CHAPTER 24

Message Formats

List of Figures

List of Tables

Acronyms

CINR	Carrier to Interference + Noise Ratio
DSCP	Diffserv Code Point
GHz	gigahertz
IOCTL	input/output control
IP	Internet Protocol
kHz	kilohertz
MAC	media access control
MDL	Metadata Description Language
RF	radio frequency
RFNM	radio frequency network message
RSSI	received signal strength indicator
TCP	Transmission Control Protocol
TE	Traffic Engineering
TAI	International Atomic Time
TLV	Type-Length-Value
TmNS	Telemetry Network Standard
TxOp	transmission opportunity

This page intentionally left blank.

CHAPTER 24

Message Formats

Application messages are message structures used to pass information between applications using an application layer protocol. The bit numbering, bit ordering, and byte ordering conventions used in this chapter are described in Chapter 21 Appendix 21-B.

24.1 Type-Length Value Structure

The Type-Length-Value (TLV) message structure is depicted in Figure 24-1.

TYPE	LENGTH	VALUE

Figure 24-1. Type-Length-Value Structure

The TLV field descriptions are provided below.

Field Name	***Field Length***	***Field Description***
Type	Fixed	Type of the TLV message, encoded as a binary value
Length	Fixed	Length, typically in bytes, of the entire TLV message (including Type and Length fields)
Value	Length = Length field value – (the length of the Type field + length of the Length field)	Data portion of the TLV message

For each defined TLV sequence, the Type and Length field sizes are fixed, a specific set of Types are defined, and each Value field may encode one or more pieces of information, as depicted in Figure 24-2.

TYPE	LENGTH	VALUE			
Type	Length	Value1	Value2	…	ValueN

Figure 24-2. Multi-Value TLF Structure

NOTE

The following figure represents an ASCII message “ABCD” (0x41, 0x42, 0x43, 0x44) with a Type of 0xA97. The Type field is two bytes long and the Length field is one byte long.

	Type		Length	Value			
Byte Number	1	2	3	4	5	6	7
Data	0x0A	0x97	0x07	0x41	0x42	0x43	0x44

24.2 TmNSMessage

A *TmNSMessage* shall contain a *TmNSMessageHeader* and may contain a *TmNSMessagePayload* as shown in Figure 24-3.

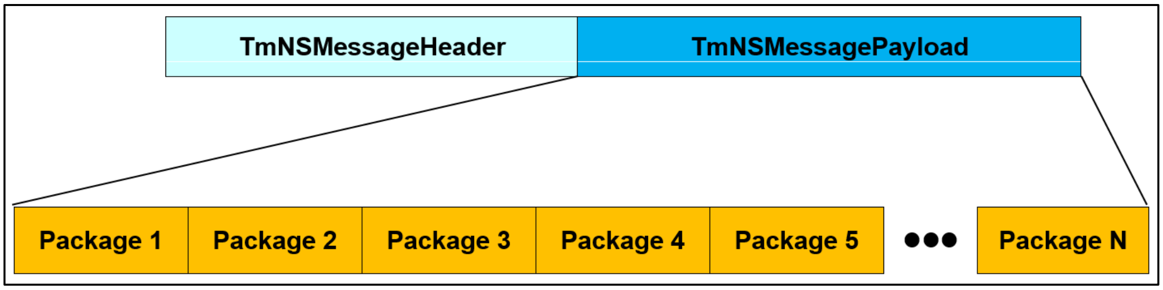

Figure 24-3. TmNSMessage Structure

All *TmNSMessageHeader* and *PackageHeader* fields in the *TmNSMessagePayload* shall use big-endian ordering as specified in Chapter 21 Section B.3 and the bit numbering specified in Chapter 21 Section B.2. *TmNSMessagePayload* fields (e.g., *PackagePayloads* fields described in MDL instance documents) are often based on acquisition data from non-Internet Protocol (IP)-network systems and, therefore, are not required to comply with the big-endian convention.

NOTE	The IP specification defines standard network byte order as big-endian for all numeric values in the IP packet headers. This standard maintains consistency with the IP specification by defining all numeric values in *TmNSMessageHeader* and *PackageHeader* fields of the *TmNSMessage* as following network byte order (i.e., big-endian).

24.2.1 TmNSMessageHeader Structure

The *TmNSMessageHeader* shall contain the following fields and associated bit-widths as outlined in Figure 24-4.

- MessageVersion – 4 bits
- OptionWordCount – 4 bits
- Reserved – 4 bits
- MessageType – 4 bits
- MessageFlags – 16 bits
- MessageDefinitionID – 32 bits
- MessageDefinitionSequenceNumber – 32 bits
- MessageLength – 32 bits
- MessageTimestamp – 64 bits
- ApplicationDefinedFields – variable (OptionWordCount * 32 bits)

