

## CHAPTER 24

### Message Formats

<b>24.</b>	<b>CHAPTER 24.....</b>	<b>24-1</b>
24.1	General.....	24-1
24.1.1	Type-Length Value (TLV) Structure.....	24-1
24.2	TmNSMessage.....	24-1
24.2.1	TmNSMessageHeader Structure.....	24-2
24.2.1.1	MessageVersion Field (4 bits).....	24-3
24.2.1.2	OptionWordCount Field (4 bits).....	24-3
24.2.1.3	Reserved Field (4 bits).....	24-4
24.2.1.4	MessageType Field (4 bits).....	24-4
24.2.1.5	MessageFlags Field (16 bits).....	24-4
24.2.1.6	MessageDefinitionID Field (32 bits).....	24-5
24.2.1.7	MessageDefinitionSequenceNumber Field (32 bits).....	24-5
24.2.1.8	MessageLength Field (32 bits).....	24-5
24.2.1.9	MessageTimestamp Field (64 bits).....	24-5
24.2.1.10	ApplicationDefinedFields Field (OptionWordCount*32 bits).....	24-5
24.2.2	TmNSMessagePayload Structure.....	24-7
24.2.2.1	<i>Package</i> Structure.....	24-8
24.3	RF Network Message.....	24-10
24.3.1	RF Network Message Header Structure.....	24-10
24.3.1.1	Message Length (16 bits).....	24-11
24.3.1.2	Destination RF MAC Address (16 bits).....	24-11
24.3.1.3	Source RF MAC Address (16 bits).....	24-11
24.3.1.4	Message Sequence Number (32 bits).....	24-11
24.3.2	RF Network Message Payload Structure.....	24-11
24.3.2.1	TxOp Assignment TLV.....	24-12
24.3.2.2	TxOp ID Acknowledgement Report TLV.....	24-13
24.3.2.3	MAC Queue Status Report TLV.....	24-14
24.3.2.4	Heartbeat TLV.....	24-14
24.3.2.5	Link Metric TLV.....	24-15
24.3.2.6	Traffic Engineering (TE) Queue Status Report TLV.....	24-15
24.4	TSS Messages.....	24-17
24.4.1	TSS Initialization Message Structure.....	24-17
24.4.1.1	Interface Parameter Identifier (4 Bytes).....	24-17
24.4.1.2	Interface Parameter.....	24-18

24.4.2	TSS Data Message Structure .....	24-18
24.4.2.1	Message Length Field (16 bits) .....	24-18
24.4.2.2	Cyclic Redundancy Check (CRC) (32 bits).....	24-18
24.4.2.3	Encapsulated Ethernet Frame (Variable Length) .....	24-19

## LIST OF FIGURES

Figure 24-1.	Type-Length-Value (TLV) Structure .....	24-1
Figure 24-2.	Multi-Value TLV Structure.....	24-1
Figure 24-3.	<i>TmNSMessage</i> Structure .....	24-2
Figure 24-4.	<i>TmNSMessageHeader</i> Structure .....	24-3
Figure 24-5.	Option-Kind Message Structure.....	24-6
Figure 24-6.	<i>TmNSDataMessagePayload</i> Structure .....	24-7
Figure 24-7.	Package Structure Containing <i>PackageHeader</i> and <i>PackagePayload</i> ....	24-8
Figure 24-8.	Standard <i>PackageHeader</i> Field Structure .....	24-9
Figure 24-9.	RF Network Message Structure .....	24-10
Figure 24-10.	RF Network Message Header Structure.....	24-11
Figure 24-11.	TSS Initialization Message Structure.....	24-17
Figure 24-12.	TSS Data Message Structure.....	24-18
Figure 24-13.	Algorithm For CRC Calculation (ANSI C Grammar) .....	24-19

## LIST OF TABLES

Table 24-1.	<b>ApplicationDefinedFields “option-kind”</b> List .....	24-6
Table 24-2.	RF Network Message TLVs.....	24-12
Table 24-3.	TxOp Assignment TLV .....	24-12
Table 24-4.	TxOp ID Acknowledgement Report TLV.....	24-13
Table 24-5.	MAC Queue Status Report TLV .....	24-14
Table 24-6.	Heartbeat TLV .....	24-14
Table 24-7.	Link Metric TLV .....	24-15
Table 24-8.	TE Queue Status Report TLV .....	24-16
Table 24-9.	TSS Initialization Message Codes.....	24-17

Distribution Statement A

Approved for public release: distribution unlimited.

