenne etson gain de sorte que la puissance isotrope rayonnée équivalente (p.i.r.e.) ne dépassepas l'intensité nécessaire àl'établissement d'une communication satisfaisante.

Australia (RCM) [pending]

Power requirements

Regulations in Australia stipulate a maximum of 30 dBm EIRP (Effective Isotropic Radiated Power). The EIRP equals the sum (in dBm) of power output, antenna gain and cable loss and cannot not exceed 30 dBm.

The EIRP formula for Australia is:

power output + antenna gain - cable loss <= 30 dBm

Note The maximum EIRP for the FCC (United States) and IC (Canada) is 36 dBm.

The XLR PRO modules comply with the requirements to be used in end products in Australia. All products with EMC and radio communications must have registered RCM marks. Registration to use the compliance mark will only be accepted from Australian manufacturers or importers, or their agents.

In order to have a RCM mark on an end product, a company must comply with a or b below.

a. have a company presence in Australia.

b. have a company/distributor/agent in Australia that will sponsor the importing of the end product.

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

XLR PRO antennas

Omni-directional antennas

Part number	Type	Connector	Gain (dBi)	Application	Minimum cable loss or TX power reduction required in dB
A09-F0	OMNI	RPN	0.0	Fixed	0
A09-F1	OMNI	RPN	1.0	Fixed	0
A09-F2	OMNI	RPN	2.1	Fixed	0
A09-F3	OMNI	RPN	3.1	Fixed	0
A09-F4	OMNI	RPN	4.1	Fixed	0
A09-F5	OMNI	RPN	5.1	Fixed	0
A09-F6	OMNI	RPN	6.1	Fixed	0.1
A09-F7	OMNI	RPN	7.1	Fixed	1.1
A09-F8	OMNI	RPN	8.1	Fixed	2.1
A09-W7	OMNI	RPN	7.1	Fixed	1.1
A09-F0	OMNI	RPSMA	0.0	Fixed	0
A09-F1	OMNI	RPSMA	1.0	Fixed	0
A09-F2	OMNI	RPSMA	2.1	Fixed	0
A09-F3	OMNI	RPSMA	3.1	Fixed	0
A09-F4	OMNI	RPSMA	4.1	Fixed	0
A09-F5	OMNI	RPSMA	5.1	Fixed	0
A09-F6	OMNI	RPSMA	6.1	Fixed	0.1
A09-F7	OMNI	RPSMA	7.1	Fixed	1.1
A09-F8	OMNI	RPSMA	8.1	Fixed	2.1
A09-M7	OMNI	RPSMAF	7.2	Fixed	1.2
A09-W7SM	OMNI	RPSMA	7.1	Fixed	1.1
A09-F0TM	OMNI	RPTNC	0.0	Fixed	0
A09-F1TM	OMNI	RPTNC	1.0	Fixed	0
A09-F2TM	OMNI	RPTNC	2.1	Fixed	0
A09-F3TM	OMNI	RPTNC	3.1	Fixed	0
A09-F4TM	OMNI	RPTNC	4.1	Fixed	0
A09-F5TM	OMNI	RPTNC	5.1	Fixed	0

Part number	Type	Connector	Gain (dBi)	Application	Minimum cable loss or TX power reduction required in dB
A09-F6TM	OMNI	RPTNC	6.1	Fixed	0.1
A09-F7TM	OMNI	RPTNC	7.1	Fixed	1.1
A09-F8TM	OMNI	RPTNC	8.1	Fixed	2.1
A09-W7TM	OMNI	RPTNC	7.1	Fixed	1.1
A09-HSM-7	OMNI	RPSMA	3.0	Fixed/mobile	0
A09-HASM-675	OMNI	RPSMA	2.1	Fixed/mobile	0
A09-HABMM-P61	OMNI	MMCX	2.1	Fixed/mobile	0
A09-HABMM-6-P61	OMNI	MMCX	2.1	Fixed/mobile	0
A09-HBMM-P61	OMNI	MMCX	2.1	Fixed/mobile	0
A09-HRSM	OMNI	RPSMA	2.1	Fixed	0
A09-HASM-7	OMNI	RPSMA	2.1	Fixed	0
A09-HG	OMNI	RPSMA	2.1	Fixed	0
A09-HATM	OMNI	RPTNC	2.1	Fixed	0
A09-HATM-10	OMNI	RPTNC	2.1	Fixed/mobile	0
A09-H	OMNI	RPSMA	2.1	Fixed	0
A09-HBMMP61	OMNI	MMCX	2.1	Fixed/mobile	0
A09-QBMMP61	OMNI	MMCX	1.9	Fixed/mobile	0
A09-QSM-3	OMNI	RPSMA	1.9	Fixed/mobile	0
A09-QSM-3H	OMNI	RPSMA	1.9	Fixed/mobile	0
A09-QBMM-P61	OMNI	MMCX	1.9	Fixed/mobile	0
		Max gain	8.1		

Yagi antennas

Part number	Type	Connector	Gain (dBi)	Application	Minimum cable loss or TX power reduction required in dB
A09-Y6	2-Element Yagi	RPN	6.1	Fixed/Mobile	0.1
A09-Y7	3-Element Yagi	RPN	7.1	Fixed/Mobile	1.1
A09-Y8	4-Element Yagi	RPN	8.1	Fixed/Mobile	2.2

Part number	Type	Connector	Gain (dBi)	Application	Minimum cable loss or TX power reduction required in dB
A09-Y9	4-Element Yagi	RPN	9.1	Fixed/Mobile	3.1
A09-Y10	5-Element Yagi	RPN	10.1	Fixed/Mobile	4.1
A09-Y11	6-Element Yagi	RPN	11.1	Fixed/Mobile	5.1
A09-Y12	7-Element Yagi	RPN	12.1	Fixed/Mobile	6.1
A09-Y13	9-Element Yagi	RPN	13.1	Fixed/Mobile	7.1
A09-Y14	10-Element Yagi	RPN	14.1	Fixed/Mobile	8.1
A09-Y14	12-Element Yagi	RPN	14.1	Fixed/Mobile	8.1
A09-Y15	13-Element Yagi	RPN	15.1	Fixed/Mobile	9.1
A09-Y15	15-Element Yagi	RPN	15.1	Fixed/Mobile	9.1
A09-Y6TM	2-Element Yagi	RPTNC	6.1	Fixed/Mobile	0.1
A09-Y7TM	3-Element Yagi	RPTNC	7.1	Fixed/Mobile	1.1
A09-Y8TM	4-Element Yagi	RPTNC	8.1	Fixed/Mobile	2.1
A09-Y9TM	4-Element Yagi	RPTNC	9.1	Fixed/Mobile	3.1
A09-Y10TM	5-Element Yagi	RPTNC	10.1	Fixed/Mobile	4.1
A09-Y11TM	6-Element Yagi	RPTNC	11.1	Fixed/Mobile	5.1
A09-Y12TM	7-Element Yagi	RPTNC	12.1	Fixed/Mobile	6.1
A09-Y13TM	9-Element Yagi	RPTNC	13.1	Fixed/Mobile	7.1
A09-Y14TM	10-Element Yagi	RPTNC	14.1	Fixed/Mobile	8.1
A09-Y14TM	12-Element Yagi	RPTNC	14.1	Fixed/Mobile	8.1
A09-Y15TM	13-Element Yagi	RPTNC	15.1	Fixed/Mobile	9.1
A09-Y15TM	15-Element Yagi	RPTNC	15.1	Fixed/Mobile	9.1
		Max gain	15.1		