ONE-NET Device Payload Format

Version 2.3.0

August 14, 2012

Table of Contents

1 Overview

In order for ONE-NET Devices to interact with other ONE-NET Devices, and for application programs to access information from ONE-NET Devices, the format and content of the payload portion of messages must be consistent across all ONE-NET Devices. This document contains a description of the various formats of ONE-NET Device payloads.

This information allows application software to control and gather information from ONE-NET Devices. Although a specific ONE-NET Device may support only one function at a time, the payload formats are designed to accommodate ONE-NET Devices that contain many functional instances (or Units). For example, a ONE-NET switch can contain as many as 15 physical toggle switches. The events (switch on/switch off) for each of these individual switches (Units) will be sent to the appropriate ONE-NET Device, either a peer Client or the Master. The payload format is also designed to allow for the 15 physical Units on a ONE-NET Device to be of differing Unit Types.

2 Parsing Mechanisms

Please see Section 4.2.9.1.3.1 for the different parsing mechanisms of Application messages. Recall from that Section that there are four of them, plus any user-defined parsing mechanisms. The four defined by ONE-NET are…

1. ON_APP_MSG
2. ON_APP_MSG_TYPE_2
3. ON_APP_MSG_TYPE_3
4. ON_APP_MSG_TYPE_4

This document will go into more details for each type.

3 Message Classes

The Message Class tells the receiving device what it should do with each message. The Message Class options fall into three categories.

1. Status Messages – The sending device INFORMS the recipient device of the sending device's status (i.e. MY switch is "on").
2. Queries – The sending device is ASKING the recipient device for its status (i.e. Is YOUR switch "on" or "off"?).
3. Commands – The sending device is COMMANDING the recipient device to change its status to what is in the message (i.e. turn YOUR switch "on"). The Message Classes defined by ONE-NET are in the following chart.

Message Class	Message Family	Raw Hex Value	Interpretation
STATUS	STATUS	0x0	Sending device/unit is informing the recipient device of the sending device's status not in response to any action by the recipient device. Status Messages will generally be sent as a Single Data Message.
STATUS CHANGE	STATUS	0x1	Sending device is informing the recipient device of the sending device's status not in response to any action by the recipient device. The sending device/unit has just had a CHANGE in status in the very recent past. Status Changes will generally be sent as a Single Data Message.
STATUS QUERY RESPONSE	STATUS	0x2	Sending device is informing the recipient device of the sending device's status in response to a regular (i.e. not "fast") query by the recipient device. Status Query Responses will generally be sent as a Single Data message.
STATUS FAST QUERY RESPONSE	STATUS	0x3	Sending device is informing the recipient device of the sending device's status in response to a fast query by the recipient device. Status Fast Query Responses will generally be sent as an ACK message to the recipient's Fast Query message.
STATUS COMMAND RESPONSE	STATUS	0x4	Sending device is informing the recipient device of the sending device's status in response to a COMMAND by the recipient device. Status Command Responses will generally be sent as an ACK message to the recipient's Command message.

COMMAND	COMMAND	0x5	Sending device/unit is commanding the recipient device to change the recipient device/unit's status to the one contained in the message. Commands will generally be sent as a Single Data message.
QUERY	QUERY	0x6	Sending device/unit is requesting that the recipient device send the recipient device/unit's status. This response containing the recipient device / unit's status does not need to be in the ACK message. Queries will generally be sent as a Single Data message.
FAST QUERY	QUERY	0x7	Sending device/unit is requesting that the recipient device send the recipient device/unit's status immediately. The status should be contained in the ACK message. Queries will generally be sent as a Single Data message.

Figure 3.1 – PCON Structure for NACK Packets

4 “Normal” Single-Packet Payloads

Most single-packet ONE-NET Device application messages are of the ON_APP_MSG type, contain 40 bits and contain message type, message class, source unit, destination unit, and a 20-bit integer value. ON_APP_MSG_TYPE_2 has an identical parsing mechanism, but the source and destination units are left off and 28 bits are used for the signed integer. In addition there is a single instance of a message that requires a 24-bit UNSIGNED integer. That leaves room for ONE, not two units. Whether the unit is a destination or source unit is not specified by ONE-NET, but is instead left to the application code to determine. This is the ONA_COLOR message type, which contains a 24-bit UNSIGNED integer containing the 24 bits of RGB(Red-Green-Blue) information.

These “normal” single-packet payloads are described in this section. As mentioned, there are “special case” single-packet payloads that are parsed differently as the data they contain cannot fit into a 20-bit signed integer. These special cases are listed in Section 5.