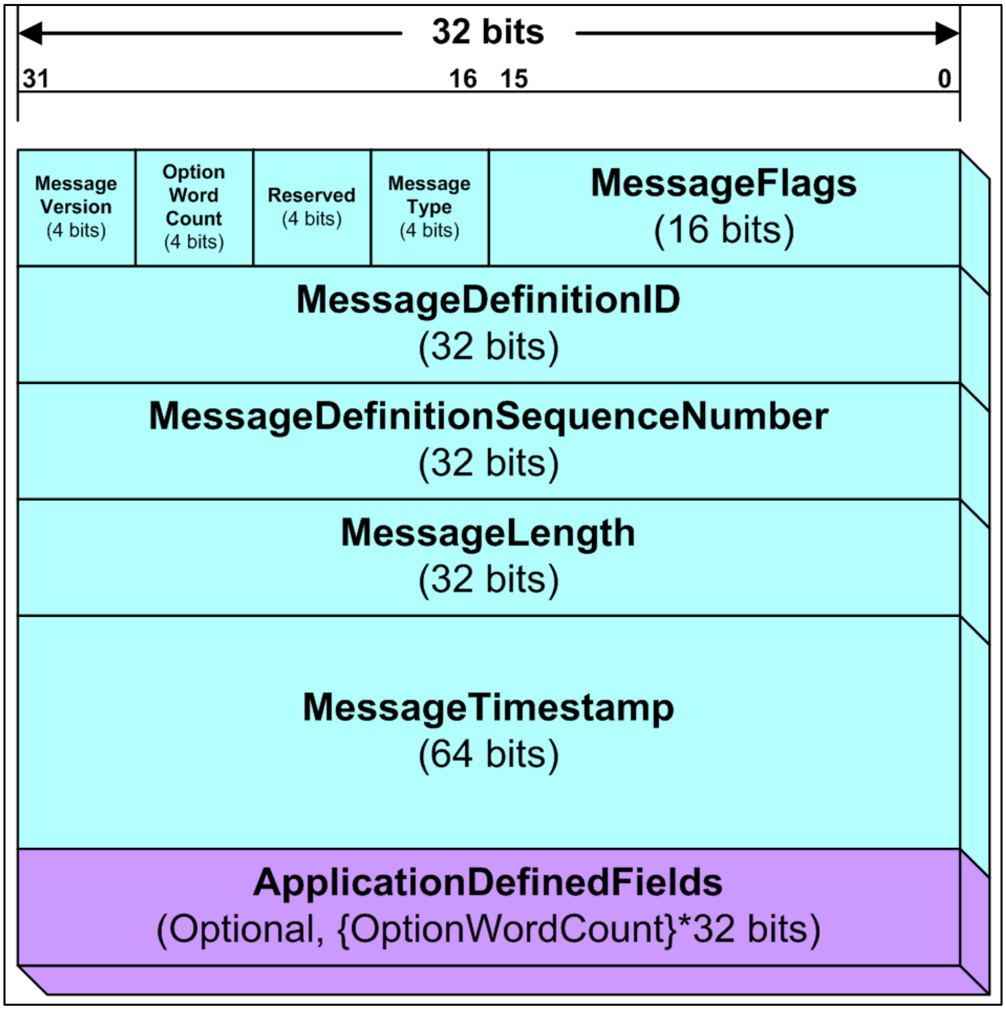

Figure 24-4. TmNSMessageHeader Structure

24.2.1.1 MessageVersion Field

The **MessageVersion** field specifies the version of the *TmNSMessage* protocol. This document defines Message Version 1 (i.e., 4’b0001)

24.2.1.2 OptionWordCount Field

The **OptionWordCount** field shall specify the number of 32-bit words in the **ApplicationDefinedFields**.

24.2.1.3 Reserved Field

This field is reserved for future use. All bits shall be set to zero (4’b0000) on transmission; ignored on reception.

24.2.1.4 MessageType Field

The **MessageType** field specifies the type of the *TmNSMessage*. This document defines the following message types:

- 4’b0000 – TmNSDataMessage

24.2.1.5 MessageFlags Field

The **MessageFlags** field shall provide indicators of *TmNSMessage* options and/or conditions. Using the bit-numbering convention specified in Chapter 21 Appendix 21-B, the bits are defined as follows.

- **Reserved for Future Use** bits (bits 15-8). All bits shall be set to zero (8'h00) on transmission; ignored on reception.
- StandardPackageHeaderFlag bit (bit 7):
  - 1'b0 – At least one *Package* uses a *PackageHeader* completely described in an MDL instance document or at least one *Package* does not contain a *PackageHeader*
  - 1'b1 – All *Packages* use the Standard *PackageHeaders* (see Subsection 24.2.2.1.1)

  For TmNSMessages that do not contain Packages, this bit shall be set to 1'b0.
- PlaybackDataFlag bit (bit 6):
  - 1'b0 – Live data (from source)
  - 1'b1 – Playback data
- MessageFragmentationFlags bits (bits 5-4):
  - 2'b00 – Complete *TmNSMessage*
  - 2'b01 – *TmNSMessage* with the first fragment
  - 2'b10 – *TmNSMessage* with a middle fragment
  - 2'b11 – *TmNSMessage* with the last fragment

  See Chapter 26 Subsection 26.5.3 for more details.
- DataSourceAcquiredDataFlag bit (bit 3):
  - 1'b0 – Acquired data in this *TmNSMessage*
  - 1'b1 – Simulated data in this *TmNSMessage*
- DataSourceTimeLockFlag bit (bit 2):
  - 1'b0 – *DataSource* time locked to IEEE 1588 master clock
  - 1'b1 – *DataSource* time NOT locked to IEEE 1588 master clock
- DataSourceHealthFlag bit (bit 1):
  - 1'b0 – No error in the portion of the *DataSource* generating this *TmNSMessage*
  - 1'b1 – Error in the portion of the *DataSource* generating this *TmNSMessage*
- EndOfDataFlag bit (bit 0)
  - 1'b0 – Normal *TmNSMessage*
  - 1'b1 – End-of-data *TmNSMessage*

  See Chapter 26 Subsection 26.4.2.2 for usage details of this bit.

See Chapter 26 Subsection 26.5.4 for rules governing the merging of the **MessageFlags** field from multiple *TmNSDataMessages*.

24.2.1.6 MessageDefinitionID Field

The *MessageDefinitionID* field shall contain the *MessageDefinitionID* of the *TmNSMessage*.

24.2.1.7 MessageDefinitionSequenceNumber Field

The **MessageDefinitionSequenceNumber** field shall provide a non-negative integer that increments by one for each *TmNSMessage* instance in a sequence of *TmNSMessages*.

See Chapter 26 Subsection 26.5.1 for additional **MessageDefinitionSequenceNumber** rules.

24.2.1.8 MessageLength Field

The **MessageLength** field shall provide the length (in bytes) of the *TmNSMessage* (or fragment), including the *TmNSMessageHeader* and *TmNSMessagePayload* (including padding).

