LoRa Software Modem Command Reference Manual

Table of Contents

Table of Figures

Table of Tables

1 Introduction

The LoRa Software Modem (also referred to as Modem) is a software set running on a LoRa based module to provide an API for end-node operation.

The LoRa Software Modem includes the LoRaWAN MAC layer functions, and some application layer functions which can be consumed via the Device and Application Services in the Semtech LoRa Cloud Service (loracloud.com).

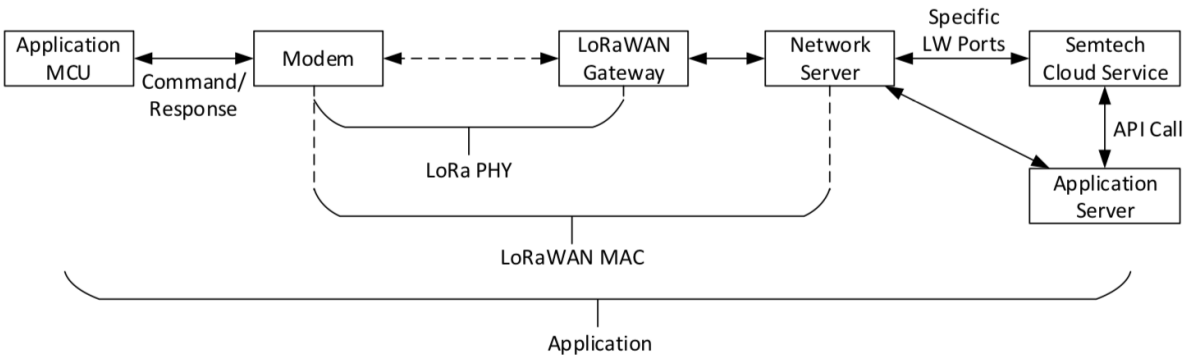

Figure 1 Scope of the Modem and Modem service

2 Murata LoRa Module Pinout

Table 1. Murata LoRa Module Pin Description

Pin No.	Terminal Name	Type	Description
1	PA12/USB_DP	O	TCXO power supply output. Must be externally connected to pin 48.
2	PA11/USB_DM	I/O	
3	GND	GND	Ground.
4	VDD_USB	PWR	Power supply for TCXO (pin 1).
5	VDD_MCU	PWR	Power supply from MCU.
6	VDD_RF	PWR	Power supply for RF section.
7	GND	GND	Ground.
8	DBG_SX1276_DIO2	—	No Connect. Internal Use
9	DBG_SX1276_DIO3	—	No Connect. Internal Use
10	DBG_SX1276_DIO4	—	No Connect. Internal Use
11	DBG_SX1276_DIO5	—	No Connect. Internal Use
12	DBG_SX1276_DIO1	—	No Connect. Internal Use
13	DBG_SX1276_DIO0	—	No Connect. Internal Use
14	PB15/SPI2_MOSI	I/O	No Connect
15	PB14/SPI2_MISO	I/O	No Connect
16	PB13/SPI2_SCK	I/O	No Connect
17	PB12/SPI2_NSS	I/O	No Connect
18	PA10/USART1_RX	I	UART RX (Serial port: host-to-modem)

19	PA9/USART1_TX	O	UART TX (Serial port: modem-to-host)
20	PA8/MCO	O	BUSY signaling line.
21	PA5/ADC5/DAC2	O	Diagnostics LED (optional). Active high, connect to LED with appropriate in-series resistor
22	PA4/ADC4/DAC1	I/O	No Connect
23	PA3/ADC3	I	Internal Use. Debug UART_RX
24	PA2/ADC2	O	Internal Use. Debug UART_TX
25	GND	GND	Ground.
26	ANT	I/O	Transmit/Receive antenna port
27	GND	GND	Ground.
28	DBG_CRF1	—	No Connect. Internal Use
29	DBG_CRF3	—	No Connect. Internal Use
30	DBG_CRF2	—	No Connect. Internal Use
31	STSAFE_nRST	I	No Connect. Internal Use
32	VREF+	PWR	Reference voltage for the MCU ADC section. Must be externally connected to VDD
33	PA0/WKUP1	I/O	No Connect
34	MCU_nRST	I	MCU reset pin. Has internal pull-up and may be left floating.
35	PB8/I2C1_SCL	I	COMMAND signaling line.
36	PB9/I2C1_SDA	O	EVENT signaling line.
37	PB2/LPTIM1_OUT	O	Host MCU reset line. Active low, Z (high impedance) when inactive.
38	PB7/LPTIM1_IN2	I/O	No Connect
39	PB6/LPTIM1_ETR	I/O	No Connect
40	PB5/LPTIM1_IN1	I/O	No Connect
41	PA13/SWDIO	I/O	Internal Use. Programming SWDIO
42	PA14/SWCLK	I/O	Internal Use. Programming SWCLK
43	BOOT0	I	No Connect. Internal Use
44	GND	GND	Ground
45	PH1-OSC_OUT	I/O	No Connect
46	PH0-OSC_IN	I/O	No Connect
47	TCXO_OUT	O	TCXO output.
48	VDD _TCXO	PWR	TCXO power supply input. Externally connected to pin 1.
49-57	GND	GND	Ground

Note: GPIO pins are to be left OPEN if not used.

3 Modem Interface

Serial port

- USART1

- Transmit (PA9/USART1_TX - pin 19)
- Receive (PA10/USART1_RX - pin 18)

The following serial port parameters are used.

Table 2 Serial Port Parameters

Baud Rate	115,200 bps
Data Bits	8
Stop Bits	1
Parity	None

3.1 GPIO Signalling Lines

3.1.1 COMMAND line

- Active-low, GPIO input line
- This line is used to signal to the Modem that the Application MCU is about to transmit a command. After driving it low the host has to wait until the BUSY line goes low or at least 10 ms before sending the first character to allow the Modem to wake up and start reception. After the command is sent, the line must be driven high, at which point the Modem will process the command. If a valid command was received, the Modem will transmit its response within 200ms.

3.1.2 BUSY line

- Active-high, GPIO output line
- This line signals if the Modem is busy or ready to receive commands. It is high while the Modem is busy and will go low as soon as the Modem is ready to receive a command. The Modem should be ready to receive after a maximum of 10 ms after the COMMAND line has been asserted. The BUSY line will go high again after the command has been received and the COMMAND line has been released.

3.1.3 EVENT line

- Active-high, GPIO output line
- This line signals to the Application MCU that the Modem has asynchronous event data pending. The Application MCU can use the GetEvent command to retrieve such data.

3.1.4 MCURESET line

- Active-low, GPIO output and floating line
- This line signals to the Application MCU and intend for the Modem to reset the Application MCU by pull LOW on this line for 10ms.

4 Message Flow

Messages are exchanged via USART1 using COMMAND, BUSY and EVENT GPIO lines for synchronization. This section describes the timing for message flow between Application MCU and the Modem.

The COMMAND line is set by Application MCU, and the EVENT, the BUSY line is set by the Modem.

All serial communication consists strictly of Application MCU-initiated command-response sequences.

