LoRa Protocol

RS26x Series

Application Note *v2.1*

1 Introduction

The RS26x family of LoRa sensors **[A]** implements the protocol described in this document for encoding uplink messages to LNS (LoRa Network Servers). Similarly, downlink messages sent via the LNS to the sensors must adhere to the protocol to ensure proper processing by the receiving device.

This document describes the structure of the protocol and the parameters accessible in the RS26x device via protocol messages.

2 Nomenclature

Device Parameters are displayed in ***italicized bold*** font.

Device Parameter values are displayed in ***'italicized bold'*** font within single quotation marks.

3 Overview

The RS26x LoRaWAN protocol encodes data in Tag-Value pairs. Each message data item is prefixed with a Tag to indicate the context of the successive data (*e.g.* the data type, the length in bytes, *etc.*). Details of the Tags and associated data are governed by the revision number of the protocol, termed the API Version.

Tables of the Tags and data associated with each API Version are presented in Appendix A: API Parameter Tables. The tables list all the details necessary for encoding and decoding the message types implemented by the protocol.

Each protocol message incorporates the identifier of the API Revision it is encoded for. RS26x devices will only process messages encoded for the API Revision implemented in the programmed firmware. **Messages received by the RS26x device encoded for earlier or later API Revisions will be rejected.**

Applications being used to decode uplink messages from the RS26x device must ensure the correct API Revision table is used to successfully decode the message. When messages are being encoded for downlink, the API Revision encoded in the message must match the API Revision table from where Tag related data is derived. The downlink message must also be encoded with the appropriate API Revision Parameter Table.

Unique Message Identifiers are used to indicate the function performed by the message. The Message Identifier has an implicit relationship with collections of Tags, with these collections being referred to as Namespaces. Namespaces are used to group data items according to their purpose within the sensor (*e.g.* read-only sensor data, sensor configuration data, *etc.*).

The following summarizes the terms above and used throughout this document.

Tag: Unique identifier used to refer to a data item.

Tag-Value: A combination of unique Tag value and associated data.

Namespace: A grouping of Tags according to the purpose of the data items in the sensor.

API Version: The identifier for the collection of Namespaces implemented within the sensor.

Parameter Tables: Detail the Namespaces associated with an API Version.

Message Identifier: Unique protocol message identifier used to indicate its function and the Namespace the message interacts with.

Figure 1 shows the relationship between protocol message API Version, Message Identifier, and Tag with Namespaces grouped by API Version.