Padding shall be used if a *TmNSMessage* does not fall on a 32-bit boundary.

24.2.1.9 MessageTimestamp Field (64 bits)

The **MessageTimestamp** field shall provide the message base time (in seconds and nanoseconds). The field shall use the lower 64 bits of the IEEE 1588-2008 specified time structure.

See Chapter 26 Subsection 26.5.2 for additional **MessageTimestamp** rules.

24.2.1.10 ApplicationDefinedFields Field (OptionWordCount*32 bits)

ApplicationDefinedFields provide for optional header fields identified by the **`option-kind`** field (similar to Transmission Control Protocol [TCP] Options). Figure 24-5 shows **ApplicationDefinedFields**.

Field Name	***Field Length***	***Field Description***
option-kind	1 byte	Indicates type of optional field
option-length	1 byte	Indicates length in bytes of particular option field + 2 bytes for the kind and length fields
option-data	0 to 58 bytes	Data associated with a particular option field

option-length = option-data length + 2 bytes

TLV Type	**TLV Length**	**TLV Value**
option-kind (1 byte)	**option-length (1 byte)**	**option-data (0 to 58 bytes)**

Figure 24-5. Option-Kind Message Structure

Multiple **option-kind** fields may be included in the **ApplicationDefinedFields** as long as the total **ApplicationDefinedFields** size does not exceed 60 bytes. The **ApplicationDefinedFields** shall fall on a 32-bit boundary (i.e., length shall be an integer number of 32-bit words). For **option-kind** values between 8’h00 – 8’h7F inclusive, neither the **option-length** nor **option-data** fields are included resulting in a length of one byte. For **option-kind** values between 8’h80 – 8’hFF inclusive, both the **option-length** and **option-data** fields are included, resulting in an **option-data** length of **option-length** – 2 bytes. Table 24-1 defines each supported **option-kind** value along with their corresponding **option-length** and **option-data** values.

Table 24-1. ApplicationDefinedFields "option-kind" List

option-kind	Type	option-length	option-data	Comment
8'h00	End of Options	N/A	N/A	Also used for padding to 32-bit boundary
8'h01	No Operation (NOP)	N/A	N/A	Allows individual options to be 32-bit aligned if needed (not required)
8'h02 – 8'h3F		N/A	N/A	Reserved for future allocation
8'h40 – 8'h7F		N/A	N/A	Reserved for implementation-specific or experimental use
8'h80				Reserved for future allocation
8'h81				Reserved for future allocation
8'h82	*DataSource* Configuration	3-32	An implementation-specific structure of configuration for the *DataSource* generating this *TmNSDataMessage*	
8'h83	*DataSource* Error	3-32	An implementation-specific structure of an error condition for the *DataSource* generating this *TmNSDataMessage*	
8'h84				Reserved for future allocation
8'h85	Destination Address	6 18	IPv4 address (unicast, multicast, broadcast) IPv6 address (unicast, multicast, broadcast)	
8'h86	Fragment Byte Offset	6	Byte offset of current fragment (32-bit length)	
8'h87	Package Count	6	Count of number of *Packages* in this message	
8'h88	Ingress Timestamp	8	Timestamp of when a message was most recently received. Timestamp format is 32-bit International Atomic Time (TAI) seconds field followed by 32-bit nanoseconds field.	This is the system time when the receiving entity received this message.
8'h89	Egress Timestamp	8	Timestamp of when a message was most recently transmitted. Timestamp	This is the system time when the transmitting entity sent the message

Table 24-1. ApplicationDefinedFields "option-kind" List

option-kind	Type	option-length	option-data	Comment
			format is 32-bit TAI seconds field followed by 32-bit nanoseconds field.	(e.g., local system time of recorder when it sends a message it received previously).
8'h8A –8'hBF				Reserved for future allocation
8'hC0 – 8'hFF				Reserved for implementation-specific or experimental use

NOTE	The use of **ApplicationDefinedFields'** **`option-kind`** value in the "implementation-specific or experimental use" range is permitted but does not ensure interoperability.

24.2.2 TmNSMessagePayload Structure

The *TmNSMessagePayload* is optional. If present, the *TmNSMessagePayload* shall include one or more *Packages* as illustrated in Figure 24-6.

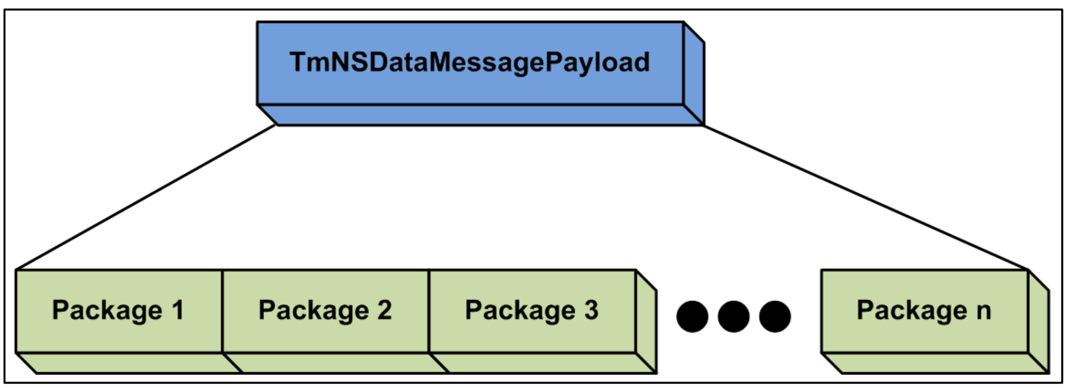

Figure 24-6. TmNSDataMessagePayload Structure

NOTE	The *MessageDefinitionID* specified in the *TmNSDataMessage* header serves as a reference to the structure, content, and ordering of *Package(s)* in the *TmNSDataMessage* payload. For details on how this information is described within an MDL instance document, refer to Chapter 23.