## 24. CHAPTER 24

### Message Formats

#### 24.1 General

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 Appendix 21B.

##### 24.1.1 Type-Length Value (TLV) Structure

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

TYPE	LENGTH	VALUE
------	--------	-------

**Figure 24-1. Type-Length-Value (TLV) Structure**

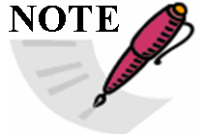
The TLV field descriptions are provided below:

<u>Field Name</u>	<u>Field Length</u>	<u>Field Description</u>
<b>Type</b>	Fixed	Message type, encoded as a binary value
<b>Length</b>	Fixed	Message length, typically in bytes, including Type and Length fields
<b>Value</b>	Length value minus the length of the Type and Length fields	Data portion of 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.

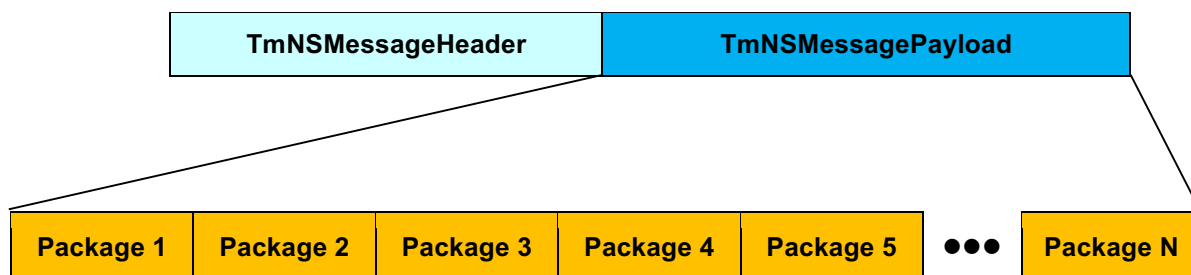
TYPE	LENGTH	VALUE			
Type	Length	Value1	Value2	...	ValueN