4.1 Message Flow for Command and Response

Flow for sending command and reading synchronous response:

1. Sending commands are initiated via operating COMMAND line: the Application MCU pulls the COMMAND line low.
2. The Modem drives the BUSY line low.
3. The Application MCU sends the command using the USART1 RX line.
4. Application MCU must drive the command line high after the last character of the command has been transmitted. The Modem will only start interpreting the command after the COMMAND line is high.
5. The Modem drives the BUSY line high.
6. If the checksum has been validated the Modem will process the command and will send a matching response within 200ms, otherwise no response will be sent. The spacing between characters of the response does not exceed 5ms.

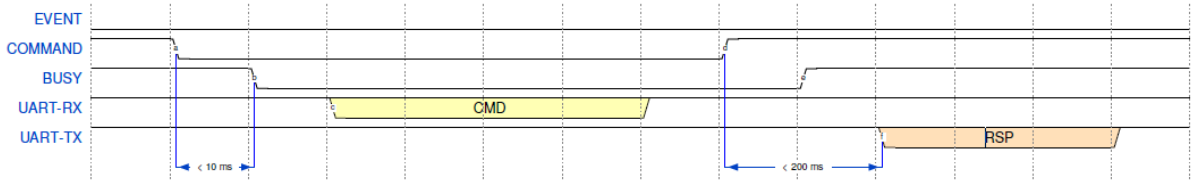

Figure 2 Timing for synchronous command/response pair

Please note that the Application MCU must not send a new command before the response has been fully received.

Please note that the Modem will not send any unsolicited response to the serial port. If event data becomes available asynchronously, the Modem will signal this using the EVENT line.

4.2 Message Flow for Asynchronously Response

There can be an asynchronous event generated from the Modem.

Flow for receiving asynchronous response:

1. Whenever data becomes available asynchronously, the Modem sets the EVENT line high.
2. The Application MCU should issue a GETEVENT command to retrieve the response data.
3. After all pending responses have been retrieved, the EVENT line will be set low again.

Please note that the timing requirement in section 4.1 must be fulfilled in the GETEVENT comm and/response pair.

5 Message Format

This section describes the packet format running on the USART1 interface, for both command packets and response packets.

The maximum packet size is 258 bytes.

5.1 Command Packets

All command messages consist of a command code that defines the operation to be performed, a payload length and optional payload, and a trailing checksum character which is the XOR of all previous bytes.

Table 3 Command message structure

Field	Size (bytes)	Description
cmd	1	Command code (see Table 6)
len	1	Length of the payload field
payload	0-255	Parameters for the command as binary data
chk	1	Packet checksum (xor over cmd, len, payload)

The Modem begins analyzing the command message after the COMMAND line is released. After the packet structure is successfully verified (correct number of bytes and correct check character), the Modem will interpret the command and in any case, will send a response. If the packet structure is invalid, the Modem ignores the packet and will send a response indicating a Frame Error.

5.2 Response Packets

The response format is similar to the command format but instead of the command code, it holds a return code that indicates the successful completion of the command or an error condition.

Table 4 Response message structure

Field	Size (bytes)	Description
rc	1	Return code of the command
len	1	Length of the payload field
payload	0-255	Response data of the command as binary data
chk	1	Packet checksum (xor over rc, len, payload)

The return code byte is defined as follows.

Table 5 List of return codes

Code	Name	Description
0x00	Ok	command executed without errors
0x01	Unknown	command code unknown
0x02	NotImpl	command not yet implemented
0x03	NotInit	command not initialized
0x04	Invalid	command parameters invalid
0x05	Busy	command cannot be executed now
0x06	Fail	command execution failed
0x07	BadFmt	format check failed
0x08	BadCrc	crc check failed
0x09	BadSig	signature verification failed
0x0A	BadSize	size check failed
0x0F	FrameError	Serial port framing error

6 Commands and Responses

6.1 Overview of commands and responses

The serial communication with the Modem is performed using strictly alternating command and response messages. Response messages convey a return code and the result of the operation triggered by the preceding command message.

Table 6 List of commands and responses

Name	Code	Description	Input	Output
GetEvent	0x00	Retrieve pending events		type[1], count[1], eventdata[n]
GetVersion	0x01	Get version		bootversion[4], firmware version[4], lorawan[2]
Reset (Async. Command)	0x02	Reset Modem		
FactoryReset (Async. Command)	0x03	Perform Modem factory reset		
ResetCharge	0x04	Reset charge counter		
GetCharge	0x05	Get accumulated charge offset		Charge[4]
GetTxPowerOffset	0x06	Get power correction offset		offset[1]
SetTxPowerOffset	0x07	Set TX power correction offset	Offset[1]	
Test	0x08	Radio Test function	(see description)	(see description)
FirmwareUpdate	0x09	Write firmware update	part[4+128]	

GetTime	0x0A	GetGPS wall time		timestamp[4]
GetStatus	0x0B	Get Modem status		status[1]
SetAlarmTimer (Async. Command)	0x0C	Set alarm / wakeup timer	seconds[4]	
GetTrace	0x0D	Get diagnostic crash log		backtrace log[n]
GetPin	0x0E	Get device registration PIN		PIN[4]
GetChipEui	0x0F	Get device chip EUI		ChipEUI[8]
GetJoinEui	0x10	Get join EUI		JoinEUI[8]
SetJoinEui	0x11	Set join EUI and derive keys	JoinEUI[8]	
GetDevEui	0x12	Get device EUI		DeviceEUI[8]
SetDevEui	0x13	Set device EUI and derive keys	DeviceEUI[8]	
SetNwkKey	0x14	Set network key	NwkKey[16]	
GetClass	0x15	Get the LoRaWAN device Class		class[1]
SetClass	0x16	Set LoRaWan class A/C	class[1]	
SetMulticast	0x17	Set multicast session parameters	param[40]	
GetRegion	0x18	Get regulatory region		region[1]
SetRegion	0x19	Set regulatory region	region[1]	
ListRegions	0x1A	List the supported regulatory regions		region[1-5]
GetAdrProfile	0x1B	Get ADR profile		type[1]
SetAdrProfile	0x1C	Set ADR profile and optional parameters	type[1]+list[16]*	
GetDmPort	0x1D	Get DM port		port[1]
SetDmPort	0x1E	Set DM port	port[1]	
GetDmInfoInterval	0x1F	Get DM reporting interval		interval[1]
SetDmInfoInterval	0x20	Set DM reporting interval	interval[1]	
GetDmInfoFields	0x21	Get default info field for DM status		inflist[n]
SetDmInfoFields	0x22	Set default info field for DM status	inflist[n]	
SendDmStatus	0x23	Send DM status now	inflist[n]	
SetAppStatus	0x24	Set application-specific status for DM	appstatus[8]	
Join (Async. Command)	0x25	Start joining the network		
LeaveNetwork	0x26	Leave the network		
SuspendModemComm	0x27	Suspend/resume radio operations	suspend[1]	

GetNextTxMaxPayload	0x28	Get max payload size for next TX		size[1]
RequestTx (Async. Command)	0x29	Transmit frame unconfirmed or confirmed	port[1], conf[1], data[n]	
EmergencyTx (Async. Command)	0x2A	Transmit frame immediately (smoke alarm)	port[1], conf[1], data[n]	
UploadInit	0x2B	Set file upload port, encryption mode, size	p[1],en[1],sz[2]	
UploadData	0x2C	Write data for file upload transmission	data[n]	
UploadStart	0x2D	Verify data and start file upload	crc[4]	
StreamInit	0x2E	Set data stream parameters	port[1],mode[1]	
SendStreamData	0x2F	Send data stream record	port[1]+record[n]	
StreamStatus	0x30	Retrieve stream status	port[1]	pending[2]+free[2]

6.2 Command Details

All multi-byte integers contained in command or response payloads are transmitted least-significant-byte-first (LSBF).

