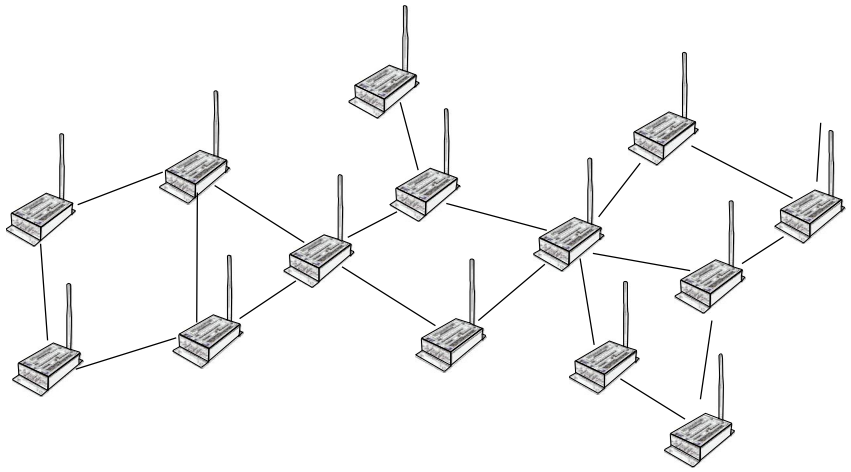
MeshX™ [Firmware Version 801(x)]
MaxStream Wireless Mesh Networking

Technical Support:
Live Chat: www.maxstream.net
E-mail: rf-xperts@maxstream.net
Phone: (801) 765-9885

Introduction

XTend OEM RF Modules containing firmware version 8010 (or above) now feature MeshX™ mesh networking support. Mesh networking allows message to be routed through several different nodes 9XTend nodes to a final destination. The MeshX firmware allows OEMs and system integrators to bolster their networks with the self-healing attributes of mesh networking. In the event that one RF connection between nodes is lost (due to power-loss, environmental obstructions, etc.) critical data can still reach its destination due to the mesh networking capabilities embedded inside the modules.

Figure 1. A Sample MeshX Network Topology



MeshX Feature Set

XTend OEM RF Modules containing firmware version 8010 (or above) support the following features:

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

Note that Sleep (low power) modes and encryption are not supported in this beta release.

Serial Communications

The XTend OEM RF Modules interface to a host device through a TTL-level asynchronous serial port. Through its serial port, the module can communicate with any UART voltage compatible device or through a level translator to any serial device (For example: RS-232/485/422 or USB interface board).

Transparent Operation

When XTend RF Modules operate in Transparent Mode ((AP (API Mode) parameter = 0), the modules act as a serial line replacement - all UART data received through the DI pin is queued up for RF transmission. When RF data is received, the data is sent out the DO pin.

When the RO (Packetization Timeout) parameter threshold is satisfied, the module attempts to initialize an RF transmission with the data that has been received. If the module cannot immediately transmit (for instance, if it is already receiving RF data), the serial data continues to be stored in the DI Buffer. Data is assembled in a packet and sent at any RO timeout or whenever RB bytes are received, whichever comes first.

The module operates as described above unless the Command Mode Sequence is detected. The Command Mode Sequence consists of three copies of the command sequence character [CC parameter] surrounded by the before and after guard times [BT & AT parameters].

Refer to the "AT Commands" section [p6] for a list of supported AT commands.

API Operation

When AP is 1 or 2, the XTend RF Modules operate using API (Application Programming Interface) communications. The API is frame-based and extends the level to which a host application can interact with the networking capabilities of the module. When in API mode, all data entering and leaving the RF module is contained in frames that define operations or events within the module.

Transmit Data Frames (received through the DI (Data In) pin) include:

- 64-bit address

Receive Data Frames (sent out the DO (Data Out) pin) include:

- Showing a received RF packet (64 bits only)
- Response to a TX (Transmit) packet
- Showing events such as hardware reset, watchdog reset, asynchronous events, etc.

The module will send data frames to the application containing status packets; as well as source, RSSI and payload information from received data packets.

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

-> Change destination addresses without having to enter command mode
-> Receive success/failure status of each RF packet
-> Identify the source address of each received packet

Refer to the "API Operation" section [p13] for a description of API operation.

Data Transmission

Unicast Addressing

When transmitting while using Unicast communications, reliable delivery of data is accomplished using retries and acknowledgements. RF data packets are sent up to NR + 1 times and ACKs (acknowledgements) are transmitted by the receiving module upon receipt. The number of retries is determined by the NR (Network Retries) parameter.