Each *Package* shall include either a *PackageHeader*, a *PackagePayload*, or both. The case where both a *PackageHeader* and *PackagePayload* are present is illustrated in Figure 24-7.

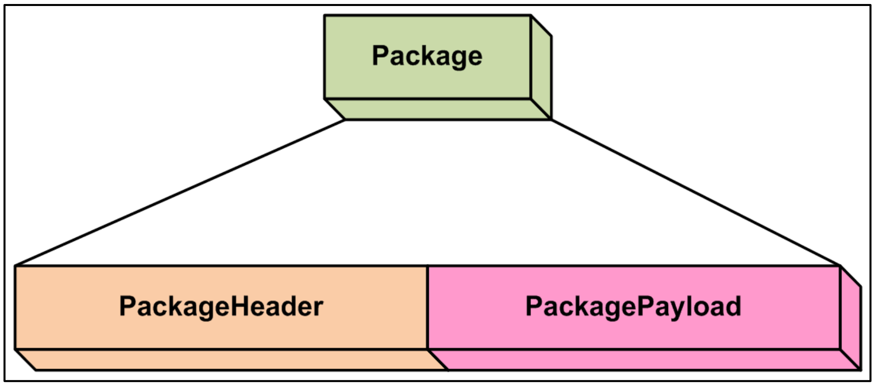

Figure 24-7. Package Structure Containing PackageHeader and PackagePayload

24.2.2.1 Package Header

The *PackageHeader* contains fields that describe the *PackagePayload*. When using a *PackageHeader*, *TmNSDataMessages* shall use either the standard *PackageHeader* or a *PackageHeader* completely described in an MDL instance document.

24.2.2.1.1 Standard PackageHeader

The standard *PackageHeader* shall contain the following fields.

- PackageDefinitionID – 32 bits
- PackageLength – 16 bits
- Reserved – 8 bits
- PackageStatusFlags – 8 bits
- PackageTimeDelta – 32 bits

Figure 24-8 illustrates the standard *PackageHeader*. When using standard *PackageHeaders*, the *Package* shall start and end on 32-bit boundaries.

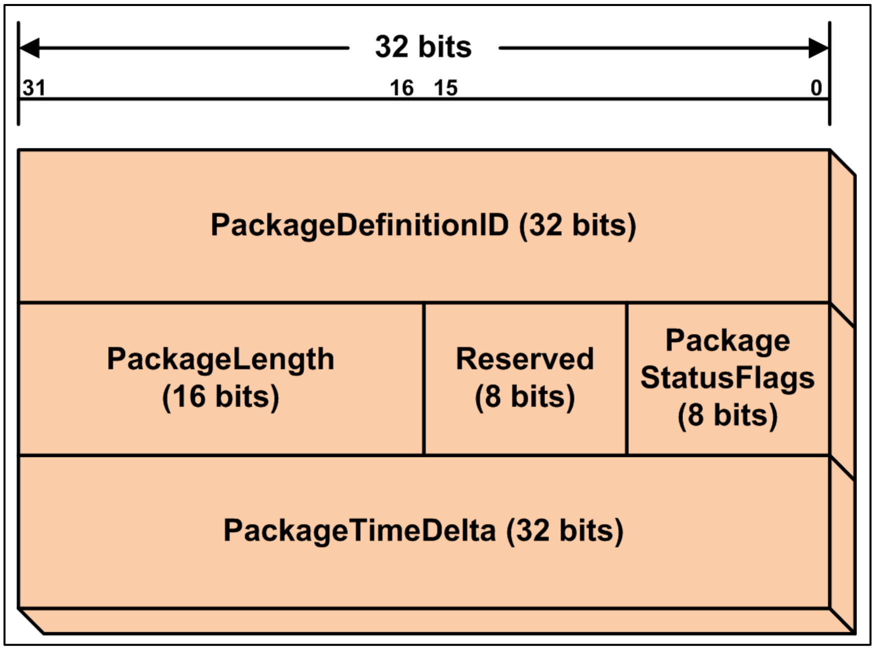

Figure 24-8. Standard PackageHeader Field Structure

24.2.2.1.1.1 PackageDefinitionID Field

The **PackageDefinitionID** field shall contain the *PackageDefinitionID* of the *Package*.

24.2.2.1.1.2 PackageLength Field

The **PackageLength** field shall specify the length, in bytes, of the entire *Package* (including *PackageHeader* and *PackagePayload*, but not including padding) to identify the end of bytes containing *MeasurementData* in the *Package*.

Padding shall be used to ensure a *Package* with a standard *PackageHeader* starts and ends on a 32-bit boundary.

24.2.2.1.1.3 Reserved

All bits shall be set to zero (8'h00) on transmission; ignored on reception.

24.2.2.1.1.4 PackageStatusFlags Field

The **PackageStatusFlags** field may provide indications on specific *MeasurementData* in a *Package* and/or error indications (e.g., parity, out of range, wrong frame size, etc.) of the *DataSource* producing the *MeasurementData*. These flags can be described by an MDL instance document. Each **PackageStatusFlags'** 1'b0 value shall be interpreted as a "no error" condition for that particular condition. Each **PackageStatusFlags** bit not described in an MDL instance document shall be set to 1'b0.

24.2.2.1.1.5 PackageTimeDelta Field

The **PackageTimeDelta** field shall provide the *Package* base time relative to the **MessageTimestamp** field in the *TmNSDataMessageHeader*. The value in the field shall be a non-negative integer that represents nanosecond resolution in the range of 0 to $2^{32} - 1$.

24.2.2.1.2 MDL-Described PackageHeader

A custom *PackageHeader* shall be used if the standard *PackageHeader* is not used for the *Package*. Custom *PackageHeaders* shall be completely described within the MDL instance document that contains the *Package* description.