**Figure 24-2. Multi-Value TLV Structure**

 <b>NOTE</b>	For example, 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

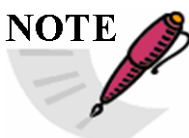
#### 24.2 TmNSMessage

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



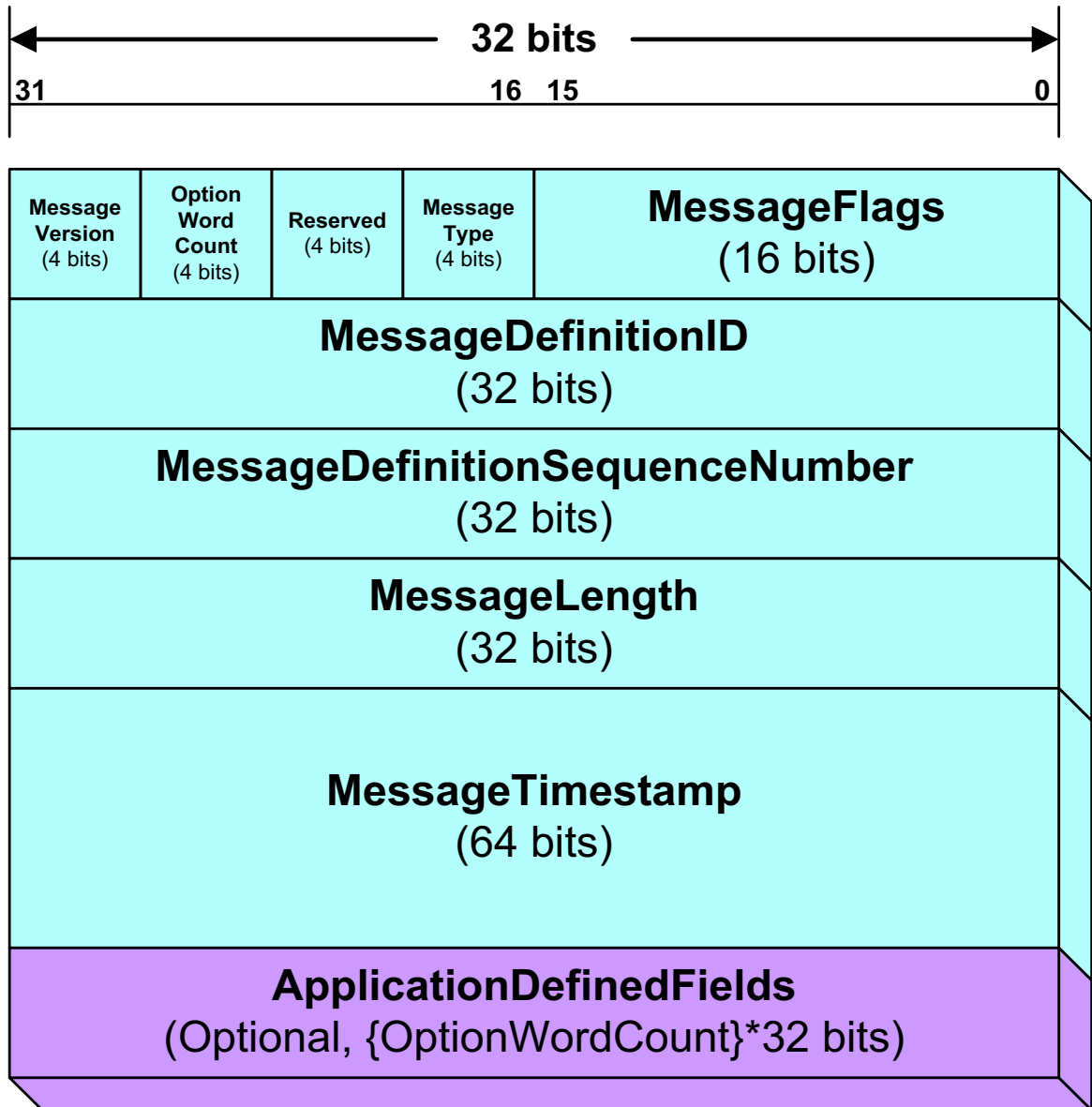
**Figure 24-3. *TmNSMessage* Structure**

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

<p><b>NOTE</b></p> 	<p>The Internet Protocol (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 <i>TmNSMessageHeader</i> and <i>PackageHeader</i> fields of the <i>TmNSMessage</i> as following network byte order (i.e., big-endian).</p>
--	---

### 24.2.1 *TmNSMessageHeader* Structure

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

Figure 24-4. *TmNSMessageHeader* Structure**24.2.1.1 MessageVersion Field (4 bits)**

The **MessageVersion** field specifies the version of the *TmNSMessage* protocol. This document defines the following Message Versions:

- Message Version 1 (i.e., 4'b0001)

**24.2.1.2 OptionWordCount Field (4 bits)**

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

#### 24.2.1.3 Reserved Field (4 bits)

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 (4 bits)

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 (16 bits)

The **MessageFlags** field shall provide indicators of *TmNSMessage* options and/or conditions. Using the bit-numbering convention specified in Chapter 21 Appendix 21B, 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 Section 24.2.2.1.1.1)  
For *TmNSMessages* that do not contain *Packages*, this bit shall be set to 1'b0.
- PlaybackDataFlag bit (bit 6) :
  - 1'b0 – Live data
  - 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 Section 26.5.3, *TmNSDataMessage* Fragmentation, in Chapter 26, TmNSDataMessage Transfer Protocol, 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 Section 26.5.4, Generating *TmNSDataMessages* from Other *TmNSDataMessages* Convention, in Chapter 26, TmNSDataMessage Transfer Protocol, for rules governing the merging of the **MessageFlags** field from multiple *TmNSDataMessages*.