Refer to the DL (Destination Address Low) and DH (Destination Address High) parameters for information on how to configure a module to operate using Unicast addresses.

All transmissions are addressed at the MAC layer.

When a new Unicast is given to the MAC layer for transmission, the following will occur:

1. The MAC header is pre-pended.
2. If incomplete transmissions precede it with the same destination address, the RF data packet is placed on a pending queue.

When a transmission is begun the following happens :

1. Start the transmission, sending the preamble (10 bits) with the necessary number of repetitions. (For an XTend Module transmitting at 125kbps, the preamble is sent 66 times.) The amount of time required for synchronizing with the receiving module is 5 msec.
2. Send the data, beginning with MAC header and ending with a 16 bit CRC
3. Wait for a MAC layer acknowledgement for a fixed amount of time. (For XTend, the wait is 1.8ms.)
4. If the acknowledgement is received the transmission has successfully completed.
5. Otherwise, start a timer after which a retry will be attempted.

The length of the retry timer started in step 3 is the subject of random exponential backoffs.

Random Exponential Back off

The back-off is random because the number of delay slots (RN parameter) between retries (RR parameter) is random. The backoff is exponential because the range of the number of the random number of delay slots doubles with each retry. Note that the randomness allows the backoff time to decrease from one retry to the next. However, because of the exponent, the backoff can quickly grow very large.

The number of time slots when the transmission can occur doubles with each retry; but the actual time between retries may be more or less than double the previous retry.

Broadcast Addressing

When operating in Broadcast Mode, reliable delivery of RF data packets is accomplished using multiple transmissions. When transmitting in Broadcast mode, ACKs are not returned upon receipt of an RF data packet. Retries don't apply to broadcasts because no acknowledgements will be used.

The delay between retransmissions is a random number of delay slots in the range between 0 and RN (“Delay Slots” parameter). After a Broadcast is sent RR + 1 times, a function will be called to indicate a successful transmission.

Refer to the DL (Destination Address Low) and DH (Destination Address High) parameters for information on how to configure a module to operate using Broadcast addresses.

9XTend Mesh Networking

Routing

A module within a MeshX network is able to determine reliable routes using a routing algorithm and table. The routing algorithm uses a reactive method derived from AODV (Ad-hoc On-demand Distance Vector). An associative routing table is used to map a destination node address with its next hop. By sending a message to the next hop address, either the message will reach its destination or be forwarded to an intermediate node which will route the message on to its destination.

A message with a Broadcast address is broadcast to all neighbors. All receiving neighbors will rebroadcast the message and eventually the message will reach all corners of the network. Packet tracking prevents a node from resending a broadcast message twice.

A message with a Unicast destination node address is looked up in an associative routing table. If the destination address is not found and the message came here from a neighboring node; then a routing error has occurred and the undeliverable message is dropped. An ACK timeout will eventually occur at the source node and route discovery (RD) will be launched to re-establish the route.

If the message originated with this node and RD is already underway to discover a route to the destination; then the message is saved until RD is completed.

If no route discovery is underway and the route to the destination is unavailable, then the message is saved and RD is launched to establish a route to the destination. When route discovery is over, the routing table will be updated and the message relayed.

Route Discovery

If the source node doesn't have a route to the requested destination, the packet is queued to await a route discovery (RD) process.

RD begins by the source node broadcasting a route request (RREQ).

Any node that receives the RREQ that is not the ultimate destination is called an intermediate node. Intermediate nodes may either drop or forward a RREQ, depending on whether the new RREQ has a better route back to the source node. If so, information from the RREQ is saved and the RREQ is updated and broadcast.

When the ultimate destination receives the RREQ, it unicasts a route reply (RREP) back to the source node along the path of the RREQ. This is done regardless of route quality and regardless of how many times an RREQ has been seen before.

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

RF Module Configuration

Two command mode protocols are supported by this MeshX version of the 9XTend RF Module:

- **AT Command Mode - Printable protocol that is intended for manual entry of commands and viewing parameter values.**
- **API Operation - Binary protocol intended for programmatic transmissions and receptions of data packets. For example, using API mode, sequential packets can be sent to different addresses without having to escape into command mode and change DL between each transmission.**

AT Commands

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

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

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

Method 1 (One line per command)

Send AT Command	System Response
+++	OK <CR> (Enter AT Command Mode)
ATDL <Enter>	{current value} <CR> (Read Destination Address Low)
ATDL00001A0D <Enter>	OK <CR> (Modify Destination Address Low)
ATWR <Enter>	OK <CR> (Write to non-volatile memory)
ATCN <Enter>	OK <CR> (Exit Command Mode)

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

AT Command Reference Table

9XTend RF Modules expect numerical values in hexadecimal. Hexadecimal values are designated by a "0x" prefix. Decimal equivalents are designated by a "d" suffix.

