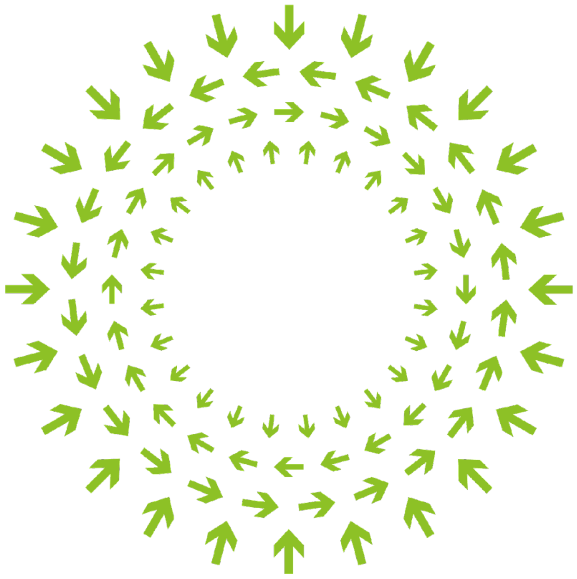


OPEN

Compute Project

NetSync Monitor Protocol

v1.0

09/27/2023

Author: Thomas Behn, Senior Software Engineer, Meinberg
Co-Author: Heiko Gerstung, Managing Director, Meinberg

Table of Contents

1. License **3**
- **1.1. OCP CLA** **3**
- **1.2. Acknowledgements** **4**

2. Compliance with OCP Tenets **5**
- **2.1. Openness** **5**
- **2.2. Efficiency** **5**
- **2.3. Impact** **5**
- **2.4. Scale** **5**
- **2.5. Sustainability** **5**

3. Change Log **6**

4. Overview **7**

5. Communication Flow **8**
- **5.1. Flow Diagram** **8**
- **5.2. Monitoring Request** **9**
- **5.3. Monitoring Response** **10**
- **5.4. Sequence Completion** **14**

6. Backward Compatibility **15**

7. Existing Implementations **18**

Appendix A - Proposal for future additional TLVs **19**

1. License

1.1. OCP CLA

Contributions to this Specification are made under the terms and conditions set forth in Open Compute Project Contribution License Agreement ("OCP CLA") ("Contribution License") by:

Meinberg Funkuhren GmbH & Co. KG

You can review the Contributor License(s) for this Specification on the OCP website at https://www.opencompute.org/legal-documents. For actual executed copies of either agreement, please contact OCP directly.

Usage of this Specification is governed by the terms and conditions set forth in **Modified OWFa1.0.2 Final Specification Agreement (FSA) (As of June 1, 2023)** also known as a "Specification License".

Notes:

1) The above license does not apply to the Appendix or Appendices. The information in the Appendix or Appendices is for reference only and non-normative in nature.

NOTWITHSTANDING THE FOREGOING LICENSES, THIS SPECIFICATION IS PROVIDED BY OCP "AS IS" AND OCP EXPRESSLY DISCLAIMS ANY WARRANTIES (EXPRESS, IMPLIED, OR OTHERWISE), INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, OR TITLE, RELATED TO THE SPECIFICATION. NOTICE IS HEREBY GIVEN, THAT OTHER RIGHTS NOT GRANTED AS SET FORTH ABOVE, INCLUDING WITHOUT LIMITATION, RIGHTS OF THIRD PARTIES WHO DID NOT EXECUTE THE ABOVE LICENSES, MAY BE IMPLICATED BY THE IMPLEMENTATION OF OR COMPLIANCE WITH THIS SPECIFICATION. OCP IS NOT RESPONSIBLE FOR IDENTIFYING RIGHTS FOR WHICH A LICENSE MAY BE REQUIRED IN ORDER TO IMPLEMENT THIS SPECIFICATION. THE ENTIRE RISK AS TO IMPLEMENTING OR OTHERWISE USING THE SPECIFICATION IS ASSUMED BY YOU. IN NO EVENT WILL OCP BE LIABLE TO YOU FOR ANY MONETARY DAMAGES WITH RESPECT TO ANY CLAIMS RELATED TO, OR ARISING OUT OF YOUR USE OF THIS SPECIFICATION, INCLUDING BUT NOT LIMITED TO ANY LIABILITY FOR LOST PROFITS OR ANY CONSEQUENTIAL, INCIDENTAL, INDIRECT, SPECIAL OR PUNITIVE DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS SPECIFICATION, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND EVEN IF OCP HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.2. Acknowledgements

The Contributors of this Specification would like to acknowledge the following companies for their feedback:

Meinberg Funkuhren GmbH & Co. KG
Meta
NVIDIA

2. Compliance with OCP Tenets

2.1. Openness

The spread and success of NetSync Monitor Protocol stick together with the openness of this specification and the possibility of it to be implemented by as many timing product vendors as possible. The more nodes in a sync infrastructure support it, the better the performance and the higher the benefit of the protocol.