#### 24.2.1.6 MessageDefinitionID Field (32 bits)

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

#### 24.2.1.7 MessageDefinitionSequenceNumber Field (32 bits)

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

See Section 26.5.1, Sequence Numbering Convention, in Chapter 26, TmNSDataMessage Transfer Protocol, for additional **MessageDefinitionSequenceNumber** rules.

#### 24.2.1.8 MessageLength Field (32 bits)

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 Section 26.5.2, Timestamp Convention, in Chapter 26, TmNSDataMessage Transfer Protocol, 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 TCP Options).

<u>Field Name</u>	<u>Field Length</u>	<u>Field Description</u>
<b>option-kind</b>	1 byte	Indicates type of optional field
<b>option-length</b>	1 byte	Indicates length in bytes of particular option field



**option-data**                      0 to 58 bytes                      + 2 bytes for the kind and length fields  
Data associated with a particular option field

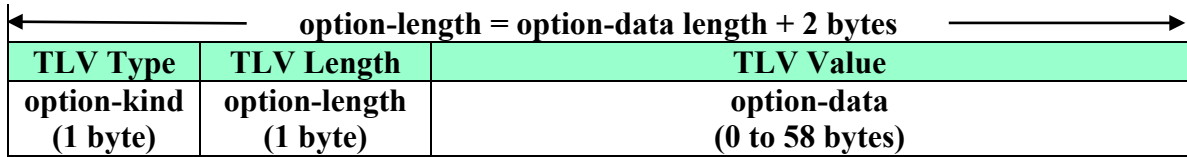


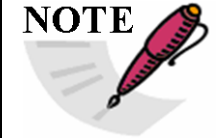
Figure 24-5. Option-Kind Message Structure

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	<i>DataSource Configuration</i>	3-32	An implementation-specific structure of configuration of the <i>DataSource</i> generating this <i>TmNSDataMessage</i>	
8'h83	<i>DataSource Error</i>	3-32	An implementation-specific structure of an error condition of the <i>DataSource</i> generating this <i>TmNSDataMessage</i>	

option-kind	Type	option-length	option-data	Comment
8'h84	Sync Pattern	3-32	An implementation-specific structure of sync pattern	
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 <i>Packages</i> in this message	
8'h88 – 8'hBF				Reserved for future allocation
8'hC0 – 8'hFF				Reserved for implementation-specific or experimental use

	<p><b>NOTE</b> The use of <b>ApplicationDefinedFields'</b> <b>option-kind</b> value in the “implementation-specific or experimental use” range is permitted but does not ensure interoperability.</p>
--	---