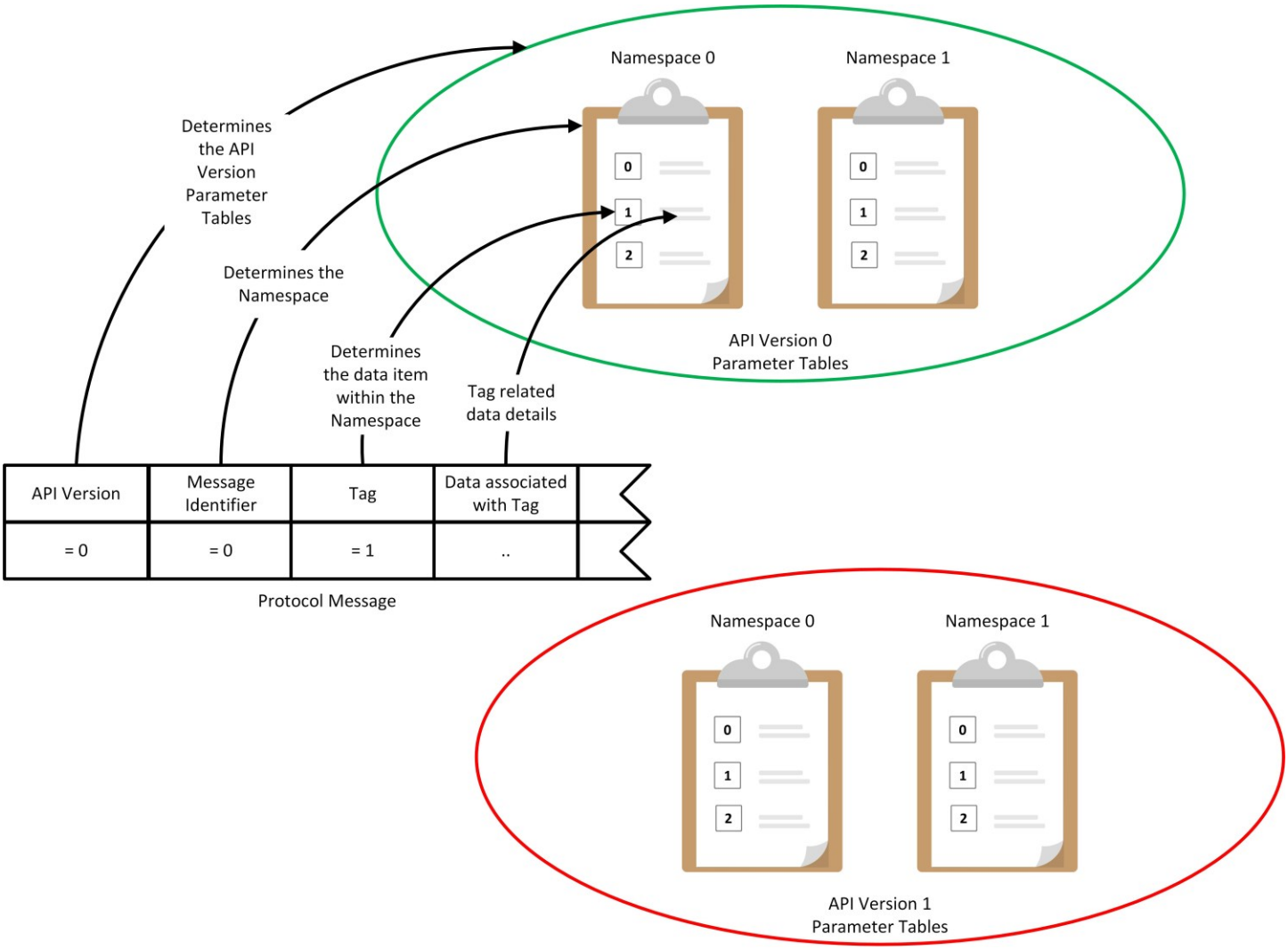

Figure 1: Relationship of protocol message API Version, Message Identifier, and Tag with Namespaces grouped by API Version

4 Message Structure

Messages are either of Uplink type, originating at the sensor, or Downlink type, terminating at the sensor. This section describes the protocol message elements that are common to both message types, and elements unique to Uplink messages.

4.1 Common Message Elements

4.1.1 Message Header

The first byte in each protocol message is termed the Message Header. This upper 4-bits of the Message Header contain the API Version. The lower 4-bits of the Message Header contain the Message Identifier.

4.1.2 Message Body

The message body consists of pairs of Tags and associated data values. The LoRaWAN data rate currently in use by the sensor limits the number of Tag-Value pairs the message can contain. Details of associated payload size limits are described in [B].

4.2 Uplink message elements

4.2.1 Status Byte

For uplink messages only, the byte immediately following the Message Header is used to indicate the status of the RS26x device. The upper 2-bits contain the device Battery Status, and the lower 6-bits Device Status Indications.

Battery Status values are treated as an enumeration, with possible values shown in Table 1.

Value	Battery Status	Meaning
0	Critical	Device can lose power at any stage
1	Replace	Battery should be replaced
2	OK	Battery beyond midpoint of lifetime
3	Good	Battery condition healthy

Table 1: Battery Status values

The Device Status bits are treated as a bitfield, with possible values shown in Table 2.

Value	Status	Meaning
0x01	Sensor Fault	The device sensor is not functioning correctly.
0x02	Bandwidth Limitation	The current LoRaWAN data rate is limiting the size of the payload the device can send and messages are being truncated.
0x04	Backlog Messages Available	Locally stored sensor measurement data is available for uplink due to previous uplink attempts not having been successful.
0x08	Backlog Wraparound	Locally stored sensor measurement data has been lost due to the available number of records being exceeded. This typically indicates long disconnection time from the LoRaWAN network.
0x10	Unsupported API	A downlink message has been received encoded with an unsupported API Revision.

Table 2: Device Status bits

5 Message Management

The following describes message level details.

5.1 Encoding and decoding

Messages are encoded and decoded in big-endian format.

5.2 LoRaWAN ports

Messages are always uplinked by the device on LoRaWAN port 1. Downlink messages can be forwarded to ports shown in Table 3.

Table 3: RS26x device port usage

Port Range	Details	Used for Protocol Messages
1 .. 198	Application ports	Yes
199	Semtech Device Management	No
200	Remote Multicast	No
201	FUOTA Fragmentation	No
202	Application port	Yes
203	Firmware Management	No
204 .. 223	Application ports	Yes
224	LoRaWAN Certification	No
225	Multi-Package Access	No
226	Relay	No

6 Message Types

This section describes the message types defined by the protocol.

6.1 Uplink Messages

Uplink messages are sent from the device via the LNS and processed by the end application. The following uplink message types are currently supported.

6.1.1 Sensor Data (Message Identifier 0x00)

Sensor Data uplink messages are associated with Namespace 0 (read-only sensor data).

This message type consists of read-only device sensor related data.