24.2.2.2 Package Time Measurement Scoping Rules

The Telemetry Network Standard (TmNS) schema in Chapter 23 defines the MeasurementTimeRef element, which is a measurement that is associated with another measurement. There shall be no MeasurementTimeRef elements that reference outside a single package instance within a single message instance.

24.3 Radio Frequency (RF) Network Message

There is one general structure for all RF network messages. The structure consists of a common RF network message header followed by the RF network message payload. The payload consists of one or more TLVs.

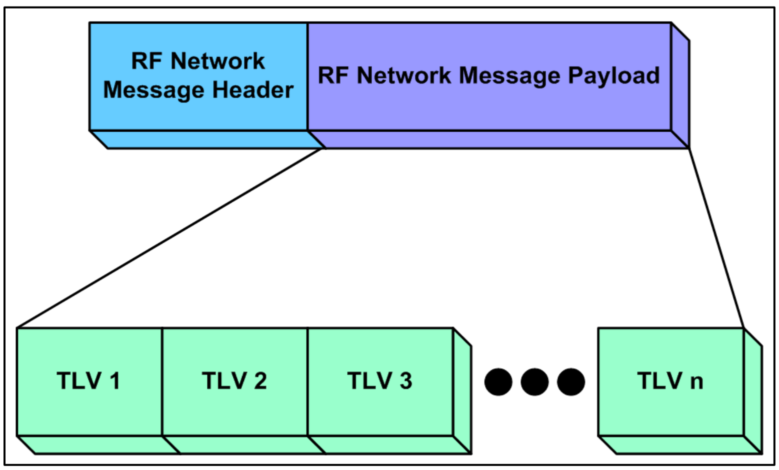

Figure 24-9. RF Network Message Structure

All fields in an RF network message shall use big-endian ordering as specified in Chapter 21 Appendix 21-B.

24.3.1 RF Network Message Header Structure

An RF network message header shall contain the following fields shown in Figure 24-10:

- Message Length – 16 bits
- Destination RF Media Access Control (MAC) address – 16 bits
- Source RF MAC Address - 16 bits
- Message Sequence Number – 32 bits

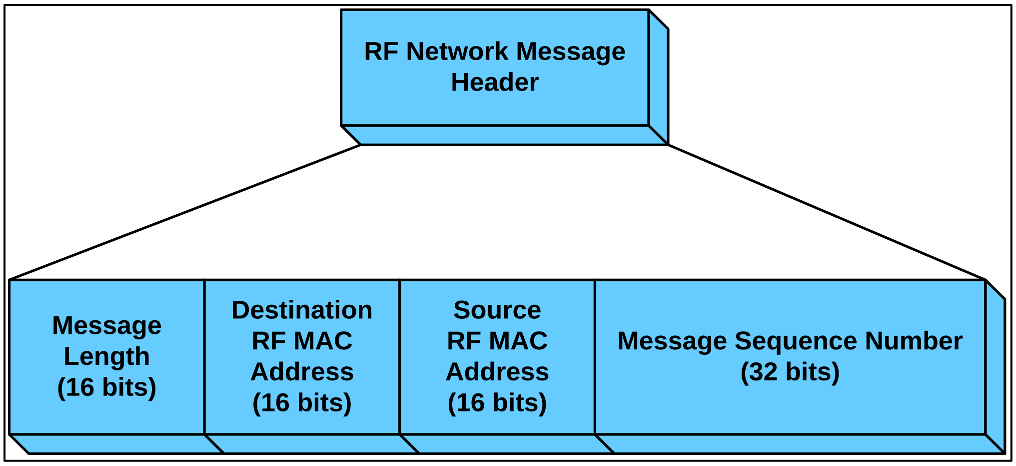

Figure 24-10. RF Network Message Header Structure

24.3.1.1 Message Length

This field indicates the remaining length in bytes of the RF network message. The size of the RF network message is the size of the Message Length field plus the value contained therein.

24.3.1.2 Destination RF MAC Address

This field contains the destination RF MAC address. The combination of the destination RF MAC address and the source RF MAC address identify a particular link for which the RF network message is associated

24.3.1.3 Source RF MAC Address

This field contains the source RF MAC address.

24.3.1.4 Message Sequence Number

The Message Sequence Number serves as an identifier of a particular RF network message. The sequence number shall be associated with the destination RF MAC address and source RF MAC address pair contained in the RF network message header. Entities that send RF network messages shall increment the sequence number associated with a particular destination RF MAC address and source RF MAC address pair after each RF network message is generated.

This value shall be initialized with 32’d0. The first RF network messages produced for a particular destination RF MAC address and source RF MAC address pair shall be 32’d0.

24.3.2 RF Network Message Payload Structure

The RF network message (RFNM) payload consists of one or more TLV structures. The defined TLVs are contained in Table 24-2.

Table 24-2. RF Network Message TLVs

TLV	Type	Descriptions
Transmission Opportunity (TxOp) Assignment	1	This TLV is used by the time-division multiple access scheduler to allocate a transmission opportunity on a radio link.

Table 24-2. RF Network Message TLVs		
TLV	**Type**	**Descriptions**
TxOp ID Acknowledgement Report	2	This TLV is used by a radio to report the acknowledgement of specific TxOps that have been received and processed and applied to the active schedule.
MAC Queue Status Report	3	This TLV is used by a radio to report MAC layer queue level.
Heartbeat	5	This TLV is used to establish a timeout value used by the radio to classify TxOps received from RF link management as stale.
Link Metric	6	This TLV contains an absolute time (converted to an internal representation) and link metric measurements pertaining to a specific radio link.
TE Queue Status Report	10	This TLV is used by a radio to report Traffic Engineering (TE) queue levels for each of the 8 TE queues associated with a particular link on the reporting radio.
Link Transmit Statistics Report	11	This TLV contains the count of IP packets transmitted over the specified RF link.