### 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.

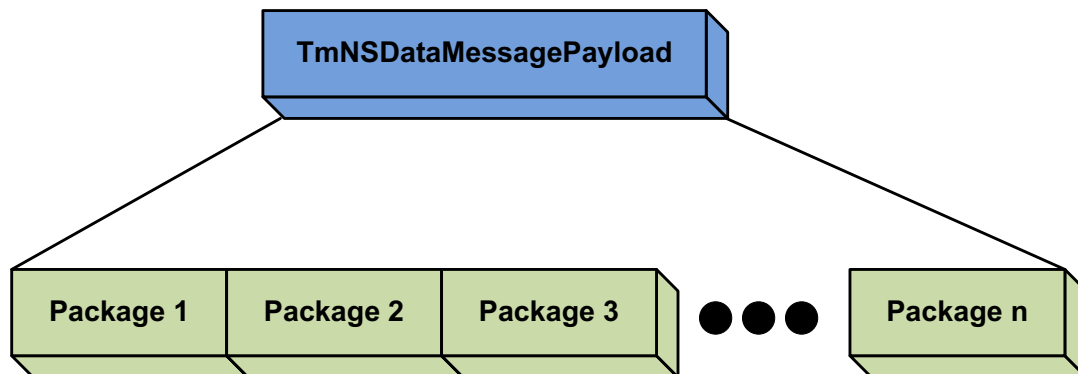

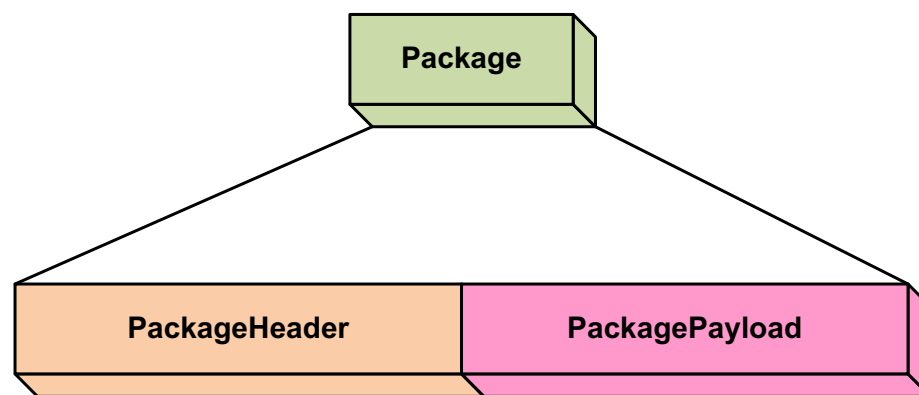


Figure 24-6. *TmNSDataMessagePayload* Structure

<p><b>NOTE</b></p> 	<p>The <i>MessageDefinitionID</i> specified in the <i>TmNSDataMessage</i> Header serves as a reference to the structure, content, and ordering of <i>Package(s)</i> in the <i>TmNSDataMessage</i> Payload. For details on how this information is described within an MDL <i>Instance Document</i>, refer to Chapter 23, Metadata Configuration.</p>
--	--

### 24.2.2.1 Package Structure

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



**Figure 24-7. Package Structure Containing *PackageHeader* and *PackagePayload***

#### 24.2.2.1.1 Package Header

When using a *PackageHeader*, *TmNSDataMessages* shall use either the Standard *PackageHeader* or a *PackageHeader* completely described in an MDL Instance Document.