6.2.1 EmergencyTx

This command sends the given data on the specified port as an unconfirmed or confirmed frame immediately. It has a higher priority than all other services and does not take duty cycle or payload size restrictions into account. It can be used to signal an alarm condition (like smoke alarm) in real-time, but it should be used with caution! A TxDone event is generated when the frame has been sent. The parameter of the TxDone event indicates if the frame was sent and acknowledged (0x02), or sent but not acknowledged (0x01).

Note: The application shall not use port 0 or the LoRaWAN test port 224 as well as the ports from 225 to 255 since they are reserved for future standardized application extensions.

Table 7 EmergencyTx

Field	Size (bytes)	Description
port	1	uplink port number
conf	1	acknowledge - 0=no ACK, 1=ACK
data	n	data

➔ See also RequestTx, Join, GetNextTxMaxPayload.

6.2.2 FactoryReset

This command performs a factory reset. In addition to the MCU reset all persistent settings are reset back to their factory state. The default setting is:

Table 8 Factory Reset

Config Setting	Default Value
DM port	200
reporting interval	1h
periodic DmInfo fields	status, charge, temp, signal, uptime, rxtime
ADR profile	0 (network-controlled)
regulatory region	0x01 (EU868)
board tx power offset	0
data streaming port	disabled

6.2.3 FirmwareUpdate

This command is used to repeatedly store parts of a firmware update to be installed. The firmware data has to be split into blocks of 128 bytes each and has to be provided in ascending order.

Table 9 Command Payload Format

Field	Size (bytes)	Description
blkno	2	Current block number (0 to blkcnt-1)
blkcnt	2	Total number of blocks
blkdata	128	Firmware section starting at byte blkno*128. If the firmware size is not a multiple of 128 then the last data block must be padded with zero bytes.

When the update has been fully stored (last part, blkno = blkcnt-1), the code signature and integrity will be validated and the update will be registered for installation. The update will be installed at the next reset of the modem.

6.2.4 GetAdrProfile

This command returns the ADR profile type.

➔ See also SetAdrProfile

6.2.5 GetCharge

This command returns the total charge counter of the Modem in mAh. This value includes the accumulated charge since the production of the Modem or since the last invocation of the ResetCharge command.

➔ See also ResetCharge

6.2.6 GetChipEui

This command returns the ChipEUI. The ChipEUI is also the default DeviceEUI. It is programmed during manufacturing and is immutable.

➔ See also GetDevEui, GetJoinEui

6.2.7 GetClass

This command gets the LoRaWAN device class.

Table 10 Get Class – Response Payload Format

Field	Size (bytes)	Description
class	1	LoRaWAN device class

➔ See also SetClass

6.2.8 GetDevEui

This command returns the DeviceEUI.

➔ See also SetDevEui, GetChipEui, GetJoinEui

6.2.9 GetDmInfoFields

This command lists the info fields to be included in the periodic DM status messages.

➔ See also SetDmInfoFields, Uplink Message Format

6.2.10 GetDmInfoInterval

This command returns the device management reporting interval. The interval is specified in seconds, minutes, hours or days.

➔ See also SetDmInfoInterval, Format of Reporting Interval

6.2.11 GetDmPort

This command gets the device management port.

➔ See also SetDmPort

6.2.12 GetEvent

This command can be used to retrieve pending events from the Modem. Pending events are indicated by the EVENT line. The EVENT line will be de-asserted after all events have been retrieved and no further events are available. When no event is available this command returns with empty response payload.

Table 11 Get Event – Response Payload Format

Field	Size (bytes)	Description
type	1	Event type, see Table 40
count	1	Number of missed events of this type in case of overrun
data	0-253	Event-specific data

Events that are not retrieved by the application might be overwritten by new event data of the same type. In this case, only the latest event data will be returned and the count field indicates how many events of this type have been missed.

➔ See also Events

6.2.13 GetJoinEui

This command returns the join EUI.

➔ See also SetJoinEui, GetDevEui, GetJoinEui

6.2.14 GetNextTxMaxPayload

This command returns the maximum application payload size possible according to the LoRaWAN regional parameters for the next transmission using the current data rate while assuming no FOpts are present and that a device is not behind a repeater.

➔ See also RequestTx

6.2.15 GetPin

This command returns the device registration PIN (4 bytes).

6.2.16 GetRegion

This command returns the regulatory region.

Table 12 Get Region

Region	Code
EU868	0x01
AS923	0x02
US915	0x03
AU915	0x04
CN470	0x05

➔ See also SetRegion, ListRegions

6.2.17 GetStatus

This command returns the Modem status which may indicate one or more notification conditions.

Table 13 Get Status

Bit	Value	Name	Description
0	0x01	Brownout	reset after brownout
1	0x02	Crash	reset after panic
2	0x04	Mute	device is muted
3	0x08	Joined	device has joined the network
4	0x10	Suspend	radio operations suspended (low power)
5	0x20	Upload	file upload in progress
6	0x40	Joining	Device is trying to join the network
7	0x80	Stream	Data stream in progress

6.2.18 GetTime

Query the current GPS time. The application layer clock synchronization protocol is used to link the device clock to GPS time. The returned time specifies the seconds since the GPS epoch (00:00:00, Sunday 6th of January 1980). If the device is not yet synchronized to GPS time then the returned value is zero. This may happen if the server has not yet answered time sync requests. The accuracy of the synchronization is in the range of seconds and depends on latencies in the network infrastructure.

Table 14 Get Time – Response Payload Format

Field	Size (bytes)	Description
seconds	4	seconds since GPS epoch

6.2.19 GetTrace

This command returns the latest crashlog saved by the current firmware. The format of the crashlog data is proprietary and will not be disclosed

6.2.20 GetTxPowerOffset

This command gets the board-specific correction offset for transmission power to be used (signed integer in dB).

➔ See also SetTxPowerOffset

6.2.21 GetVersion

This command returns the version of the bootloader and the version of the installed firmware.

Table 15 Get Version – Response Payload Format

Field	Size (bytes)	Description
bootversion	4	boot loader version number.
fwversion	4	Modem firmware version number.
loRaWan	2	LoRaWan version number.

6.2.22 Join

This command starts joining or rejoining the network. During the join procedure, no further transmissions can occur. When the network has been successfully joined, a Joined event is generated. If the device is already joined to a network, or is in the process of joining, this command has no effect.

Once this command has been issued, the modem will attempt to join the network indefinitely while adhering to the join duty cycle mandated by the LoRaWAN specification. To abort an ongoing join attempt, use the LeaveNetwork command.

➔ See also LeaveNetwork, RequestTx

6.2.23 LeaveNetwork

This command leaves the network if already joined, or cancels an ongoing join process. After leaving the network, no further transmissions can occur.

➔ See also Join

6.2.24 ListRegions

This command returns the regulatory regions supported by the Modem.