Periodic Sensor Data uplinks are performed according to the setting of the device ***Sensor Update Rate*** parameter. If the device ***Aggregation Mode*** parameter is set to either the ***'Aggregation'*** or ***'Averaging'*** setting, the periodic uplink is performed following the number of sensor readings determined by the ***Aggregation Count*** parameter. For example, a ***Sensor Update Rate*** of 30s, an ***Aggregation Mode*** setting of ***'Aggregation'*** and an ***Aggregation Count*** of 2 will result in a Sensor Data uplink every 60s.

The structure of the Sensor Data uplink message is shown in Figure 2.

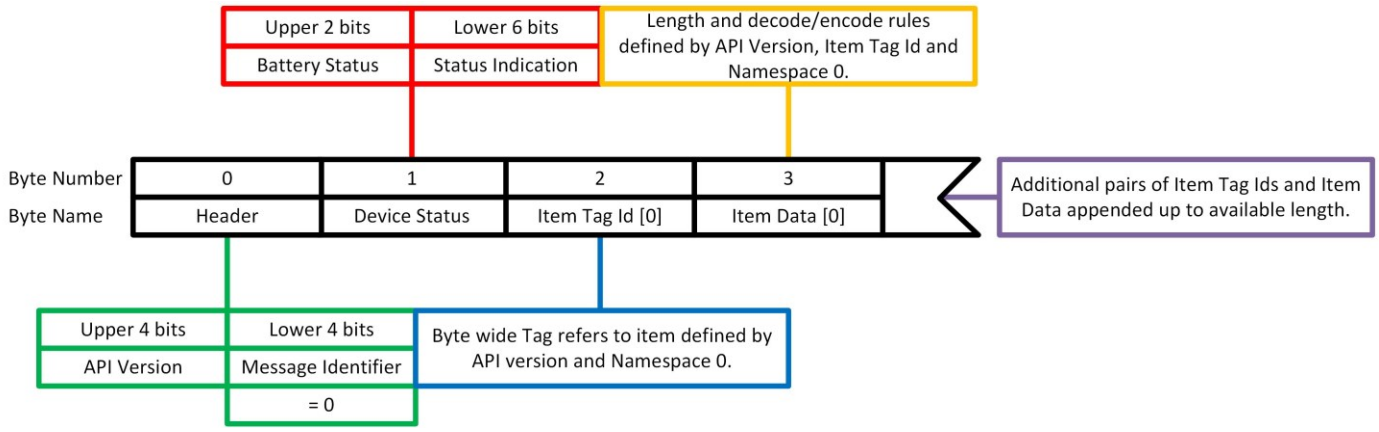

Figure 2: Sensor Data uplink message structure

Note: If records are available in the device Backlog, these will be appended to the Sensor Data uplink messages, to the payload size defined the current LoRaWAN data rate and number of records available.

Note: If the device ***Aggregation Mode*** is set to ***'Aggregation'*** and the resultant payload size exceeds that defined by the current LoRaWAN data rate, the aggregated data is truncated to the first reading associated with the aggregated data, and a Bandwidth Limitation Status Indication annunciated, as described in Section 4.2.1. In this case, the number of aggregated readings should be reduced, or the device data rate increased to allow longer payload lengths.

6.1.2 Configuration Data (Message Identifier 0x01)

Configuration Data uplink messages are associated with Namespace 1 (device parameter data).

This message type consists of device configuration related data (*e.g.* device parameterisation, user-interface statistics, *etc.*).

Configuration Data uplink are performed by the device in response to both Configuration Get and Configuration Set downlink messages.

The structure of the Configuration Data uplink message is shown in Figure 3.