##### 24.2.2.1.1.1 Standard PackageHeader

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

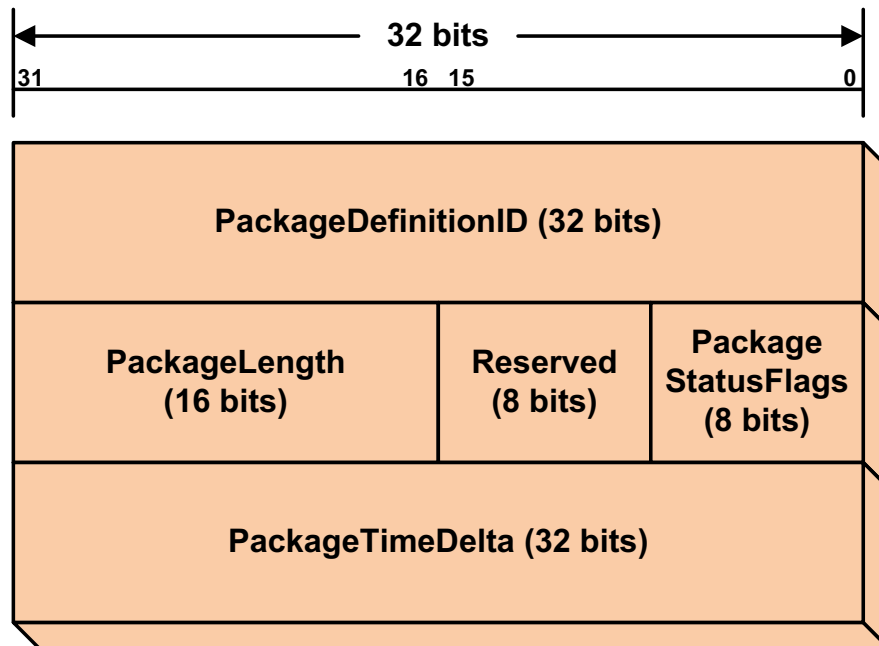


Figure 24-8. Standard *PackageHeader* Field Structure

#### 24.2.2.1.1.1.1 *PackageDefinitionID* Field (32 bits)

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

#### 24.2.2.1.1.1.2 *PackageLength* Field (16 bits)

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.

#### 24.2.2.1.1.1.3 *Reserved* (8 bits)

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

#### 24.2.2.1.1.1.4 *PackageStatusFlags* Field (8 bits)

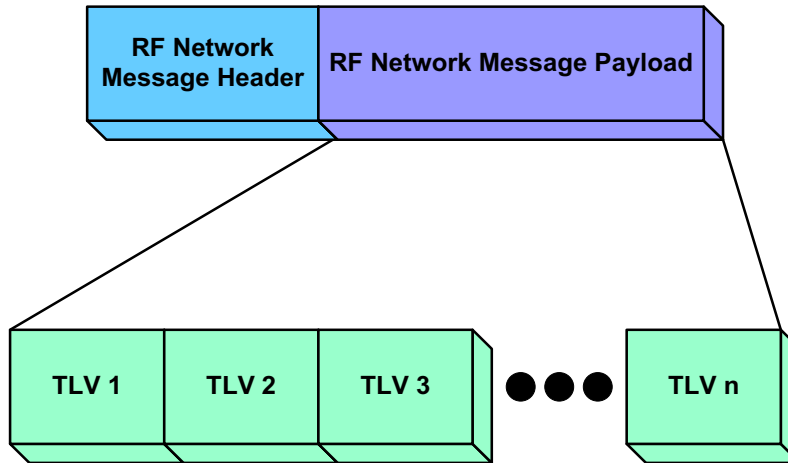
The **PackageStatusFlags** field may provide indications on specific *MeasurementData* in a *Package* and/or error indications (e.g., parity, out of range, wrong frame size, etc.) of the *DataSource* producing the *MeasurementData*. These flags can be described by an *MDL Instance Document*. Each **PackageStatusFlags**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

#### 24.2.2.1.1.1.5 *PackageTimeDelta* Field (32 bits)

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  $2^{32} - 1$ .

### 24.3 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.



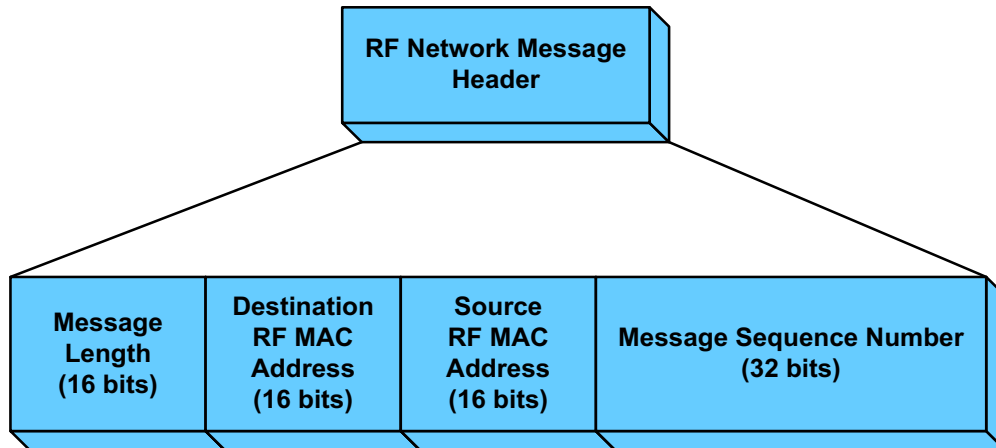
**Figure 24-9. RF Network Message Structure**

All fields in a RF Network Message shall use big-endian ordering as specified in Appendix 21B.