➔ See also GetRegion, SetRegion

6.2.25 RequestTx

This command requests sending the given data on the specified port as an unconfirmed (0x00) or confirmed (0x01) frame.
Command protocol: port[1], conf[1], data[n]

Table 16 Request Tx

Field	Size (bytes)	Description
port	1	uplink port number
conf	1	acknowledge - 0=no ACK, 1=ACK
data	n	data

The request will be queued, and the frame will be sent as soon as the current bandwidth usage of the regulatory region permits. A TxDone event is generated regardless of whether the frame has been successfully sent or failed to be sent due to specified data exceeded the maximum payload size. The parameter of the TxDone event indicates if the frame was sent and acknowledged (0x02), sent but not acknowledged (0x01) or not sent (0x00). When application downlink data has been received in RX1 or RX2 a DownData event will be generated containing the port and data received. If a RequestTx command is issued before the TxDone event of a previous transmission request is generated the command will fail with Busy return code. If the command is issued before the network has been joined it will fail with NotInit return code.
Note: The application shall not use port 0 or the LoRaWAN test port 224 as well as the ports from 225 to 255 since they are reserved for future standardized application extensions.

➔ See also GetNextTxMaxPayload, EmergencyTx, Join

6.2.26 Reset

This command performs a reset of the Modem MCU. All transient state (including session information) will be lost and the Modem needs to join the network again.

6.2.27 ResetCharge

This command resets the accumulated charge counter to zero.

➔ See also GetCharge

6.2.28 SendDmStatus

This command sends the specified set of information fields in one or more DM status messages immediately. The set is specified as a list of field codes as defined in Uplink Message Format. Duplicate and invalid fields will be rejected (see note in Periodic Status Reporting).

➔ See also SetDmInfoFields, Uplink Message Format, Periodic Status Reporting

6.2.29 SendStreamData

The command payload is pushed as a data record into the FIFO of the streaming encoder. Data is drained from the FIFO with every transmission managed autonomously by the Modem. New data records can be added to the application at any time provided there is enough space in the FIFO (total FIFO size is 512 bytes). A data record must have a size SZ which satisfies 0 < SZ < 255. The record encoding requires two overhead bytes.

Table 17 Send Stream Data – Command Payload Format

Field	Size (bytes)	Description
port	1	port
record	SZ	record

Return codes:

Ok: Data added as a record to the FIFO
Busy: The FIFO has not enough space. No data has been added to the stream.
BadSize: The data size is illegal

➔ See also StreamInit, StreamStatus, GetDmPort

6.2.30 SetAdrProfile

This command sets the ADR profile and parameters.

Table 18 Set ADR Profile

Name	Code	Parameters
Network Server Controlled	0x00	
Mobile Long Range	0x01	
Mobile Low Power	0x02	
Custom	0x03	list of preferred data rates [16]

A custom ADR profile consists of a list of 16 preferred data rates. For every transmission, a random entry in that list is selected. This makes it possible to create distributions of preferred data rates. For example, the list 00 00 00 00 00 00 00 00 01 01 01 01 02 02 03 03 results in 50 % DR0, 25 % DR1, 12.5 % DR2, and 12.5 % DR 3. If the selected data rate is unavailable at the time of transmission, the closest available data rate is used instead.
The predefined profiles use the following distributions:

- Mobile Long Range: 50% MinDr, 25% MinDr + 1, 25% MinDr + 2
- Mobile Low Power: 25% MaxDr, 25% MaxDr -1, 25% MaxDr – 2, 25% MaxDr -3

For he US915 region, MaxDr = 4, and MinDr = 0; for the EU868 region, MaxDr = 7, and MinDr = 0.
AS with custom profiles, if the selectd data rate is unavailable at the time of transmission, the closest available data rate is used instead.

➔ See also GetAdrProfile

6.2.31 SetAlarmTimer

This command sets an application alarm timer (in seconds). When the timer has expired an Alarm event is generated.

6.2.32 SetAppStatus

This commands sets application-specific status information to be reported to the DM service. This information is an application-defined, arbitrary 8-byte data blob. Once set, it is included in the appstatus info field sent as part of the periodic status reports to the DM service. On the cloud side, this information can then be retrieved from the service.
Note that this command does not trigger an immediate status report (⟶ SetDmInfoInterval). If the application desires to send the status immediately, it can issue the SendDmStatus command with the appstatus tag.
The application status is not stored persistently, i.e. after reset, no application status will be reported.

➔ See also SendDmStatus, Uplink Message Format, SetDmInfoInterval

6.2.33 SetClass

This command sets the LoRaWAN device class. Currently, only class A and class C are supported.

If the command is successful, a change from class A to class C is effective after a completed TX transaction. The network server should also be informed about the class change, typically on a separate channel for LoRaWAN 1.0.3. For a change from class C to class A, the RX remains enabled until the next TX transaction. Note that the class settings are not persistent.

Table 19 Set Class – Command Payload Format

Field	Size (bytes)	Description
class	1	Class value

Value	LoRaWAN Class	Description
0x00	A	Open short RX windows after TX
0x01	C	RX remains on

➔ See also GetClass

6.2.34 SetDevEui

This command sets the DeviceEUI and triggers a new key derivation and automatically sets a new AppKey. The device EUI is set to D0-D1-D2-D3-D4-D5-D6-D7 where Di represents the contents of the respective payload byte.

Table 20 Set Dev EUI

Command Payload Format								
Byte Length	1	1	1	1	1	1	1	1
Field	D0	D1	D2	D3	D4	D5	D6	D7

➔ See also GetDevEui, GetJoinEui, SetJoinEui

6.2.35 SetDmInfoFields

This command is used to specify the set of info-fields to be reported in the periodic DM status messages. The set is specified as a list of field codes as defined in Uplink Message Format. Duplicate and invalid fields will be rejected (see note in Periodic Status Reporting). An empty set is valid and will effectively disable the DM status message.

➔ See also SetDmPort, Uplink Message Format, Periodic reporting

6.2.36 SetDmInfoInterval

This command sets the device management reporting interval. The interval is specified in seconds, minutes, hours or days.

Any value of 0 (seconds, minutes, hours or days) disables device management reporting.

➔ See Also GetDmInfoInterval, Format of Reporting Interval

6.2.37 SetDmPort

This command sets the device management port.

➔ See also GetDmPort, SetDmInfoFields

6.2.38 SetJoinEui

This command sets the join EUI and triggers a new key derivation and automatically sets a new AppKey. The device join EUI is set to J0-J1-J2-J3-J4-J5-J6-J7 where Ji represents the contents of the respective payload byte.

Table 21 Set Join EUI

Command Payload Format								
Byte Length	1	1	1	1	1	1	1	1
Field	J0	J1	J2	J3	J4	J5	J6	J7

➔ See also GetJoinEui, GetDevEui, SetDevEui

6.2.39 SetMulticast

This command configures the LoRaWAN a downlink multicast session.

For details, refer to the LoRaWAN Standard Specification and the LoRaWAN Remote Multicast Setup Specification. Multicast groups are not persistent and are always kept until the next reset. A maximum of 2 multicast groups is allowed. If the group address is already configured it will be kept and the session parameters will be overwritten.

Table 22 Set Multicast – Command Payload Format

Field	Size (bytes)	Description
grpaddr	4	Multicast group address
nwkkeydn	16	Multicast group network key to verify the MIC of downlink traffic
appkey	16	Multicast group application key to decrypt the application payload
seqnoadn	4	The initial sequence counter for the application downlinks

6.2.40 SetNwkKey

This command sets the LoRaWAN 1.0.3 DevKey or LoRaWAN 1.1 NwkKey.

Note that a factory reset will erase this information. The device is required to rejoin the network.