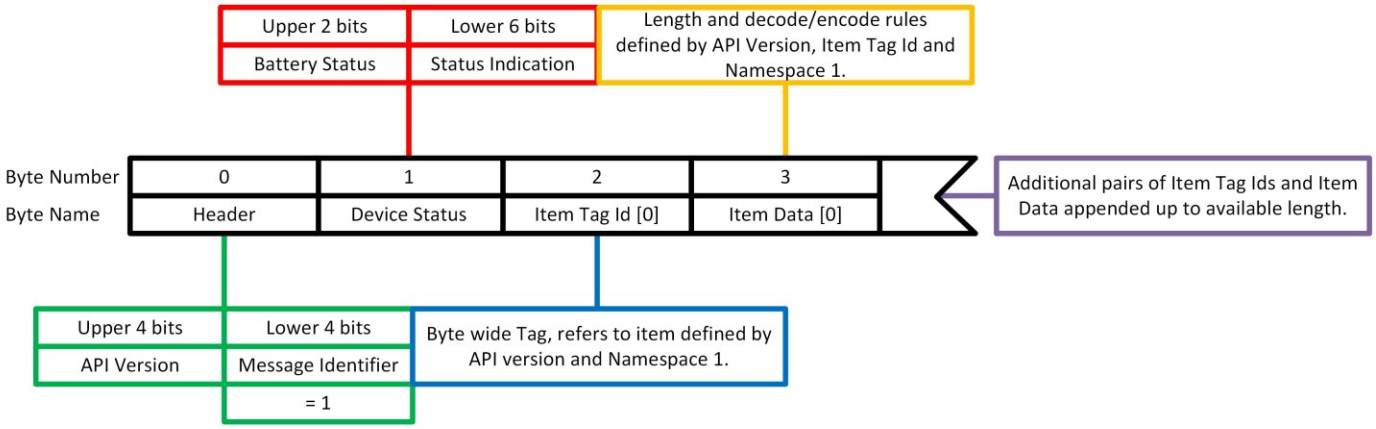

Figure 3: Configuration data uplink message structure

6.2 Downlink Messages

Downlink messages originate at the end application and are sent via the LNS to the device. The following downlink message types are currently supported.

6.2.1 Configuration Get (Message Identifier 0x00)

Configuration Get downlink messages are associated with Namespace 1 (device parameter data).

This message type is sent to the device to query configuration and statistical data.

The structure of the Configuration Get downlink message is shown in Figure 4.

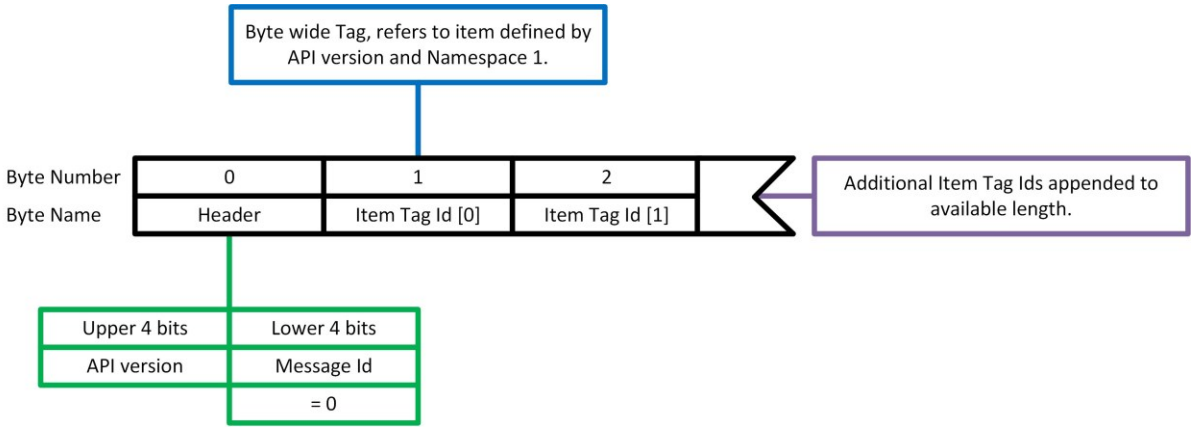

Figure 4: Configuration Get downlink message structure

Note: Configuration Get downlink messages are processed atomically by the device. No response uplink message is issued in event of an invalid Tag being requested.

Note: Configuration Get downlink messages encoded with an unsupported API Version result in a Configuration Data uplink with no Tag related data suffixed and the Unsupported API Version Status Indication set.

6.2.2 Configuration Set (Message Identifier 0x01)

Configuration Set downlink messages are associated with Namespace 1 (device parameter data).

This message type is sent to the device to set configuration data.

The structure of the Configuration Set downlink message is shown in Figure 5.