This “normal” payload consists of 5 bytes (40 bits), divided into 4 fields:

- Source Unit/Destination Unit (8 bits)
- Message Class (4 bits)
- Message Type (8 bits)
- Message Data (20 bits)

0 7	8 11	12 19	20 39
Message Type 8 Bits	Message Class 4 bits	Source And Destination Units (4 Bits Each)	Message Data 20 Bits

Figure 4.1 – Single Packet Payload – ON_APP_MSG (40 bits)

0 7	8 11	12 39
Message Type 8 Bits	Message Class 4 bits	Message Data 28 Bits

Figure 4.2 – Single Packet Payload – ON_APP_MSG_TYPE_2 (40 bits)

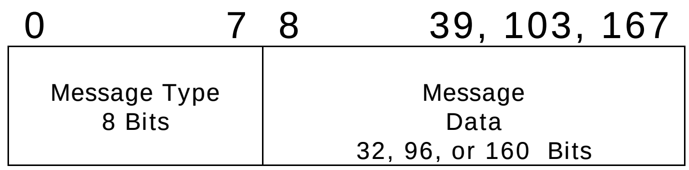

Figure 4.3 – ON_APP_MSG_TYPE_3 (40, 104, or 168 bits)

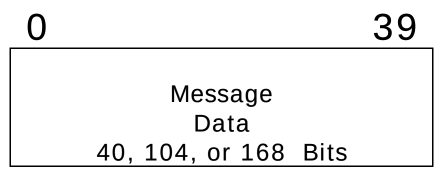

Figure 4.4 – ON_APP_MSG_TYPE_4 (40, 104, or 168 bits)

The following chart lists all ONE-NET defined “normal” message types that can be parsed as ONA_APP_MSG. Developers should see the ona_msg_type_t structure in the one_net_packet.h file for more information. Any Application-specific message types should be added to this list and given a value of at least 0xC0. Note that many message types appear more than once, but with different units.

Message Type	Raw Hex Value	Units / Increments / Values
Switch	0x00	OFF = 0, ON = 1, TOGGLE = 2
Percent	0x01	1/100th of a Percent
Temperature	0x02	1/100th Degrees Kelvin
Humidity	0x03	1/100th of a Percent
Pressure	0x04	1/100th of a Millibar
Pressure	0x05	Millibar
Length	0x06	Angstrom
Length	0x07	Micrometer
Length	0x08	1/10th of a Millimeter
Length	0x09	Centimeters
Length	0x0A	Meters
Depth	0x0B	1/10th of a Millimeter
Depth	0x0C	Centimeters
Depth	0x0D	Meters
Velocity	0x0E	1/10th of A Millimeter Per Second
Velocity	0x0F	Centimeters Per Second
Direction	0x10	1/100th of a Degree
Direction	0x11	1/1000th of a Degree Radians
opening	0x12	0 = Closed, 1 = Open
Seal	0x13	0 = Intact, 1 = Broken
Unit type count	0x14	Unsigned Integer