➔ See also SetDevEui, SetJoinEui

6.2.41 SetRegion

This command sets the regulatory region. Note that not all regions listed in this document may be supported. Please refer to your product datasheet or use the ListRegions command to enumerate the available regions. Additionally, this command resets the ADR profile to Network Server Controlled. If a different ADR profile is desired, the profile needs to be set again.

➔ See also GetRegion, ListRegions, SetAdrProfile

6.2.42 SetTxPowerOffset

This command sets the board-specific correction offset for transmission power to be used. The offset depends on the board design and antenna matching and is expressed in dB (signed integer).

6.2.43 StreamInit

The data stream encoder is initialized with specific parameters. This command must be called once before the first call to SendStreamData. If not called before the first call to SendStreamData then a default initialization takes place with port=0 and mode=0.

Table 23 Stream Init

Command Payload Format		
Byte Length	1	1
Field	port	mode

port: The port the streaming protocol operates on. If this value is zero then the streaming protocol is multiplexed on the DM port. No extra port needs to allocated in this case. Note: The application shall not use port 0 or the LoRaWAN test port 224 as well as the ports from 225 to 255 since they are reserved for future standardized application extensions.
mode: This byte is optional and defaults to zero. If this is zero then data records are not encrypted. If it has value 1 then data records will be subject to extra encryption to preserve privacy towards the stream decoding engine.
The stream decoding engine is not operating on clear text user data if this extra encryption layer is enabled. The application has to decrypt the data records returned by the decoding engine.

Return codes:

Busy: The stream is already initialized
Ok: Success

➔ See also SendStreamData, StreamStatus, GetDmPort

6.2.44 StreamStatus

Query the status of the streaming FIFO on the specified port. It returns two unsigned 16-bit integer values indicating the number of bytes pending transmission and the number of bytes still free in the FIFO.

Table 24 Stream Status

Response Payload Format		
Byte Length	2	2
Field	pending	free

Return codes:

Ok Return 4 bytes of information
NotInit Stream is not initialized

➔ See also StreamInit, SendStreamData

6.2.45 SuspendModemComm

This command temporarily suspends or resumes the Modem's radio operations. It can be used to prevent extra power consumption by the Modem in case the Application MCU temporarily needs more power itself and wants to prevent exceeding limits. Operations are suspended with parameter value 0x01 and resumed with parameter value 0x00.

6.2.46 Test

This command is used to implement test functionality for regulatory conformance, certification, and functional testing. With the exception of the TST_START command, test commands are only available if test mode is active. Test mode can only be activated if the Modem has not yet received a command that results in radio operation. Once test mode is active, all other Modem commands are disabled.

Table 25 Encoding for SF, BW, and CR

Value	0	1	2	3	4	5	6
SF	FSK	SF7	SF8	SF9	SF10	SF11	SF12
BW	125kHz	250kHz	500kHz	-	-	-	-
CR	4/5	4/6	4/7	4/8	-	-	-

6.2.47 TST_START

Start test mode. This command enables all other test functions.

Table 26 Tst Start

Command Payload Format		
Byte Length	1	8
Field	0x00	"TESTTEST"

6.2.48 TST_NOP

No operation. This command may be used to terminate an ongoing continuous TX operation.

Table 27 Tst Nop

Command Payload Format	
Byte Length	1
Field	0x01

6.2.49 TST_TX_SINGLE

Transmit a single packet.

Table 28 TST_TX_Single

Command Payload Format							
Byte Length	1	4	1	1	1	1	1
Field	0x02	frequency	tx_power	sf	bw	cr	payload_length

6.2.50 TST_TX_CONT

Continuously transmit packets as fast as possible.

Table 29 TST_TX_CONT

Command Payload Format							
Byte Length	1	4	1	1	1	1	1
Field	0x03	frequency	tx_power	sf	bw	cr	payload_length

6.2.51 TST_TX_HOP

Continuously transmit packets as fast as possible while respecting regional regulatory constraints (channel hopping, output power, dwell time, duty cycle).

Table 30 TST_TX_HOP

Command Payload Format					
Byte Length	1	1	1	1	1
Field	0x04	region	dr	tx_power	payload_length

6.2.52 TST_TX_CW

Transmit a continuous wave.

Table 31 TST_TX_CW

Command Payload Format			
Byte Length	1	4	1
Field	0x06	frequency	tx_power

6.2.53 TST_RX_CONT

Continuously receive packets.

Table 32 TST_RX_CONT

Command Payload Format					
Byte Length	1	4	1	1	1
Field	0x07	frequency	sf	bw	cr

6.2.54 TST_RSSI

Measure RSSI.

Table 33 TST_RSSI

Command Payload Format				
Byte Length	1	4	2	1
Field	0x08	frequency	time_ms	dr

The supplied datarate determines the bandwidth.

Response Payload Format	
Byte Length	1
Field	RSSI+64

6.2.55 TST_RADIO_RST

Reset the SX1276 radio.

Table 34 TST_RADIO_RST

Command Payload Format	
Byte Length	1
Field	0x09

6.2.56 TST_SPI

Send and receive SPI commands directly to the SX1276 radio.

Table 35 TST_SPI

Command Payload Format			
Byte Length	1	Var.	1
Field	0x0A	spi_command	resp_len

Response Payload Format	
Byte Length	resp_len
Field	spi_response

6.2.57 TST_EXIT

Exit test mode and reset Modem.

Table 36 TST_EXIT

Command Payload Format	
Byte Length	1
Field	0x0B

6.2.58 TST_BUSY_LOOP

It causes the Modem to enter an endless loop with interrupts enabled. Eventually, the software watchdog will save a crashlog, flash the diagnostics LED (if available) and reset the device. Note that this command will render the Modem completely unresponsive until it is reset.

Table 37 TST_BUSY_LOOP

Command Payload Format	
Byte Length	1
Field	0x0C

6.2.59 TST_PANIC

It causes an unrecoverable fault condition (panic). The Modem will immediately save a crashlog and halt, the diagnostics LEDs (if available) will flash for some time, and then the Modem will reset. Note that this command will render the Modem completely unresponsive until it is reset.

Table 38 TST_PANIC

Command Payload Format	
Byte Length	1
Field	0x0D

6.2.60 TST_WATCHDOG

It causes the Modem to enter an endless loop with interrupts disabled. Eventually, the hardware watchdog will reset the device. No crashlog will be written, diagnostic LED will not be flashing. Note that this command will render the Modem completely unresponsive until it is reset.

Table 39 TST_WATCHDOG

Command Payload Format	
Byte Length	1
Field	0x0E

6.2.61 UploadData

This command can be used to repeatedly set file data to be uploaded. The file data needs to be split into parts of a maximum of 255 bytes each and the submitted parts will be appended to an internal buffer. In total exactly as many bytes as specified by the UploadInit command have to be provided. The maximum size of file data is limited to 8K bytes.

➔ See also UploadData, GetDmPort, GetStatus

6.2.62 UploadInit

This command prepares a fragmented file upload. It specifies the port for the subsequent upload, optional encryption mode, and the file size. The port is included as metadata in the file stream and the stream is sent on the device management port so it can be processed by the Semtech Cloud Service. When encryption is used (mode value 0x01), the file data is encrypted using a 128-bit AES key derived from the AppSKey before it is further processed by the upload service. The mode value 0x00 disables the file encryption. For the derivation of the decryption key see Format of Defragmented upload Application File Data. Using encryption, full privacy can be ensured even if the file data is reassembled by the Semtech DM service. In this case, Semtech has no knowledge of the contents of the file data being uploaded! This command can also be used to cancel an ongoing file upload by specifying a file size of zero for the port in use.