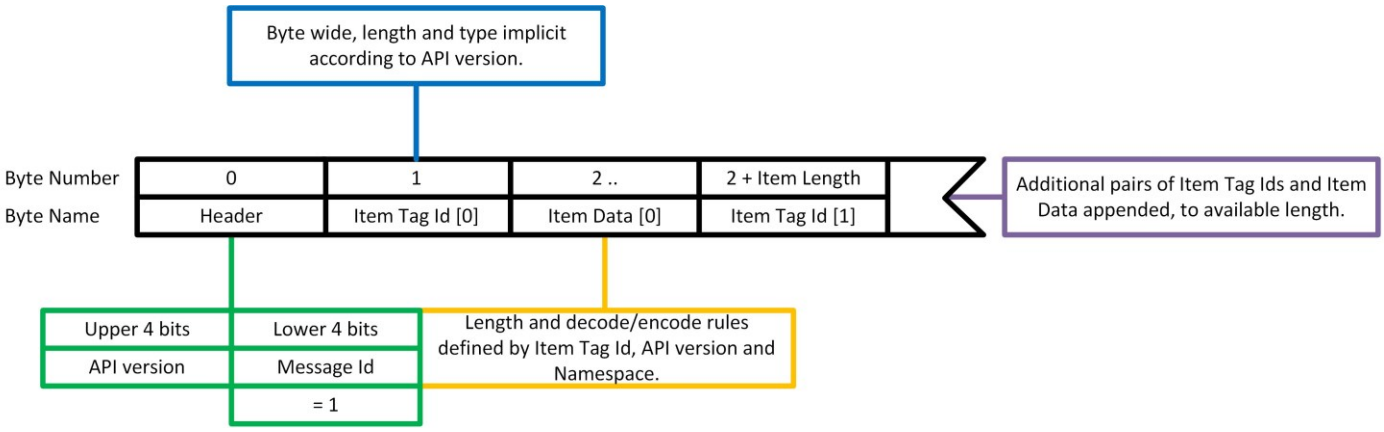

Figure 5: Configuration Set downlink message structure

Note: Configuration Set downlink messages are processed atomically by the device. No response uplink message is issued in event of an invalid Tag being requested.

Note: Configuration Set downlink messages encoded with an unsupported API Version result in a Configuration Data uplink with no Tag related data suffixed and the Unsupported API Version Status Indication set.

---

7 Appendix A: API Parameter Tables

The following describe the data items associated with each API Version. Table keys are described as follows.

- **Name:** This is the parameter name.
- **Tag:** This is the Tag associated with the parameter.
- **Data Size:** This is the number of bytes occupied with the parameter data.
- **Access:** These are the access rights associated with the parameter, with possible values as follows.
  - **R:** Parameter can be read.
  - **W:** Parameter can be written.
- **Storage:** This indicates the parameter storage mechanism, with possible values as follows.
  - **NV:** The parameter is stored in device Non-Volatile storage, can be modified and persists through power cycles.
  - **RAM:** The parameter is stored in device RAM and is lost following a power cycle.
  - **ROM:** The parameter is stored in device read-only storage, cannot be modified and persists through power cycles.
- **Range:** This is allowable range of values that a parameter can be written to.
- **Default:** This is the default value associated with a parameter.
- **Factory Reset:** This indicates whether a parameter is set to its Default value following a Factory Reset of the device parameters.
- **Description:** Details of the parameter.

7.1 API Revision 0

7.1.1 Namespace 0 – Read-only sensor data

Table 4: Sensor Data Parameters

Name	Tag	Data Size (bytes)	Data Type	Access	Storage	Range	Default	Factory Reset	Description
Temperature	0	2	Temperature	R	RAM	-	-	-	This is the last read temperature for standard temperature range (e.g. Internal, External Thermistor) type temperature sensors.
Temperature Backlog	1	6	temperature_ backlog	R	NV	-	-	-	This is a temperature backlog for standard temperature range (e.g. Internal, External Thermistor) type temperature sensors.
Aggregated Temperature	2	5 + (2 * n)	temperature_ aggregate	R	RAM	-	-	-	This is an aggregated temperature reading for standard temperature range (e.g. Internal, External Thermistor) type temperature sensors.
Wide Temperature	3	4	temperature_ wide	R	RAM	-	-	-	This is the last read temperature for extended temperature range (e.g. External RTD) type temperature sensors.
Wide Temperature Backlog	4	8	temperature_ wide_backlog	R	NV	-	-	-	This is a temperature backlog for extended temperature range (e.g. External RTD) type temperature sensors.
Wide Aggregated Temperature	5	5 + (4 * n)	temperature_ wide_aggreg ate	R	RAM	-	-	-	This is an aggregated temperature reading for extended range (e.g. External RTD) type temperature sensors.

7.1.2 Namespace 1 – Sensor configuration data

Name	Tag	Data Size (bytes)	Data Type	Access	Storage	Range	Default	Factory Reset	Description
Friendly Name	0	1..21	char_string	RW	NV	-	RS26x	Y	This is a textual name that can be used to uniquely identify the sensor.
Sensor Type	1	1	enum_8	R	RAM	0 - 8	-	-	Indicates the sensor type, possible values as follows. 0: None/Invalid 1: Internal Temperature 2: External Thermistor Temperature 3: External One-wire Temperature 4: External Contact 5: External Leak Detector 6: External I2C 7: External SPI 8: External RTD Temperature
Read Period	2	2	uint16_t	RW	NV	>= 30	300	Y	Sets the interval in seconds following which sensor measurements are performed.
BLE Address	3	14	hex_string	R	ROM	-	-	-	This is the unique BLE Address of the sensor.
AU Region	5	1	enum_8	RW	NV	0 - 1	0	Y	For Australia region sensors only, sets the Australia region type of the sensor. Allowable values as follows. 0: AU915 1: AU923
Confirmed Packets	6	1	bool	RW	NV	0 - 1	True	Y	Determines whether confirmation is required from the LNS for LoRaWAN packets. Allowable values as follows. 0: Unconfirmed 1: Confirmed
Confirmed Packets Retries	7	1	uint8_t	RW	NV	0 - 10	4	Y	Sets the number of retries performed by the sensor following no confirmation being received from the LNS following uplink of a Confirmed packet.
LoRa RF Power	8	1	int8_t	R	RAM	0 - 22	-	-	Indicates the current sensor LoRaWAN TX power in dBm.
Downlink Packet Count	12	4	uint32_t	R	RAM	-	0	-	Indicates the number of LoRaWAN packets received during the current join session.