#### 24.3.1 RF Network Message Header Structure

A RF Network Message Header shall contain the following fields shown in Figure 24-10:

- Message Length – 16 bits
- Destination RF 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 (16 bits)**

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. This field is 16 bits long.

#### **24.3.1.2 Destination RF MAC Address (16 bits)**

This field contains the 16-bit 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 with.

#### **24.3.1.3 Source RF MAC Address (16 bits)**

This field contains the 16-bit Source RF MAC Address. The combination of the Source RF MAC Address and the Destination RF MAC Address identify a particular link for which the RF Network message is associated with.

#### **24.3.1.4 Message Sequence Number (32 bits)**

The Message Sequence Number is a 32-bit number that 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 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 TDMA scheduler to allocate a transmission opportunity on a radio link.
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 (MQSR)	3	This TLV is used by a Radio to report MAC layer queue level and, if ARQ is enabled, ARQ queue levels.
Heartbeat	5	This TLV is used to establish a timeout value used by the Radio to classify TxOps received from the LM 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 TE queue levels for each of the 8 TE queues associated with a particular link on the reporting Radio.

**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 kHz (up to 16 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.	$[0, 2^{16}-1]$

Field	Width (bits)	Descriptions	Value/Range
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 has an infinite lifetime and will remain in effect until explicitly deleted or until the transmission heartbeat times out.	[0, 255]
TxOp Start Subseconds	20	The value specifying the fractional sub-seconds portion of a transmission opportunity 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 transmission opportunity 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 TxOp 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 TxOp ID values from TxOps that were supplied to different links on the transceiver. TxOps whose TxOp 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 <sup>16</sup> -1]
...	...	... Optional.	...
TxOp ID N	16	The TxOp ID of the Nth TxOp being acknowledged in this TLV. Optional.	[1, 2 <sup>16</sup> -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, 24]
Reserved	2	Reserved	2'b00
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	$[0, 2^{16}-1]$
ARQ Queue Levels	128	Reserved for ARQ	TBD

**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	11
Timestamp	32	<p>The time that this snapshot of Link Metric information was taken. This timestamp format consists of the following three subfields:</p> <p>Bits 31-26 - Reserved</p> <p>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</p> <p>Bits 19-0 - microseconds The fractional sub-second portion of the timestamp, measured in microseconds.</p>	<p>6'b000000 [0-63]</p> <p>[0-999,999]</p>
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 complement signed integer indicating the RSSI in 1dBm step with a maximum Range of -127dBm to 127dBm. 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 complement signed integer indicating the CINR in 1dB step with a maximum Range of -127dB to 127dB. 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.	$[0, 2^8-1]$

**24.3.2.6 Traffic Engineering (TE) 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 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 TE Queue Status was sampled, measured in microseconds relative to the timestamp Seconds field.	[0, 999,999]
MSLPID	32	Identifier for the Mission SLP associated with this Radio link	[0, 2 <sup>32</sup> -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 DSCP Class 0 (DSCP values 0 to 7)	[0, 2 <sup>16</sup> -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 <sup>16</sup> -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 <sup>16</sup> -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 <sup>16</sup> -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 <sup>16</sup> -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 <sup>16</sup> -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 <sup>16</sup> -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 <sup>16</sup> -1]