2.2. Efficiency

As NetSync Monitor Protocol falls back on well-known PTP messages, it allows for reuse of existing software (source code) as well as hardware (timestampers) and therefore helps reduce costs for R&D.

2.3. Impact

The increasing use of PTP to synchronize critical infrastructure goes hand in hand with the growth in demand to monitor the status and performance of PTP systems within a network. NetSync Monitor Protocol permits to measure and compare path delays and clock offsets of time transmitting and/or receiving devices and therefore helps to detect problems and to make sure that the infrastructure is in sync.

2.4. Scale

The availability of a commonly used protocol for monitoring the timing performance within a network promotes interoperability, collaboration, and innovation, making monitoring systems (software, hardware or both) more versatile and appealing to a wider range of users and developers.

2.5. Sustainability

Monitoring and improving the quality of time within time aware applications tends to improve overall efficiency of distributed applications, driving lower consumption of compute, storage and networking components, and positively impacting DC utilization.

3. Change Log

Date	Version	Author	Description
27SEP2023	v1.0	Thomas Behn	Submission for OCP IC Approval
17AUG2023	v0.1 Rev 7	Thomas Behn	Multiple terminology and spelling mistake fixes
21NOV2019	v0.1 Rev 6	Thomas Behn	Replaced custom NSM TLVs (Reserved TLV IDs had been used) by ORG_EXT_TLVs
06SEP2017	v0.1 Rev 5	Thomas Behn	First publicly shared version

4. Overview

This approach describes a possibility to measure the time offset and monitor the current synchronization status of PTPv2 (IEEE1588) devices in the network by exchanging the standard Delay Request, Delay Response, Sync and Follow Up messages and an additional TLV for the PTP status information of the monitored device.

The concept requires software changes on PTP nodes and introduces a device role called the Monitoring System (MS), which can be an add-on software running on an already existing Grandmaster Clock (GM) or can be implemented as a completely separate device. It is assumed that the monitoring system clock is synchronized to the time used by the Grandmaster Clock(s), e.g. by using its own GNSS receiver (keeping its clock close to the clock of a GNSS-synchronized GM) or by using any other synchronization source that maintains a good-enough time synchronization between the MS and the GMs.

It is possible to deploy multiple MS in a network. The concept currently requires that the MS is configured and maintains a list of all PTP nodes it should monitor. On the nodes to be monitored, a small software change is required that enables the devices to respond to incoming requests from the monitoring node. For security reasons it should be possible to tell the node which hosts are allowed to monitor it. Implementing a secure public/private key scheme would remove the requirement to manually configure the allowed MS on each node and would add protection against ARP/DNS poisoning or Man-in-the-Middle attacks.