Tabel 1. 9XTend AT Commands (as of firmware version 8010)
(Special)

AT Command	AT Command Name	Parameter Range	Command Category	# Bytes Returned	Factory Default
PL	TX Power Level. Set/Read the power level at which the RF module transmits data	0 – 4 0 = 1 mW 1 = 10 mW 2 = 100 mW 3 = 500 mW 4 = 1000 mW (1 Watt)	RF Interfacing	1	4 (1 Watt)
R1	Restore Compiled. Restore module parameters to compiled defaults.	--	(Special)	--	--
RE	Restore Defaults. Restore module parameters to custom defaults.	--	(Special)	--	--
WR	Write. Write configurable parameters to non-volatile memory	--	(Special)	--	--
FR	Force Reset. Force module to take a physical reset.	--	(Special)	--	--

Tabel 2. 9XTend AT Commands (as of firmware version 8010)
Networking

AT Command	AT Command Name	Parameter Range	Command Category	# Bytes Returned	Factory Default
DH	Destination Address High. Set/Read the destination address (high 32 bits) of a module.	0 - 0xFFFFFFFF	Networking	4	0x0013A200
DL	Destination Address Low. Set/Read the destination address (low 32 bits) of a module.	0 - 0xFFFFFFFF	Networking	4	0x00000000
HP	Hopping Channel. Set/Read the channel hopping sequence. Nodes must have the same hopping sequence to communicate.	0 - 9	Networking	1	0
ID	Network Address. Set/Read the user network address. Nodes must have the same network address to communicate.	0x10 - 0x7FFF	Networking	2	0x3332
NH	Network Hops. Set/Read the maximum number of hops expected in a network route. This value doesn't limit the number of hops allowed, but it is used to calculate timeouts waiting for network acknowledgements.	0 – 0xFF [Max number of hops]	Networking	2	7
NN	Network Delay Slots. Set/Read the maximum number of network delay slots before re-broadcasting a network packet.	0 – 0x10	Networking	2	3
NQ	Network Route Requests. Set/Read the maximum number of route discovery retries allowed to find a path to the destination node. If NQ = 0, a route request will only be sent once.	0 – 0x0A	Networking	2	3
NR	Network Retries. Set/Read the maximum number of network packet delivery attempts. If NR > 0, packets sent will request a network ACK and can be resent up to NR+1 times if no ACKs are received.	0 – 0xFF	Networking	2	1
SH	Source Address High. Set/Read the source address (high 32 bits) of a module.	0x0013A200 [read-only]	Networking	2	0x0013A200
SL	Source Address Low. Set/Read the source address (low 32 bits) of a module.	0 - 0xFFFFFFFF [read-only]	Networking	2	varies

Tabel 3. 9XTend AT Commands (as of firmware version 8010)
Command Mode Options

AT Command	AT Command Name	Parameter Range	Command Category	# Bytes Returned	Factory Default
AT	Guard Time After. Set/Read required DI pin silent time after the Command Sequence Characters of the Command Mode Sequence. The DI silent time is used to prevent inadvertent entrance into Command Mode.	0 – 0xFFFF [x 100 msec]	Command Mode Options	2	0x0A (1 decimal second)
BT	Guard Time Before. Set/Read required DI pin silent time before the Command Sequence Characters of the Command Mode Sequence. The DI silent time is used to prevent inadvertent entrance into Command Mode.	0 - 0xFFFF [x 100 msec]	Command Mode Options	2	0x0A (10d)
CC	Command Sequence Character. Set/Read the ASCII character used between guard times of the AT Command Mode Sequence (BT + CC + AT)	0x20 - 0x7F	Command Mode Options	1	0x2B [ASCII "+"]
CN	Exit Command Mode. Explicitly exit the module from AT Command Mode. (The same action occurs automatically when CT expires.)	--	Command Mode Options	--	--
CT	Command Mode Timeout. Set/Read the amount of inactive time that elapses before the module automatically exits from AT Command Mode.	2 - 0xFFFF [x 100 ms]	Command Mode Options	2	0xC8 (200d)
E0	Echo Off. Turn off character echo in AT Command Mode. By default, echo is off.	--	Command Mode Options	--	--
E1	Echo On. Turn on character echo in AT Command Mode. Each input character is echoed back to out to the host.	--	Command Mode Options	--	--