➔ See also UploadData, GetDmPort

6.2.63 UploadStart

After all data bytes indicated to UploadInit have been provided using UploadData this command can be issued to actually start the transmission stream. The CRC parameter must match the CRC of the supplied data, otherwise, the command is rejected with BadCrc return code. If encryption was requested with the UploadInit command the data will be encrypted before the stream is started (see Format of Defragmented upload Application File Data). When the stream is started redundant fragments will be continuously sent in the background until a confirmation is received from the server that it has fully decoded the file or until twice the required number of chunks has been sent and no confirmation is received (see Format of upload Application File Data Fragments). When the upload stream is stopped a UploadDone event is generated which carries a status byte which indicates success or timeout. Running file upload is indicated by the Upload flag in the Modem status.

➔ See also UploadInit, UploadData

7 Events

Events are notifications that occur asynchronously to the command-response flow. Pending events are indicated via the EVENT line and can be retrieved via the GetEvent command. Multiple events for different event types might be queued and can be retrieved subsequently. However multiple events for the same event type will overwrite the previous event of that type. For example, if multiple downlinks are received, and a new downlink arrives before the DownData event of the previous one has been retrieved, only the new DownData event can be retrieved. Therefore it is good practice to retrieve pending events as soon as the EVENT line is set. Eventually missed events are indicated in the count field of the response of the GetEvent command.

7.1 Event Overview

Table 40 List of event codes

Name	Code	Description	Data

Reset	0x00	Modem has been reset	reset count[2]
Alarm	0x01	Alarm timer expired	
Joined	0x02	Network successfully joined	
TxDone	0x03	Frame transmitted	status[1]
DownData	0x04	Downlink data received	port[1], downdata[n]
UploadDone	0x05	Fileupload completed	status[1]
SetConf	0x06	Config has been changed by DM	info tag[1]
Mute	0x07	Modem has been muted or unmuted by DM	mute[1]
StreamDone	0x08	Data stream fragments sent	
LinkStatus	0x09	Network connectivity status changed	Status[1]
JoinFail	0x0A	Attempt to join network failed	

7.2 Event Details

7.2.1 Reset

This event indicates that the Modem has been reset.

It returns the current reset counter which is incremented at every reset. A reset can be caused by the UART command, DM request, power failure, or an internal error condition. The Application MCU might use this indication to eventually restore some state of the Modem (e.g. re-join).

7.2.2 Alarm

This event indicates that the programmed alarm timer has expired.

7.2.3 Joined

This event indicates that the Modem has successfully joined the network.

7.2.4 TxDone

This event indicates that the previously initiated RequestTx operation has completed.

It returns a status byte which indicates whether the frame was sent (value 0x01), or if the frame could not be sent because its length exceeded the maximum payload size for the current data rate (value 0x00).

7.2.5 DownData

This event indicates that a downlink with application data has been received.

It returns the RSSI, SNR and RX flags of that downlink, followed by the port and the payload of that time. RSSI is a signal value in dBm + 64, SNR is a signal value in 0.25 dB steps (see signal field of DM status as described in Uplink Message Format).

Table 41 Down Data

Format of Down event payload					
Byte Length	1	1	1	1	0-242
Field	RSSI	SNR	flags	port	Frame data

The RX flags are encodes as follow:

Flag	Code	Description
ACK	0x80	confirm UP frame was acked
NAK	0x40	confirm UP frame was not acked
DNW1	0x01	received in 1st DN slot
DNW2	0x02	received in 2nd DN slot
PING	0x04	Received in a scheduled RX slot (class B)

7.2.6 UploadDone

This event indicates the end of a previously initiated file upload.

It returns a status byte which indicates whether the file has been successfully received by the server (value 0x01) or if the upload stream has been aborted because no confirmation has been received from the server (value 0x00).

7.2.7 SetConf

This event indicates that a configuration setting has been changed by the DM service.

It returns one byte holding the tag code of the information field changed. Tags codes are described in section Uplink Message Format.

7.2.8 Mute

This event indicates that the Modem has been muted or unmuted by the DM service.

It returns one byte mute status (muted = 0x01, unmuted = 0x00).

7.2.9 StreamDone

This event indicates that the Modem has sent all fragments of the previously submitted data record and is now ready to accept new data records.

7.2.10 LinkStatus:

This event occurs if network-managed ADR is active and network connectivity status changes. This happens either when the connection has been lost after reaching the end of the ADR back-off sequence, i.e. The lowest datarate with all default channels enabled (status byte 0x00), or if a downlink is received after the connection has been previously lost (status byte 0x01). These are purely informational events; the device will continue to function normally, as offline conditions may be transient and the device could move back into coverage.

8 Command Sequence Examples

Note: The Application MCU (also referred to as Host).