24.3.2.1 TxOp Assignment TLV

The TxOp Assignment TLV shall be used to allocate transmission opportunities on a radio for the link comprised of the destination RF MAC address and source RF MAC address in the RF network message header. The TxOp Assignment TLV is described in Table 24-3.

Table 24-3. TxOp Assignment TLV			
Field	**Width (bits)**	**Descriptions**	**Value/Range**
Type	8	Type: Transmission Opportunity Assignment	1
Length	8	Length in bytes	13
Center Frequency	16	Carrier or operating frequency given in units of 250 kilohertz (kHz) (up to 16 gigahertz [GHz])	$[0, 2^{16}-1]$
Reserved	8	This field is reserved for future use. The value shall be set to 8’h80.	8’h80
TxOp ID	16	An identifier for the TxOp. If the value of this field is set to zero (16’h0000), no acknowledgement for the TxOp will be provided through the TxOp ID Ack Report TLV.	$[0, 2^{16}-1]$
TxOp Timeout	8	The value specifying the number of consecutive epochs for which this transmission opportunity is valid. Additionally, the value of 8’h0 is reserved to indicate that any existing TxOp with a non-zero remaining timeout value whose interval is wholly contained by Start and Stop Subseconds field of this message is deleted from all future epochs. The value of 8’hFF is reserved to indicate that this TxOp	[0, 255]

Table 24-3. TxOp Assignment TLV

Field	Width (bits)	Descriptions	Value/Range
		has an infinite lifetime and will remain in effect until explicitly deleted or until the transmission heartbeat times out.	
TxOp Start Subseconds	20	The value specifying the fractional sub-seconds portion of a TxOp start time, measured in microseconds relative to the start of the epoch.	[0, 999,999]
TxOp Stop Subseconds	20	The value specifying the fractional sub-seconds portion of a TxOp stop time, measured in microseconds relative to the start of the epoch.	[0, 1,000,000]

24.3.2.2 TxOp ID Acknowledgement Report TLV

The TxOp ID Acknowledgement Report TLV shall be used to deliver one or more ID values of TxOps that have been applied to the transmission schedule of the transceiver. This TLV is not directly accountable to the link identified in the RFNM header of the message containing this TLV, thus a single RFNM may contain ID values from TxOps that were supplied to different links on the transceiver. Any TxOps whose ID value is set to zero (16'h0000) shall not be acknowledged. The TxOp ID Acknowledgement Report TLV is described in Table 24-4.

Table 24-4. TxOp ID Acknowledgement Report TLV

Field	Width (bits)	Descriptions	Value/Range
Type	8	Type: TxOp ID Ack Report	2
Length	8	Length in bytes	2+2N, where 'N' is the number of TxOpIds being acknowledged in this TLV
TxOp ID 1	16	The TxOp ID of the first TxOp being acknowledged in this TLV. Required.	$[1, 2^{16}–1]$
…	…	… Optional.	…
TxOp ID N	16	The TxOp ID of the Nth TxOp being acknowledged in this TLV. Optional.	$[1, 2^{16}–1]$

24.3.2.3 MAC Queue Status Report TLV

The MAC Queue Status Report TLV shall be used to report the MAC layer queue level of the radio for the link comprised of the destination RF MAC address and source RF MAC address in the RF network message header. The MAC Queue Status Report TLV is described in Table 24-5.

Table 24-5. MAC Queue Status Report TLV

Field	Width (bits)	Descriptions	Value/Range
Type	8	Type: MAC Queue Status Report	3
Length	8	Length in bytes	8
Reserved	2	Reserved	2'b00

Table 24-5. MAC Queue Status Report TLV

Field	Width (bits)	Descriptions	Value/Range
Timestamp Seconds	6	The value specifying the seconds portion of a timestamp of when the MAC Queue Status was sampled, measured in seconds and corresponding to the least significant 6 bits of the seconds portion of TAI time.	[0, 63]
Reserved	4	Reserved	4'b0000
Timestamp Subseconds	20	The value specifying the fractional sub-seconds portion of when the MAC Queue Status was sampled, measured in microseconds relative to the timestamp Seconds field.	[0, 999,999]
MAC Queue Level	16	Amount of data (reported in units of 64 bytes, rounded up) buffered in transceiver, pending transmission	$[0, 2^{16}-1]$

24.3.2.4 Heartbeat TLV

The Heartbeat TLV shall be used to deliver an updated transmission heartbeat to a radio. The Heartbeat TLV is described in Table 24-6.

Table 24-6. Heartbeat TLV

Field	Width (bits)	Descriptions	Value/Range
Type	8	Type: Heartbeat	5
Length	8	Length in bytes	4
Timeout	16	Number of future epochs that this radio is authorized to execute TxOps. The value of 65,535 (16'hFFFF) is reserved to indicate a heartbeat that has an infinite lifetime and will remain in effect until explicitly changed.	$[0, 2^{16}-1]$

24.3.2.5 Link Metric TLV

The Link Metric TLV shall be used to deliver receiver statistics for the link comprised of the destination RF MAC address and source RF MAC address in the RF network message header. The Link Metric TLV is described in Table 24-7.