Tabel 4. 9XTend AT Commands (as of firmware version 8010)
Diagnostics

AT Command	AT Command Name	Parameter Range	Command Category	# Bytes Returned	Factory Default
%V	Board Voltage	0x2CCA to 0x5BFFA	Diagnostics	4	--
CF	Command Format. Set/Read the format of data entered and displayed for commands. Use decimal format unless Hex is forced or preferred.	0 – 2 0 = Decimal with units 1 - Hexadecimal without units. All input and output is in hexadecimal format. 2 - Decimal without units.	Diagnostics	1	1
DB	Received Signal Strength. Read the receive signal strength (in decibels relative to milliWatts) of the last received packet.	0x6E - 0x28 [read-only] Sample Output: -88 dBm (when ATCF = 0) 58 (when ATCF = 1) -88 (when ATCF = 2)	Diagnostics	2	--
ER	Receive Error Count. Set/Read the number of receive-errors.	0 - 0xFFFF	Diagnostics	2	0
GD	Receive Good Count. Set/Read the count of good received RF packets.	0 - 0xFFFF	Diagnostics	2	0
HV	Hardware Version Read and display the version of the hardware	0 – 0Xffff	Diagnostics	2	--
RC	Ambient Power - Single Channel. Examine & report the power level on a given channel.	0 - 0x31 [dBm, read-only] Sample output: -78 dBm [when CF = 0] 4e [when CF = 1] -78 [when CF = 2]	Diagnostics	1	--
RM	Ambient Power - All Channels. Examine and report power levels on all channels.	No parameter - 0x7D0	Diagnostics	2	--
RP	RSSI PWM Timer. Enable PWM ("Pulse Width Modulation") output on the Config/RSSI pin (pin 11 of the OEM RF Module)	0 - 0xFF [x 100 msec]	Diagnostics	1	0x20 (32d)
TP	Board Temperature. Read the current temperature of the board.	0 - 0x7F [read-only]	Diagnostics	1	--
TR	Delivery Failure Count. Report the number of retransmit failures.	0 - 0xFFFF [read-only]	Diagnostics	2	0
VL	Firmware Version – verbose. Read detailed version information including application build date and time.	Returns string	Diagnostics	--	--
VR	Firmware Version. Read the 4-digit version number.	0 - 0xFFFF [read-only]	Diagnostics	2	--
WA	Active Warning Numbers. Report the warning numbers of all active warnings - one warning number per line.	Returns string	Diagnostics	--	--
WN	Warning Data. Report data for all active and sticky warnings	Returns string	Diagnostics	--	--
WS	Sticky Warning Numbers. Report warning numbers of all warnings active since the last use of the WS or WN command	Returns string	Diagnostics	--	--

Tabel 5. 9XTend AT Commands (as of firmware version 8010)
Serial Interfacing

AT Command	AT Command Name	Parameter Range	Command Category	# Bytes Returned	Factory Default
AP	API Enable. Set/Read the API mode of the radio.	0 – 2 0 = API Disabled 1 = API-enabled 2 = API-enabled (w/escaped control characters)	Serial Interfacing	1	0
BD	Interface Data Rate. Set/Read the serial interface data rate (baud rate) used between the RF module and host.	0 - 8 (standard rates) 0 = 1200 bps 1 = 2400 2 = 4800 3 = 9600 4 = 19200 5 = 38400 6 = 57600 7 = 115200 8 = 230400 0x39 - 0x1C9C38 (non-standard rates)	Serial Interfacing	4	3 (9600 baud)
CD	GPO2 Configuration. Select/Read the behavior of the GPO2 line (pin 3).	0 – 4 0 = RX LED (when data is received whether or not the address is valid.) 1 = Assert RX LED 2 = De-assert RX LED 3 = (reserved) 4 = RX LED (valid address only)	Serial Interfacing	1	2
CS	GPO1 Configuration. Select/Read the behavior of the GP01 pin (pin 9)	0 – 4 0 = RS-232 CTS flow control 1 = RS-485 TX enable low 2 = CTS always High 3 = RS-485 TX enable high 4 = CTS always Low	Serial Interfacing	1	0
FL	Software Flow Control. Enable/Disable software flow control (XON/XOFF).	0 – 1 0 = Disabled 1 = Enabled	Serial Interfacing	1	0
FT	Flow Control Threshold. Set/Read the flow control threshold. When FT bytes have accumulated in the DI buffer (UART Receive), CTS is de-asserted or the XOFF software flow control character is transmitted.	0x10 – 0x17E [Bytes]	Serial Interfacing	2	0x16D (365 decimal)
NB	Parity. Select/Read parity settings.	0 – 4 0 = No parity 1 = 8-bit even 2 = 8-bit odd 3 = Forced high 4 = Forced low	Serial Interfacing	1	0
RB	Packetization Threshold. Set/Read the character threshold value. RF transmission begins after receiving RB bytes, or after receiving at least 1 byte and detecting RO character times of silence on the UART.	0 – 0xD3 [Bytes]	Serial Interfacing	2	0xC8 (200 decimal)
RO	Packetization Timeout. Set/Read the number of character times with no UART data before a packet is created for RF output (assuming UART data was received prior to the idle time). If RO = 0, it is ignored and no data will be transmitted until RB characters are in the DO buffer.	0 - 0xFFFF [x UART character time]	Serial Interfacing	2	3
RT	GPI1 Configuration. Set/Read the behavior of the GPI1 pin (pin 10).	0 – 2 0 = No RTS flow control 2 = RTS flow control	Serial Interfacing	1	0
SB	Stop Bits. Set/Read the number of stop bits in the data packet.	0 – 1 0 = 1 stop bit 1 = 2 stop bits	Serial Interfacing	1	0