Name	Tag	Data Size (bytes)	Data Type	Access	Storage	Range	Default	Factory Reset	Description
LoRa SNR	13	1	int8_t	R	RAM	-	-	-	Indicates the Signal to Noise Ratio for the last received LoRaWAN packet.
Uplink Packet Count	15	4	uint32_t	R	RAM	-	0	-	This is the number of packets transmitted during the current join session.
Max TX Power	16	1	int8_t	RW	NV	0 - 22	13	Y	This is the limit being applied to the sensor LoRaWAN TX power in dBm.
Region	17	1	enum_8	R	NV	0 - 6	0	N	Indicates the configured region of the sensor. Possible values as follows. 0: Unconfigured 1: EU868 3: US915 4: AU915 5: AU923 6: NZ923
Sub-band	18	1	uint8_t	RW	NV	1 - 8	2	Y	For US915 and AU915 sensors only, sets the LoRaWAN sub-band in use. Allowable values as follows. 1: Sub-band 1 (Channels 0 to 7) 2: Sub-band 2 (Channels 8 to 15) 3: Sub-band 3 (Channels 16 to 23) 4: Sub-band 4 (Channels 24 to 31) 5: Sub-band 5 (Channels 32 to 39) 6: Sub-band 6 (Channels 40 to 47) 7: Sub-band 7 (Channels 48 to 55) 8: Sub-band 8 (Channels 56 to 63)
Reading Aggregate Count	19	1	uint8_t	RW	NV	2 - 10	2	Y	Sets the number of sensor readings to be included in one aggregate reading.
Aggregation Mode	20	1	enum_8	RW	NV	0 - 2	0	Y	Sets the sensor aggregation mode, allowable values as follows. 0: None 1: Aggregation 2: Averaging
Clear Backlog	22	0	action	W	-	-	-	-	Used to clear backlog data stored by the sensor.

Name	Tag	Data Size (bytes)	Data Type	Access	Storage	Range	Default	Factory Reset	Description
Device Type	23	1	enum_8	R	ROM	0 - 1	-	-	Indicates the underlying RM126x module type within the sensor. Possible values as follows. 0: RM1261 1: RM1262
Factory Reset	25	0	action	W	-	-	-	-	Restores the device to factory settings.
Firmware Version	26	1..21	char_string	R	ROM	-	-	-	This is the firmware version of the Canvas Core image.
App Version	27	1..21	char_string	R	ROM	-	-	-	This is the firmware version of the Canvas script image.
LoRa Module Firmware Version	30	1.22	char_string	R	ROM	-	-	-	This is the firmware version of the sensor RM126x LoRaWAN module.
RTC Time	31	4	utc_seconds	RW	RAM	-	0	N	Time in seconds from January 1, 1970 in UTC format.
Software Device Reset	32	0	action	W	-	-	-	-	Triggers a device software reset.
Thermistor 560 Cal Actual	33	8	dual_float	R	NV	-	-	N	Two 32-bit floating point numbers: actual resistor, ADC counts
Thermistor 330k Cal Actual	34	8	dual_float	R	NV	-	-	N	Two 32-bit floating point numbers: actual resistor, ADC counts
Heartbeat LED Flash Period	35	1	uint8_t	RW	NV	0 - 60	10	Y	Sets the interval in seconds following which the sensor Heartbeat LED is flashed. Setting this to 0 disables the Heartbeat LED.
BLE RSSI	36	1	int8_t	R	RAM	-128 to +127	-	-	The Received Signal Strength Indication for the current sensor BLE connection.
BLE TX Power	37	1	int8_t	R	RAM	-	-	-	The TX Power in dBm for the current BLE connection.
Device Model	38	1..21	char_string	R	ROM	-	-	-	Returns details of the underlying Canvas Core hardware platform.