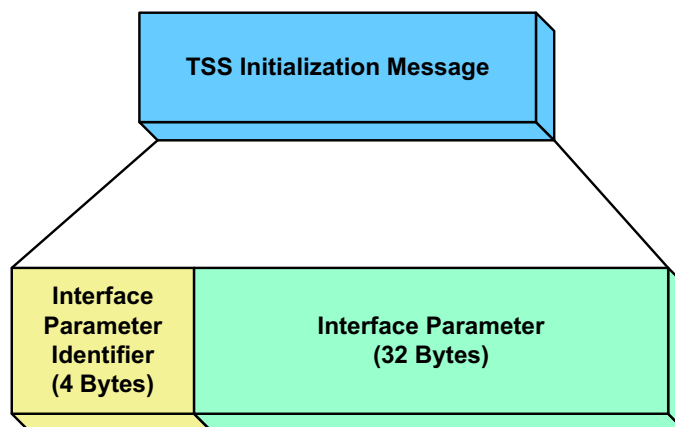
## 24.4 TSS Messages

TSS functionality is described in Chapter 28, but the TSS messages are defined in this section. 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 is shown in Figure 24-11.



**Figure 24-11. TSS Initialization Message Structure**

#### 24.4.1.1 Interface Parameter Identifier (4 Bytes)

The Interface Parameter Identifier field shall contain one of the six values shown in Table 24-9. These values have been chosen to match Linux 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 24-9.

**Table 24-9. TSS Initialization Message Codes**

IOCTL Name	Description	Value
SIOCSIFHWADDR	MAC address of the interface	32'h00008924
SIOCSIFMTU	MTU (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 is a 32-byte container that 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 TmNS network traffic over other networks. The structure of the TSS Data Message is shown in Figure 24-12.

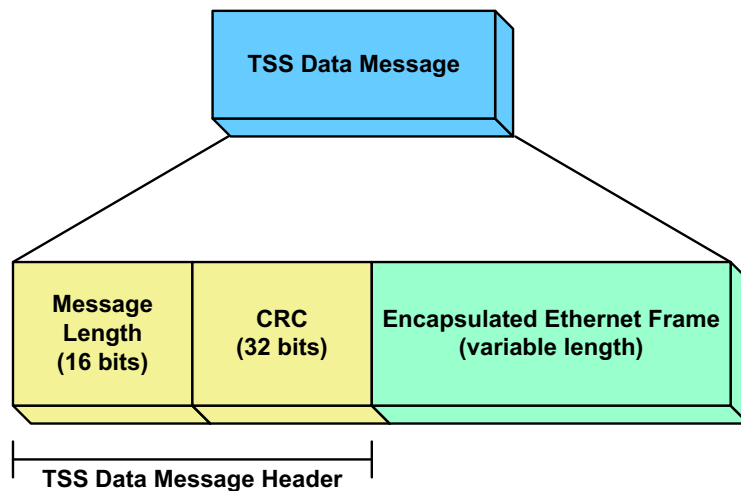


Figure 24-12. TSS Data Message Structure

##### 24.4.2.1 Message Length Field (16 bits)

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. This field is 16 bits long.

##### 24.4.2.2 Cyclic Redundancy Check (CRC) (32 bits)

The CRC field of the TSS Data Message is 32 bits, and it serves as a message identifier for the TSS Data Message. The 32-bit CRC calculation is performed on the entire Encapsulated Ethernet Frame of the message, with the result being stored in this field.

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

The algorithm for the 32-bit CRC calculation shall be equivalent to that shown in Figure 24-13. The constant POLY is defined as the polynomial listed above.

```

/*-----*/
/*----- 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) & 0x0FFFFFFFL;
        t2 = get_crcByte(((int)remainder^(*(data+bytes)))&0xFF);
        remainder = t1^t2;
    }

    for (bytes = 0; bytes < sizeof(remainder); bytes++)
    {
        t1 = (remainder >> 8) & 0x0FFFFFFFL;
        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 at [something.like.rcc.gov/tbd](http://something.like.rcc.gov/tbd).

**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.