API Operation

API operation requires that communication with the module be done through a structured interface (data is communicated in frames in a defined order). The API specifies how commands, command responses and module status messages are sent and received from the module using a UART data Frame.

API Frame Specifications

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

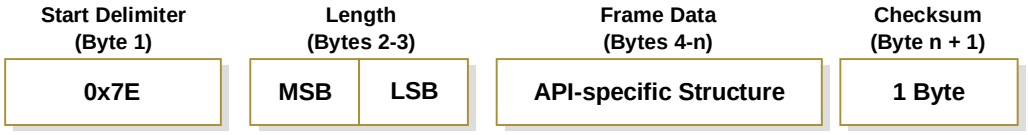
- **AP = 0 (default): Transparent Operation (UART Serial line replacement)**
  API modes are disabled.
- **AP = 1: API Operation**
- **AP = 2: API Operation (with escaped characters)**

Any data received prior to the start delimiter is silently discarded. If the frame is not received correctly or if the checksum fails, the data is silently discarded.

API Operation (AP parameter = 1)

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

Figure 2. UART Data Frame Structure:

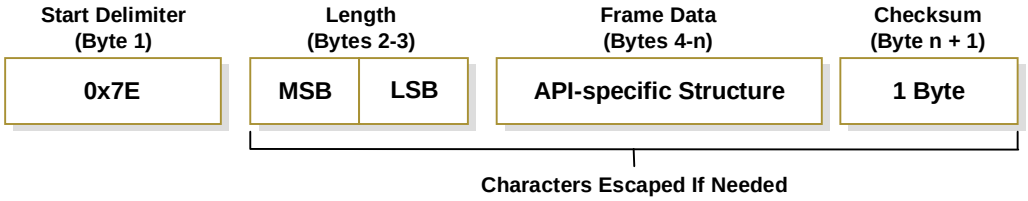


MSB = Most Significant Byte, LSB = Least Significant Byte

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

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

Figure 3. UART Data Frame Structure - with escape control characters:



MSB = Most Significant Byte, LSB = Least Significant Byte

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

Data bytes that need to be escaped:

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

Example - Raw UART Data Frame (before escaping interfering bytes):
0x7E 0x00 0x02 0x23 0x11 0xCB

0x11 needs to be escaped which results in the following frame:
0x7E 0x00 0x02 0x23 0x7D 0x31 0xCB

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

Checksum

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

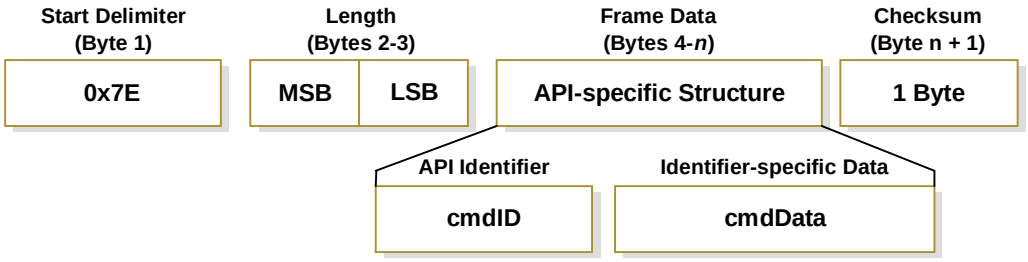
To calculate: Not including frame delimiters and length, add all bytes keeping only the lowest 8 bits of the result and subtract from 0xFF.