Table 24-7. Link Metric TLV

Field	Width (bits)	Descriptions	Value/Range
Type	8	Type	6
Length	8	Length in bytes	15
Timestamp	32	The time that this snapshot of Link Metric information was taken. This timestamp format consists of the following three subfields: Bits 31-26 - Reserved Bits 25-20 - seconds	 6'b000000 [0-63]

Table 24-7. Link Metric TLV			
Field	**Width (bits)**	**Descriptions**	**Value/Range**
		Time, in seconds, when snapshot was taken, corresponding to the least-significant 6 bits of the seconds portion of TAI time Bits 19-0 - microseconds The fractional sub-second portion of the timestamp, measured in microseconds.	[0-999,999]
Center frequency	16	Indicates the center frequency where measurements are made. The center frequency is given in units of 250 kHz (up to 16 GHz)	$[0, 2^{16}-1]$
RSSI	8	Received signal strength indicator. This is a 2's compliment signed integer indicating the RSSI in 1-dBm step with a maximum range of −127 dBm to 127 dBm. The field is assigned −128 (hex 0x80) when RSSI measurement is not available.	[−128, 127]
CINR	8	Carrier to Interference + Noise Ratio. This is a 2's compliment signed integer indicating the CINR in 1-dB step with a maximum range of −127 dB to 127 dB. The field is assigned −128 (hex 0x80) when CINR measurement is not available.	[−128, 127]
Average channel bit error rate	8	This is an unsigned integer indicating the channel error rate in units of $1/2^8$ with a range of $1/2^8$ to $1-1/2^8$. The field is assigned 0 when channel bit error rate measurement is not available.	$[0, 2^8-1]$
Received IP Packet Count	32	The number of IP packets that have been received over the RF link identified by the RFNM header.	$[0, 2^{32}-1]$

24.3.2.6 Traffic Engineering Queue Status Report TLV

The TE Queue Status Report TLV shall be used to report the queue levels of the eight different TE queues of the radio for the link comprised of the destination RF MAC address and source RF MAC address in the RF network message header. The TE Queue Status Report TLV is described in Table 24-8**.**

Table 24-8. TE Queue Status Report TLV			
Field	**Width (bits)**	**Descriptions**	**Value/Range**
Type	8	Type: TE Queue Status Report	10
Length	8	Length in bytes	27
Reserved	2	Reserved	2'b00
Timestamp Seconds	6	The value specifying the seconds portion of a timestamp of when the TE Queue Status was sampled, measured in seconds and	[0, 63]

Table 24-8. TE Queue Status Report TLV			
Field	**Width (bits)**	**Descriptions**	**Value/Range**
		corresponding to the least significant 6 bits of the seconds portion of TAI time.	
Reserved	4	Reserved	4'b0000
Timestamp Subseconds	20	The value specifying the fractional sub-seconds portion of when the TE Queue Status was sampled, measured in microseconds relative to the timestamp Seconds field.	[0, 999,999]
QoS Policy ID	32	Identifier for the QoS policy associated with this radio link	[0, 2^{32}–1] Default: 0
Version	8	Unique identifier for this specific queue status report: TE Queue depth	0
DSCP Class 0 Queue Level	16	Amount of data (reported in units of 64 bytes, rounded up) in the queue(s) for Diffserv Code Point (DSCP) Class 0 (DSCP values 0 to 7)	[0, 2^{16}–1]
DSCP Class 1 Queue Level	16	Amount of data (reported in units of 64 bytes, rounded up) in the queue(s) for DSCP Class 1 (DSCP values 8 to 15)	[0, 2^{16}–1]
DSCP Class 2 Queue Level	16	Amount of data (reported in units of 64 bytes, rounded up) in the queue(s) for DSCP Class 2 (DSCP values 16 to 23)	[0, 2^{16}–1]
DSCP Class 3 Queue Level	16	Amount of data (reported in units of 64 bytes, rounded up) in the queue(s) for DSCP Class 3 (DSCP values 24 to 31)	[0, 2^{16}–1]
DSCP Class 4 Queue Level	16	Amount of data (reported in units of 64 bytes, rounded up) in the queue(s) for DSCP Class 4 (DSCP values 32 to 39)	[0, 2^{16}–1]
DSCP Class 5 Queue Level	16	Amount of data (reported in units of 64 bytes, rounded up) in the queue(s) for DSCP Class 5 (DSCP values 40 to 47)	[0, 2^{16}–1]
DSCP Class 6 Queue Level	16	Amount of data (reported in units of 64 bytes, rounded up) in the queue(s) for DSCP Class 6 (DSCP values 48 to 55)	[0, 2^{16}–1]
DSCP Class 7 Queue Level	16	Amount of data (reported in units of 64 bytes, rounded up) in the queue(s) for DSCP Class 7 (DSCP values 56 to 63)	[0, 2^{16}–1]

24.3.2.7 Link Transmit Statistics Report TLV

The Link Transmit Statistics Report TLV shall be used to report the number of IP packets transmitted by the transmitter over the link comprised of the destination and source RF MAC address in the RF network message header. Table 24-9 describes the specifics of the link transmit statistics.

Table 24-9. Link Transmit Statistics Report TLV

Field	Width (bits)	Description	Value/Range
Type	8	Type	11
Length	8	Length in bytes	10
Timestamp	32	The time that this snapshot of link transmission statistics was taken. This timestamp format consists of the following three subfields.	
		Bits 31-26 – reserved	6’b000000
		Bits 25-20 – seconds Time, in seconds, when snapshot was taken, corresponding to the least-significant 6 bits of the seconds portion of TAI time.	[0-63]
		Bits 19-0 – microseconds The fractional sub-second portion of the timestamp, measured in microseconds	[0-999,999]
Transmitted IP Packet Count	32	The number of IP packets that have been transmitted over the RF link identified by the RFNM header.	$[0, 2^{32}-1]$

24.4 TSS Messages

TmNS Source Selector (TSS) functionality is described in Chapter 28, but the TSS messages are defined in this section. The TSS messages shall be exchanged between TSS interfaces. There are two types of TSS messages defined:

- TSS Initialization Message
- TSS Data Message

24.4.1 TSS Initialization Message Structure

After initial TCP socket connection is established, the TSS server (e.g., typically a radio) shall send 6 TSS initialization messages. The TSS initialization message structure shall contain the following fields as shown in Figure 24-11.