8.1 Join

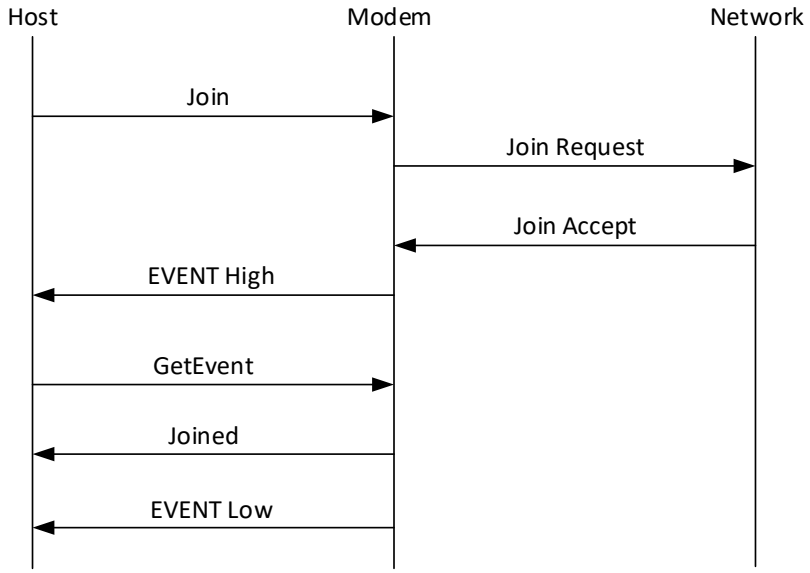

Figure 3 Command sequence for joining the network

8.2 Class A Uplink

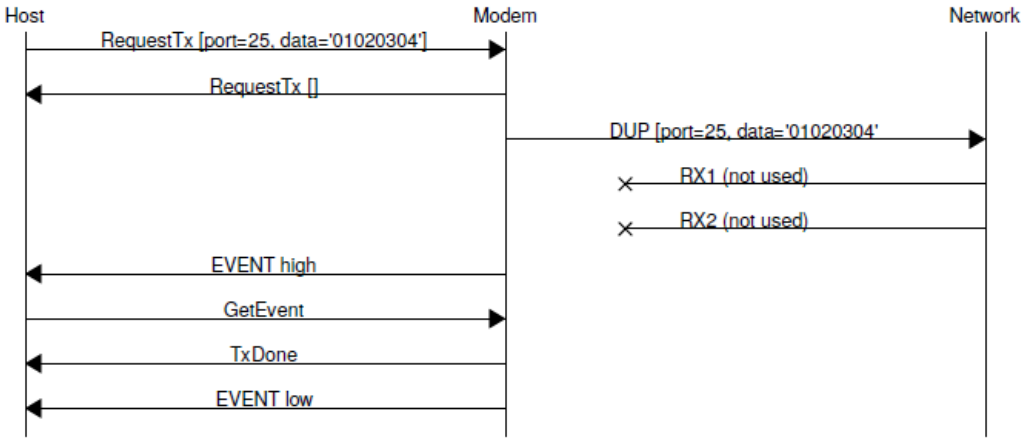

Figure 4 Command sequence for uplink without downlink

8.3 Class A Uplink with Downlink

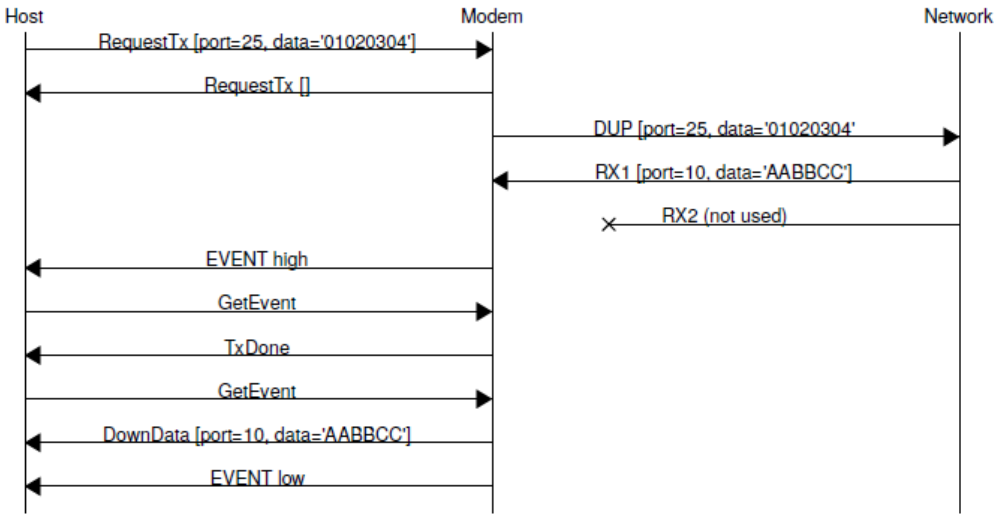

Figure 5 Command sequence for uplink with downlink

9 Modem Service

The Semtech Device Management Cloud Service has an HTTP-based API and is fed with the uplink service messages sent by the Modem. In return to each call, it can provide downlink messages to be delivered back to the Modem. It also can return defragmented application data records and defragmented application file data.

All uplink messages received on the device management port (default 200) should be forwarded to the cloud service.

9.1 Uplink Message Format

Uplink messages contain one or more concatenated device information fields. All fields begin with a one-byte field tag code and are followed by the field information. The content length of most fields is implicit and is defined in the table below. Fields specified as variable-length usually have larger content and are sent in separate frames. These fields are not mixed with other fields and the field content length is the full payload length. When due to duty cycle limitations, not all requested fields fit into one message, the fields are split over multiple messages.

All multi-byte integers contained in message payloads are transmitted least-significant-byte-first (LSBF).

Table 42 List of Modem Service Codes

Name	Code	Size (bytes)	Description
status	0x00	1	Modem status
charge	0x01	2	charge counter [mAh]
voltage	0x02	1	supply voltage [1/50 V]
temp	0x03	1	junction temperature [deg Celsius]
signal	0x04	2	RSSI [dBm]+64, SNR [0.25 dB]
uptime	0x05	2	duration since last reset [h]
rxtime	0x06	2	duration since last downlink [h]
firmware	0x07	4+4	firmware CRC and fuota progress
adrmode	0x08	1	ADR profile (0-3)
joineui	0x09	8	JoinEUI
interval	0x0A	1	reporting interval [values 0-63, units s/m/h/d]
region	0x0B	1	regulatory region
crashlog	0x0D	variable	crash log data
upload	0x0E	variable	application file stream fragments
rstcount	0x0F	2	Modem reset count
deveui	0x10	8	DevEUI
owncnt	0x11	2	owner number
session	0x12	2	session id / join nonce
chipeui	0x13	8	ChipEUI
stream	0x14	variable	data stream fragments
streampar	0x15	2	data stream parameters
appstatus	0x16	8	application-specific status
alcsync	0x17	variable	application layer clock sync data

9.1.1 Periodic Status Reporting

The Modem periodically sends a status message, which by default contains the status, charge, voltage, temp, signal, uptime, and rxtime fields. The default set of fields can be changed by the SetDmInfo Modem command or by the SetDmInfo DM request. The first status message after joining additionally contains the rstcount and session fields. Other fields might be explicitly requested using the GetInfo downlink message. The reporting interval can be queried and set using the GetInterval and SetInterval Modem commands. Additionally, it can be set by the DM via the SetConf downlink message specifying a value for the interval field.

Note: The upload, stream and alcsync fields are not part of periodic status messages and are sent as separate messages by the respective protocols. Therefore these fields cannot be requested by the SendDmStatus, SetDmInfoFields commands and by the GetInfo downlink request.

9.1.2 Format of Reporting Interval

The periodic status reporting interval field is encoded in one byte where the two top-most bits specify the unit (sec/min/hour/day), and the lower six bits the value 0-63. A value of zero disables the periodic status reporting.

Table 43 Format of the reporting interval

Bits	Variable	Description
7-6	Unit	00-sec, 01-day, 10- hour, 11- min
5-0	Value	0-63

9.1.3 Format of Upload Application File Data Fragments

upload fields are of variable length and are sent in a separate message containing only this field.

Each upload field begins with a 16-bit header (LSBF) consisting of 2-bit session id, 4-bit session counter, and 10-bit chunk count-1.

This header is followed by one or more 8-byte chunks of spread file entropy data.

9.1.4 Format of Defragmented Upload Application File Data

When the cloud service has received enough fragments to reconstruct the complete file, it will verify the integrity and will return the file to the application together with a downlink message to indicate stop the streaming of fragments.

This downlink message needs to be delivered back to the device. The file consists of a 12-byte header followed by the file data, which is optionally encrypted.

Table 44 Format of Defragmented upload Application File Data

Header	Size (bytes)	Description
port	1	application port
flags	1	file data is encrypted if non-zero
size	2	size of file data
hash1	4	truncated SHA256 hash over received file data
hash2	4	outer or inner hash over received file data (*)
data	size	file data

If no encryption is used, hash2 is the next 32-bit of the SHA256 hash over the received file data. When the file data is encrypted (flags non-zero), hash2 is the truncated SHA256 hash of the plain file data. The decryption of the file data can be performed using the AppSKey for AES in counter mode with the following initial block:

Table 45 AES-CTR initial block

AES-CTR initial block						
Byte Length	1	4	1	4	4	2
Field:	0x49	0x00	0x40	file size (lsbf)	hash2	0x00

After decryption, hash2 can be used to verify the integrity of the plain file data.

9.1.5 Format of stream Application Data Record Fragments

stream fields are of variable length and are sent in a separate message containing only this field.