Name	Tag	Data Size (bytes)	Data Type	Access	Storage	Range	Default	Factory Reset	Description
Operating Mode	39	1	uint8_t	RW	NV	0	0	Y	Selects the protocol format used for LoRaWAN uplinks. Allowable values as follows. 0: Ezurio RS26x
Network Time	40	1	uint8_t	RW	NV	0 - 1	0	Y	Determines whether the sensor RTC time is requested from the LNS or manually entered. Allowable values as follows. 0: Automatic 1: Manual
LoRa RSSI	41	1	int8_t	R	RAM	-128 to +127	-	-	The Received Signal Strength Indication for the current sensor LoRaWAN connection.
LoRaWAN Data Rate	42	1	enum_8	R	RAM	0 - 7	-	-	Indicates the transmit data rate in use for the current LoRaWAN connection. Possible values as follows. 0: DR0 1: DR1 2: DR2 3: DR3 4: DR4 5: DR5 6: DR6 7: DR7
Thermistor S-H Coefficient A	48	4	float	RW	NV	-	0.001134	Y	Thermistor A coefficient value.
Thermistor S-H Coefficient B	49	4	float	RW	NV	-	0.0002335	Y	Thermistor B coefficient value.
Thermistor S-H Coefficient C	50	4	float	RW	NV	-	0.0000008876	Y	Thermistor C coefficient value.
Battery Voltage Threshold – Good	51	2	uint16_t	RW	NV	-	2250	-	This is the threshold in millivolts above which battery voltage readings result in a Good Battery Status indication.

Name	Tag	Data Size (bytes)	Data Type	Access	Storage	Range	Default	Factory Reset	Description
Battery Voltage Threshold – Bad	52	2	uint16_t	RW	NV	-	1850	-	This is the threshold in millivolts below which battery voltage readings result in a Bad Battery Status indication.
Battery Voltage Threshold - Critical	53	2	uint16_t	RW	NV	-	1790	-	This is the threshold in millivolts below which battery voltage readings result in a Critical Battery Status indication.
Battery Voltage	54	2	uint16_t	R	RAM	-	-	-	The last read battery voltage in millivolts.

8 Appendix B: Datatype encode and decode

The following describes the structure of datatypes used by the device. Reference uplink decoder and downlink encoders are available at [F], [G], [H] and [K].

Examples of complete messages being encoded and decoded can be found at [I], [J] and [L].

8.1 Temperature

Temperature values are encoded in fixed point format with a scaling factor of 256 (Q Format 8).
This allows temperature values within the range of -128 through 127C to be represented in two bytes.

Refer to [C] for further details.

8.2 UTC Seconds

Timestamp values are encoded as unsigned 32-bit numbers based on the UNIX epoch. This is the number of seconds that have occurred since January 1st, 1970.
Refer to [D] for further details.

8.3 Temperature Backlog

Temperature Backlog values consist of a UTC Seconds timestamp, followed by the Temperature value associated with the timestamp.

8.4 Aggregated Temperature

Aggregated Temperature values consist of an unsigned 8-bit value indicating the number of aggregate values, a UTC Seconds timestamp indicating the timestamp associated with the first Temperature reading, followed by the aggregated Temperature readings.

8.5 Wide Temperature

Wide Temperature values are encoded in fixed point format with a scaling factor of 65536 (Q Format 16).

This allows temperature values within the range of -32768 through 32676C to be encoded in four bytes.

Refer to [C] for further details.

8.6 Wide Temperature Backlog

Wide Temperature Backlog values are structured in the same way as Temperature Backlog values, with the temperature represented in Wide Temperature format.

8.7 Aggregated Wide Temperature

Aggregated Wide Temperature values are structured in the same way as Aggregated Temperature values, with the temperatures represented in Wide Temperature format.

8.8 Char String

Character string data must be null terminated. If an empty character string is being encoded or decoded, the last byte must be set to the null character value (0x00). This allows for payload sizes to be reduced for smaller character string lengths. In event of a character string being blank, a single null character must be included to indicate this. Character string data is encoded in ASCII format.

8.9 Hex String

Hex strings are always fixed in length and are not null character terminated. The hex data is encoded in ASCII format.

8.10 Floating-point

Single precision floating-point format data is encoded with the hexadecimal representation of the single precision floating-point data.

Refer to [E] for further details.

9 Appendix C: Downlink message JSON encoding

Downlink messages can either read parameter data from the device or write parameter data. JSON messages utilising codec functionality should be structured as follows, depending upon the operation being performed.

9.1 Reading Configuration data

Device configuration data is queried with a JSON message including two fields as shown below.

```
{
  "Message Type" : "Configuration Get",
  "Parameters" : []
}
```

The 'Parameters' field is an array where the names of the Parameters to be read are entered. Details of parameters available for reading are defined in Appendix A: API Parameter Tables.

The following will read back the Friendly Name only.

```
{
  "Message Type" : "Configuration Get",
  "Parameters" : [
                   "Friendly Name"
                 ]
}
```

Entries in the list of Parameters are separated by commas. The following will encode a downlink to read the device Friendly Name and Sensor Type parameters.

```
{
  "Message Type" : "Configuration Get",
  "Parameters" : [
                   "Friendly Name",
                   "Sensor Type"
                 ]
}
```

9.2 Writing Configuration data

Device configuration data is written with a JSON message including a 'Message Type' field set to 'Configuration Set', followed by the names of parameters to write with associated values. Details of parameters available for writing, type information and associated limits are defined in section 7.1.2.