UNIT TYPE	0x15	Unsigned Integer
TIME	0x16	Microseconds
TIME	0x17	Milliseconds
TIME	0x18	Seconds
TIME OF DAY	0x19	Number Of Seconds After Midnight
VOLTAGE	0x1A	No Units. Query / Fast Query Only
VOLTAGE	0x1B	Volts
VOLTAGE	0x1C	1/100th Of a Volt
VOLTAGE	0x1D	0 = Good, 1 = Bad
SIMPLE TEXT	0x1E	Two ASCII Characters
TEXT	0x1F	NULL-Terminated C-Style ASCII String
ENERGY	0x20	No Units. Query / Fast Query Only
ENERGY	0x21	Watt Seconds
ENERGY	0x22	100 Watt Seconds
ENERGY	0x23	10 KiloWatt Seconds
ENERGY	0x24	MegaWatt Seconds
ENERGY	0x25	KiloWatt Hours
ENERGY	0x26	100 KiloWatt Hours
ENERGY	0x27	10,000 KiloWatt Hours
ACCUMULATED ENERGY	0x28	No Units. Query / Fast Query Only
ACCUMULATED ENERGY	0x29	Watt Seconds
ACCUMULATED ENERGY	0x2A	100 Watt Seconds
ACCUMULATED ENERGY	0x2B	10 KiloWatt Seconds
ACCUMULATED ENERGY	0x2C	MegaWatt Seconds
ACCUMULATED ENERGY	0x2D	KiloWatt Hours
ACCUMULATED ENERGY	0x2E	100 KiloWatt Hours
ACCUMULATED ENERGY	0x2F	10,000 KiloWatt Hours
PEAK ENERGY	0x30	No Units. Query / Fast Query Only
PEAK ENERGY	0x31	Watt Seconds
PEAK ENERGY	0x32	100 Watt Seconds
PEAK ENERGY	0x33	10 KiloWatt Seconds
PEAK ENERGY	0x34	MegaWatt Seconds
PEAK ENERGY	0x35	KiloWatt Hours
PEAK ENERGY	0x36	100 KiloWatt Hours
PEAK ENERGY	0x37	10,000 KiloWatt Hours
POWER	0x38	No Units. Query / Fast Query Only
POWER	0x39	Watts
POWER	0x3A	100 Watts
POWER	0x3B	10 KiloWatts
POWER	0x3C	MegaWatts
INSTANTANEOUS GAS	0x40	No Units. Query / Fast Query Only
INSTANTANEOUS GAS	0x41	Therm Seconds
INSTANTANEOUS GAS	0x42	Therm Minutes
INSTANTANEOUS GAS	0x43	Therm Hours
ACCUMULATED GAS	0x44	No Units. Query / Fast Query Only
ACCUMULATED GAS	0x45	Therm Seconds
ACCUMULATED GAS	0x46	Therm Minutes
ACCUMULATED GAS	0x47	Therm Hours
AVERAGE GAS	0x48	No Units. Query / Fast Query Only
AVERAGE GAS	0x49	Therm Seconds
AVERAGE GAS	0x4A	Therm Minutes

AVERAGE GAS	0x4B	Therm Hours
PEAK GAS	0x4C	No Units. Query / Fast Query Only
PEAK GAS	0x4D	Therm Seconds
PEAK GAS	0x4E	Therm Minutes
PEAK GAS	0x4F	Therm Hours
ANGULAR VELOCITY	0x50	Rotations Per Minute
ANGULAR VELOCITY	0x51	Rotations Per Second
ANGULAR VELOCITY	0x52	Rotations Per Hour
ANGULAR VELOCITY	0x53	1/1000th Rotations Per Minute
ANGULAR VELOCITY	0x54	1/1000th Rotations Per Second
ANGULAR VELOCITY	0x55	1/1000th Rotations Per Hour
ANGULAR VELOCITY	0x56	Degrees Per Second
ANGULAR VELOCITY	0x57	Degrees Per MilliSecond
ANGULAR VELOCITY	0x58	Degrees Per MicroSecond
ANGULAR VELOCITY	0x59	Radians Per Second
ANGULAR VELOCITY	0x5A	Radians Per MilliSecond
ANGULAR VELOCITY	0x5B	Radians Per MicroSecond

Figure 4.5 – Single Packet Payload – Normal Message Types

4.1 Source Unit / Destination Unit

The SRC Unit/DST Unit field contains two 4-bit fields. The high order 4 bits represent the SRC Unit, or the unit number on the sending device that is originating this message. The low order 4 bits represent the DST Unit, or the unit number on the receiving device to which this message is addressed.

4.2 Sample Messages

Scenario: Device 002, Unit 1 is an input switch peered with Device 003, unit 2, which is a relay. Let's assume that both the switch and the relay are currently off (hence the message type is ONA_SWITCH, which is 0, and the current state of both units is ONA_OFF, which is 0.

4.2.1 Device 002 Decides To Send Unit 1's Status To Device 003, Unit 2 For Whatever Reason

This will be a status message. The type of Status message will be ONA_STATUS because nothing prompted 002 to do this. No message was sent to device 002 and Unit 1 was not flipped. Perhaps device 002 is simply programmed to give regular updates to device 003.

Device 002, Unit 1 sends an ONA_STATUS message to device 003, Unit 2. What device 003 does with this message will depend on the application code. However, a very common setup for simple devices like relays is to assume that any message sending a status message to them will be a light switch telling them to turn off or on. Therefore many devices will treat all status messages as though they were command messages. However, once again, ONE-NET DOES NOT MANDATE THIS BEHAVIOR. ONE-NET provides different types of status messages in case the setup is more complex. For example, devices may want to know whether this message was prompted by a light being switched (i.e. an event driven status CHANGE) versus a timer-related regular message. Hence the sending device gives as much information as possible for what prompted this message. Certain devices will have more complex decision making processes. For example, a relay may go on if and only if TWO light switches are both on. Hence the light switch should NOT assume that it should COMMAND the relay to match its status. Hence the light switch merely gives the relay as much information as possible. The application code must interpret the message. Again, though, generally simple relay devices will treat all status messages as commands.