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

API Types

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

Figure 4. UART Data Frame & API-specific Structure:



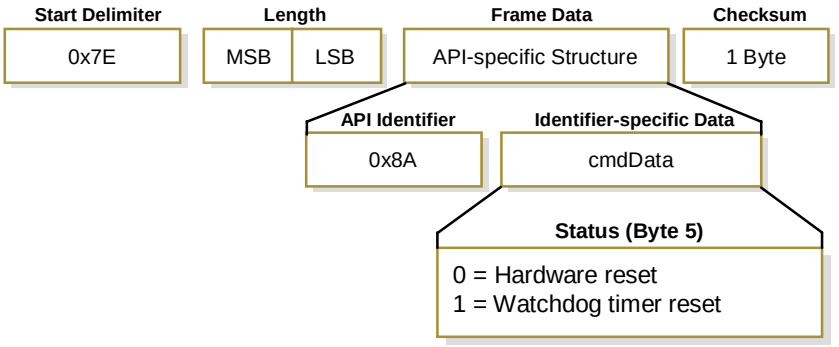
The cmdID frame (API-identifier) indicates which API messages will be contained in the cmdData frame (Identifier-specific data). Refer to the sections that follow for more information regarding the supported API types. Note that multi-byte values are sent big endian.

RF Module Status

API Identifier: 0x8A

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

Figure 5. RF Module Status Frames

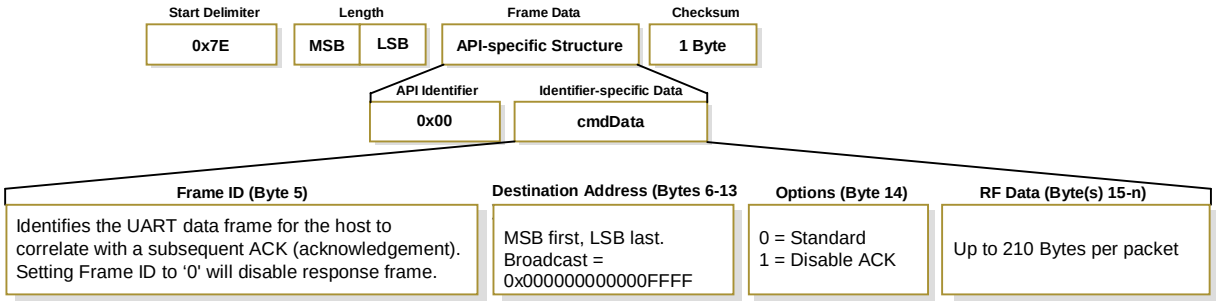


TX (Transmit) Request: 64-bit address

API Identifier Value: 0x00

A TX Request message will cause the module to send RF Data as an RF Packet.

Figure 6. TX Packet (64-bit address) Frames

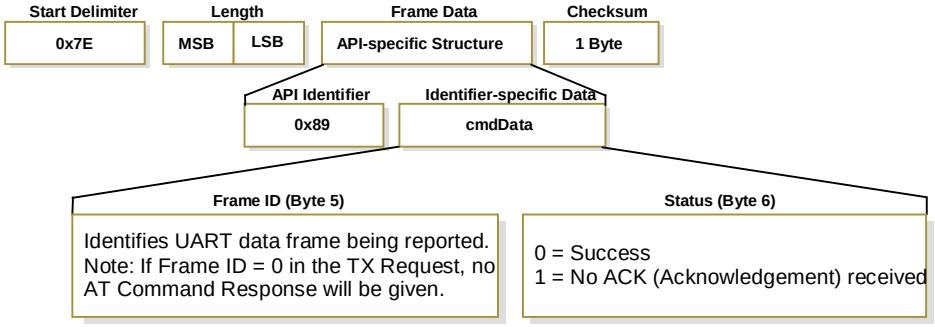


TX (Transmit) Status

API Identifier Value: 0x89

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

Figure 7. TX Status Frames



NOTE: "STATUS = 1" occurs when all retries are expired and no ACK is received.

RX (Receive) Packet: 64-bit address

API Identifier Value: 0x80

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

Figure 8. RX Packet (16-bit address) Frames