The general form of a write type JSON message is shown below.

```
{
  "Message Type" : "Configuration Set",
  ...
}
```

Where '...' indicates a placeholder for the details of parameters to be written.

Parameters to be updated and associated values follow the Message Type field. The following will set the Friendly Name parameter.

```
{
  "Message Type" : "Configuration Set",
  "Friendly Name" : "Test RS26x Name"
```

```
}
```

Additional parameters for update are separated with commas. The following will encode a downlink to update the Friendly Name and Sensor Update Rate.

```
{
  "Message Type" : "Configuration Set",
  "Friendly Name" : "Test RS26x Name",
  "Sensor Update Rate" : 10
}
```

Enumeration values are written using the textual form of the enumeration value. The following will set the 'Aggregation Mode' parameter to 'Averaging'.

```
{
  "Message Type" : "Configuration Set",
  "Aggregation Mode" : "Averaging"
}
```

10 Appendix D: References

Ref	Details
[A]	Ezurio RS26x product family https://www.ezurio.com/iot-devices/lorawan-iot-devices/rs26x-sensor
[B]	LoRaWAN Regional Parameters RP002-1.03 https://lora-alliance.org/wp-content/uploads/2021/05/RP002-1.0.3-FINAL-1.pdf
[C]	Fixed to Floating Point Converter https://www.rfwireless-world.com/calculators/floating-vs-fixed-point-converter.html
[D]	Unix Epoch Converter https://www.epochconverter.com/
[E]	Floating Point Converter https://www.h-schmidt.net/FloatConverter/IEEE754.html
[F]	RS26x TTI Codec https://github.com/Ezurio/rs26x_tti
[G]	RS26x Chirpstack Codec https://github.com/Ezurio/rs26x_chirpstack
[H]	RS26x Actility Codec https://github.com/Ezurio/rs26x_actility
[I]	RS26x Internal Temperature sensor example messages https://github.com/TheThingsNetwork/lorawan-devices/blob/master/vendor/ezurio/rs26x-int-temp-sensor-codec.yaml
[J]	RS26x External Thermistor sensor example messages https://github.com/TheThingsNetwork/lorawan-devices/blob/master/vendor/ezurio/rs26x-ext-therm-temp-sensor-codec.yaml
[K]	RS26x AWS Codec https://github.com/Ezurio/rs26x_aws
[L]	RS26x External RTD sensor example messages https://github.com/TheThingsNetwork/lorawan-devices/blob/master/vendor/ezurio/rs26x-ext-rtd-temp-sensor-codec.yaml

11 Appendix E: Definitions, Abbreviations and Acronyms

Term	Definition
API	Application Programming Interface.
BLE	Bluetooth Low Energy.
DR	Date Rate.
JSON	Java Standard Object Notation.
LNS	LoRaWAN Network Server.
LoRa	Long Range.
LoRaWAN	Long Range Wide Area Network.
NV	Non-Volatile. Persistent readable and writable device storage.
RAM	Random Access Memory. Non-persistent readable and writable device storage.
ROM	Read Only Memory. Persistent readable device storage.
RSSI	Received Signal Strength Indication.
RTC	Real Time Clock.
RX	Receive.
S-H	Steinhart-Hart. Equation describing the relationship of semiconductor resistance to varying temperature.
TX	Transmit.
UTC	Coordinated Universal Time.

12 Revision History

Version	Date	Notes	Contributor(s)	Approver
1.0	3 March 2025	Initial release	Seokwoo Yoon Rikki Horrigan Damien Fourcade Erik Lins Florian Baumgartl Scott Lederer Randy Scott Greg Leach	Jonathan Kaye
1.1	9 April 2025	Updated for codec release v0.1. Removed Shipping Mode parameter from API Parameter Tables Namespace 1. Moved Battery Voltage parameter in API Parameter Tables Namespace 1 from index 51 to 54. Added Good, Bad and Critical Battery Voltage Threshold parameters to API Parameter Tables Namespace 1 to index 51, 52 and 53. Updated parameter names in line with XBit Sensor Applet.	Greg Leach	Jonathan Kaye
2.0	25 Aug 2025	Updated for Product Release.	Seokwoo Yoon Robert Gosewehr Rikki Horrigan Greg Leach	Jonathan Kaye
2.1	21 Nov 2025	Updated for codec release v0.2.	Seokwoo Yoon Robert Gosewehr Rikki Horrigan Greg Leach	Jonathan Kaye

Ezurio's products are subject to standard Terms & Conditions.