- Interface Parameter Identifier – 4 bytes
- Interface Parameter – 32 bytes

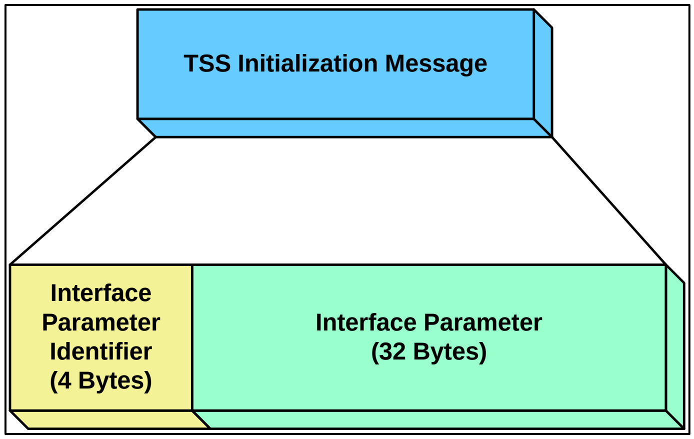

Figure 24-11. TSS Initialization Message Structure

24.4.1.1 Interface Parameter Identifier

The Interface Parameter Identifier field shall contain one of the six values shown in Table 24-10. These values have been chosen to match Linux input/output control (IOCTL) names that are shown so as to ease Linux implementations. The six TSS initialization messages shall be sent in the order shown in Table 24-10.

Table 24-10. TSS Initialization Message Codes

IOCTL Name	Description	Value
SIOCSIFHWADDR	MAC address of the interface	32’h00008924
SIOCSIFMTU	Maximum transfer unit of the interface	32’h00008922
SIOCSIFADDR	Interface IP address of the interface	32’h00008916
SIOCSIFDSTADDR	Destination IP address of the interface when point to point	32’h00008918
SIOCSIFBRDADDR	Broadcast IP address for the interface	32’h0000891a
SIOCSIFNETMASK	Network mask for the interface	32’h0000891c

24.4.1.2 Interface Parameter

The Interface Parameter field shall contain the value associated with the parameter.

24.4.2 TSS Data Message Structure

A TSS data message is a wrapper used to aid specialized routing of network traffic between TmNS networks over other networks. The structure of a TSS data message is shown shall contain the following fields as shown in Figure 24-12.

- Message Length – 16 bits
- Cyclic Redundancy Check (CRC) – 32 bits
- Encapsulated Ethernet Frame – variable length

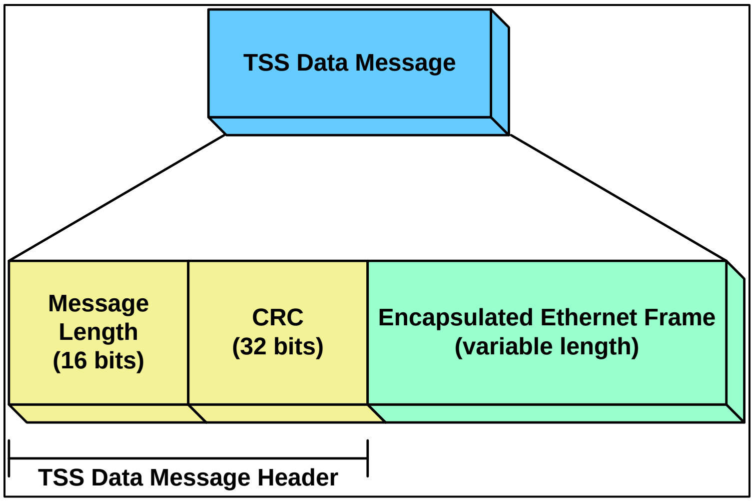

Figure 24-12. TSS Data Message Structure

24.4.2.1 Message Length

This field indicates the remaining length in bytes of the TSS data message. The size of the TSS data message is the size of the Message Length field plus the value contained therein.

24.4.2.2 Cyclic Redundancy Check (CRC)

The CRC field of the TSS data message serves as a message identifier for the TSS data message. The CRC calculation is performed on the entire Encapsulated Ethernet Frame of the message excluding the Ethernet header (e.g., offset by 14 bytes [6 dst + 6 src + 2 type]), with the result being stored in this field.

The polynomial to be used for CRC calculation shall be 32’h82608edb.

The algorithm for the CRC calculation shall be equivalent to that shown in Figure 24-13. The constant POLY is defined as the polynomial listed above.

```c
/*-----------------------------------------------------------------------*/
/*--- get_crcByte - perform byte calculations for CRC process       ---*/
/*-----------------------------------------------------------------------*/
static inline uint32_t get_crcByte(int input)
{
    uint32_t val = input;
    int i;

    for (i=0; i<8; i++)
    {
        if (val & 1)
            val = (val >> 1) ^ POLY;
        else val >>= 1;
    }

    return val;
}


/*-----------------------------------------------------------------------*/
/*--- get_crc32 - calculate the 32-bit CRC of the provided buffer   ---*/
/*-----------------------------------------------------------------------*/
static inline uint32_t get_crc32(unsigned char *data, int sz)
{
    uint32_t remainder, t1, t2;
    int bytes;

    remainder = 0;

    for (bytes = 0; bytes < sz; bytes++)
    {
        t1 = (remainder >> 8) & 0x00FFFFFFL;
        t2 = get_crcByte(((int)remainder^(*(data+bytes)))&0xFF);
        remainder = t1^t2;
    }

    for (bytes = 0; bytes < sizeof(remainder); bytes++)
    {
        t1 = (remainder >> 8) & 0x00FFFFFFL;
        t2 = get_crcByte(((int)remainder)&0xFF);
        remainder = t1^t2;
    }

    return remainder;
}
```

Figure 24-13. Algorithm For CRC Calculation (ANSI C Grammar)

NOTE	A reference implementation of TSS interfaces and functionality is available here.

24.4.2.3 Encapsulated Ethernet Frame (Variable Length)

The Encapsulated Ethernet Frame field encapsulates an entire Ethernet frame so that it can be reproduced after transport.