Each stream field begins with one byte consisting of 1-bit type (redundant/systematic) and 7-bit fragment counter. This header is followed by one or more data record fragments.

9.1.6 Format of Defragmented stream Application Data Records

When the cloud service has received enough frames to reconstruct one or more application data records it will return these records retaining the order of transmission. I.e. when a missing record is reconstructed from subsequent redundancy frames, the original order of the returned records is retained.

Since the returned data records are optionally encrypted, the application might need to decrypt them. The decryption of the record data can be performed using the AppSKey for AES in counter mode with the following initial block.

Table 46 AES-CTR initial block

AES-CTR initial block						
Byte Length	1	4	1	4	4	2
Field:	0x49	0x00	0x41	record size (lsbf)	application record counter (lsbf)	0x00

9.2 Downlink Message Format

The cloud service might return one or more request messages which have to be delivered over the network back to the Modem on the device management port (default 200). All downlink messages have the following format:

Table 47 Downlink message format for device management port

Field	Size (bytes)	Description
upcount	1	uplink count
updelay	1	uplink delay [s]
request	1	request code
payload	variable	request data

Next to the request code and data, each message contains an upcount field which indicates the number of uplinks to generate. These uplinks can be used to create additional downlink opportunities and should be generated at the rate specified by the updelay field. The reception of a new request message resets the uplink generation.

9.2.1 Overview of Downlink Messages

The following downlink requests are defined:

Table 48 Downlink request codes for device management port.

Request	Code	Parameters	Description
Reset	0x00	mode[1]+rstcnt[2]	reset Modem or Application MCU
Fuota	0x01	n	fuota firmware update chunk
FileDone	0x02	sessionidctr[1]	file upload complete
GetInfo	0x03	taglist[n]	get info fields
SetConf	0x04	tag[1]+value[n]	set config field
Rejoin	0x05	sesscnt[2]	rejoin network
Mute	0x06	mute[1]	permanently disable/enable Modem
SetDmInfo	0x07	taglist[n]	set list of default info fields
Stream	0x08	param[n]	set data stream parameters
ALCSync	0x09	n	application layer clock sync data

9.2.2 Downlink Request Details

9.2.2.1 Reset

This request performs a reset of the Modem or Application MCU (if the expected reset counter matches).

The first parameter indicates which units need to be reset (0x01 = Modem, 0x02 = app MCU, 0x03 = both). The second parameter is the expected reset count of the Modem. At startup, after reset, the Modem generates a Reset event.

9.2.2.2 Fuota

This request delivers a FUOTA firmware chunk to the Modem.

In return, the Modem will send an uplink with the firmware field indicating the progress of the update. When the Modem has received enough chunks to reconstruct the full update, it will automatically reset and install the update.

9.2.2.3 UploadDone

This request signals the successful reception of a file upload in progress.

It has the session id and session counter of the completed file upload as a parameter to identify the completed upload session. The parameter is one byte and is encoded as a 2-bit session id (high-nibble) and 4-bit session counter (low-nibble). The Modem should stop streaming file fragments on the reception of this request. An UploadDone event will be generated for the Application MCU.

9.2.2.4 GetInfo

This request specifies a list of information tags to be reported by the Modem as soon as possible.

9.2.2.5 SetConf

This request allows setting a specific configuration field.

The payload begins with the tag byte of the field to be set and is followed by the value. The length of the value is implicitly defined by the information field. A SetConf event will be generated for the Application MCU.

9.2.2.6 Rejoin

This request instructs the Modem to rejoin the network.

The parameter is the expected session counter. The Modem only rejoins the network if the counter matches, otherwise it will report the current value of the session counter.

9.2.2.7 Mute

When the parameter is non-zero this request will permanently mute the Modem.

It will reenable the Modem when the parameter is zero. Muting will prevent the Modem from transmitting any application messages. Therefore, Modem commands which would trigger transmissions will fail with Fail and Mute status. A Mute event will be generated for the Application MCU indicating the current mute state in the status byte. When the parameter is non-zero, its value specifies the number of days when the Modem will send a status message anyway (0xFF never).

9.2.2.8 SetDmInfo

This request specifies the list of information tags to be included in the periodic status messages.

By default, the status, charge, voltage, temp, signal, uptime and rxtime fields are reported.

9.2.2.9 StreamRate

This request sets the new coding rate for the data stream on the specified port.

9.2.2.10 ALCSync

This request provides time correction feedback for the application layer clock sync protocol.

10 Mbed Shield

For development purposes, the Modem is available on the SEMTECH Mbed Shield.

Figure 6 CMWX1ZZABZ-104 mbed shield

10.1 Power Supply

The CMWX1ZZABZ-104 mbed shield needs power with 3.3V via the following pins:

Table 49 Power Supply

Function	Shield Header/Pin
VDD	J3/4
GND	J3/6 or J2/7

10.2 Serial interface

The Serial Interface of the Modem shield is accessible through the following pins.

Table 50 Serial Interface

Modem Function	STM32 GPIO	Shield Header/Pin
UART RX	PA10	J2/1
UART TX	PA9	J1/3
COMMAND	PB8	J2/4
BUSY	PA8	J2/5
EVENT	PB9	J2/3
MCURESET	PB2	J2/2

10.3 Debug Interface

The Modem provides diagnostic output on pin 1 of the TEST header with 115200bps / 8N1 parameters. To view the output this pin needs to be connected to the RX line of a serial port or of a serial-to-USB converter and displayed with a terminal application.

10.4 JTAG/SWD Programming

The firmware of the Modem shield can be programmed via the 6-pin JTAG header using a JTAG debugger/programmer and a Tag-Connect™ adapter cable (TC2030-MCP-NL)[1].

[1] Tag-Connect is the registered trademark of Tag-Connect. For more information visit www.tag-connect.com

Table 51 JTAG/SWD Programming

Function	JTAG Pin	ST-LINK V2 Pin
VCC	1	1
SWDIO	2	7
nRESET	3	15
SWCLK	4	9
GND	5	20
-	6	-

11 Revision History

Date	Version	Log
10/28/2019	1.0	First final release
1/16/2020	1.1	Updated to match the changes in the modem milestone 17
1/22/2020	1.2	Updated Header and Footer
1/27/2020	1.3	Removed Watermark & corrected reference error in the footer

Important Notice

Information relating to this product and the application or design described herein is believed to be reliable, however such information is provided as a guide only and Semtech assumes no liability for any errors in this document, or for the application or design described herein. Semtech reserves the right to make changes to the product or this document at any time without notice. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. Semtech warrants performance of its products to the specifications applicable at the time of sale, and all sales are made in accordance with Semtech's standard terms and conditions of sale.

SEMTECH PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS, OR IN NUCLEAR APPLICATIONS IN WHICH THE FAILURE COULD BE REASONABLY EXPECTED TO RESULT IN PERSONAL INJURY, LOSS OF LIFE OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. INCLUSION OF SEMTECH PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE UNDERTAKEN SOLELY AT THE CUSTOMER'S OWN RISK. Should a customer purchase or use Semtech products for any such unauthorized application, the customer shall indemnify and hold Semtech and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs damages and attorney fees which could arise.

Contact Information

Semtech Corporation
Wireless & Sensing Products
200 Flynn Road, Camarillo, CA 93012
E-mail: sales@semtech.com
Phone: (805) 498-2111, Fax: (805) 498-3804
www.semtech.com