Unit 2 of Device 003 should ACK the message. It will generally send back Unit 2's status. However, this is only REUIRED for commands. If it treated the status update as a command, it will return its new status with Message Class "Status Command Response". Otherwise it will either simply ACK or send back an ONA_STATUS message.

4.2.2 Unit 1 Of Device 002 Has Its Switch Flipped

This will be a status message of type ONA_STATUS_CHANGE. Everything about the previous ONA_STATUS message is relevant. There is now the added information that the status of Device 002, Unit 1 has changed. Very often, state machines will only process information when something has changed. This message might trigger some code that a regular ONA_STATUS message would not.

4.2.3 Unit 1 Of Device 002 Commands That Device 003, Unit 2 Turn On

This will is a command message. The state of Device 002, Unit 1 is not transmitted. Instead the NEW state of Device 003, Unit 2 is enclosed in the packet. Device 003, Unit 2 should turn on and return an ACK with a Message Class of "Status Command Response".

4.2.4 Unit 1 Of Device 002 Queries Device 003, Unit 2 For Its Status

There are two types of queries: queries and fast queries. They are identical in that each wants the recipient device to send back its status. The only difference is that the fast query wants the device to send it back right away in the Acknowledgment and the regular query wants the device to send back its status in a subsequent data message. Regular queries are largely obsolete with the introduction of fast querying. However, they are still reported. Device 003, Unit 002 should not change states. It should either simply ACK the message in the case of a regular query, then quickly follow up with a Single Data message of type "Status Query Response", or in the case of a fast query, send it back in the ACK with a "Status Fast Query Response".

Note that "fast querying" is sometimes colloquially known as "polling".

5 “Special” Single-Packet Payloads

The following message types do not have message data that can renderable as a 20-bit signed integer. Therefore they require “special” parsing.

5.1 ONA_SEC_SINCE_1970 (Message Type 0x80)

This message type can be useful when determining system time. The data is a 32-bit unsigned integer representing the number of seconds since 1970. Hence it can be useful for any device maintaining the common system time as the number of seconds since January 1, 1970. If the network devices are powerful enough to use a compiler that can use the standard time.h C library, two or more devices can sync their system times. The format of this payload is…

Message Type 0x80 8 Bits	Message Data 32 Bits – Number of Seconds since January 1, 1970

Figure 5.1 – ONA_SEC_SINCE_1970 Message Type Payload

5.2 ONA_DATE (Message Type 0x81)

This is a simple Month / Day Year Message. The format of this payload is…

Message Type 0x81 8 Bits	Message Class 4 Bits	Month (1 – 12) 4 Bits	Day (1 – 31) 8 Bits	Year (0 – 2100) 16 Bits

Figure 5.2 – ONA_DATE Message Type Payload

5.3 ONA_PROGRAM_RUN_TIME (Message Types 0x82 and 0x83)

This is the number of milliseconds or seconds since the program started or the machine or chip “booted”. It can be valuable when syncing two devices. The payloads are as follows. Use message 0x82 to send the program time in milliseconds or 0x83 to send it in seconds.

Message Type 0x82 8 Bits	Elapsed Program Time In Milliseconds 32 Bits

Figure 5.3 – ONA_PROGRAM_RUN_TIME_MS Message Type Payload

Message Type 0x83 8 Bits	Elapsed Program Time In Seconds 32 Bits

Figure 5.4 – ONA_PROGRAM_RUN_TIME_SEC Message Type Payload

Note that very often the elapsed program time can fit in a 28-bit signed integer. If it can, consider setting a parsing mechanism of ON_APP_MSG_TYPE_2. In this way, you can include a message class. If the data cannot fit, the recipient device will have to deduce the message class.

5.4 ONA_COLOR (Message Type 0x84)

This is the normal 24 bit unsigned integer RGB (Red-Green-Blue) color message used everywhere. There is room for one unit, but not two. Thus the recipient will have to deduce whether that unit is the source or the destination unit, which should be easily deducible from the message class. The other unit should be assumed to be ONE_NET_DEV_UNIT, or the device as a whole.

Message Type 0x84 8 Bits	Message Class 4 Bits	Unit 4 Bits	Red Value (0-255) 8 Bits	Green Value (0-255) 8 Bits	Blue Value (0-255) 8 Bits

Figure 5.5 – ONA_COLOR Message Type Payload