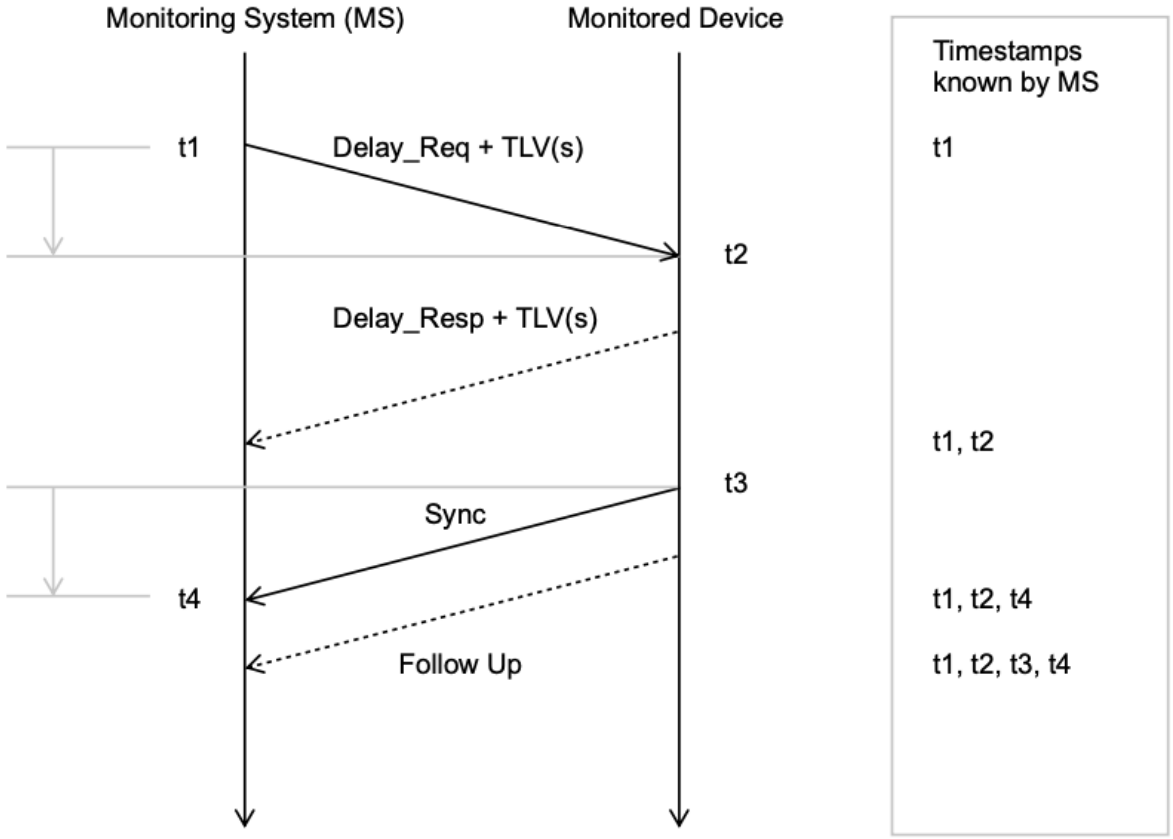
Using standard message formats ensures that the monitoring protocol can make full use of existing hardware timestamping implementations deployed in today's PTP nodes. In addition to measuring the time error of a device, the approach can also determine whether the monitored node has the correct UTC parameters (UTC-TAI offset and leapsecond information for upcoming leapseconds), enabling the monitoring system to audit that a monitored node can make use of legal time within predetermined error bounds, for example to comply with legal requirements like MIFID II.

An additional MTIE TLV, which can simply be requested together with the monitoring TLV by attaching a requesting TLV to the same Delay Request, enables the verification of compliance with specified MTIE time deviation limits.

In addition to monitoring end nodes (timeReceiver-Only Clocks), this concept can also monitor Boundary Clocks, Transparent Clocks and Grandmaster Clocks, if they implement the required software changes.

5. Communication Flow

5.1. Flow Diagram



5.2. Monitoring Request

The monitoring system (MS) sends a unicast Delay Request message to the device to be monitored and stores the egress timestamp for later calculation of offset and delay.
It attaches an organization extension TLV (0x0003) with organizationId 0xEC4670, organizationSubType 0x000001 (PTPMON_REQ_TLV) and no dataField. The lengthField must be set to the size of organizationId and organizationSubType.

Table 1 - PTPMON_REQ_TLV

Bits								Octets	TLV Offset
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**		
tlvType								2	0
lengthField								2	2
organizationId								3	4
organizationSubType								3	7

If needed, it attaches an additional organization extension TLV (0x0003) with organizationId 0xEC4670, organizationSubType 0x000003 (MTIE_REQ_TLV) and no dataField to request MTIE compliance monitoring values. The lengthField must be set to the size of organizationId and organizationSubType.

Table 2 - MTIE_REQ_TLV

Bits								Octets	TLV Offset
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**		
tlvType								2	0
lengthField								2	2
organizationId								3	4
organizationSubType								3	7

5.3. Monitoring Response

The monitored device receives the Delay Request message with the TLV(s) and – regardless of its current port state – responds with a unicast Delay Response message containing the ingress timestamp and the sequence ID of the Delay Request message.

It attaches an organization extension TLV (0x0003) with organizationId 0xEC4670, organizationSubType 0x000002 (PTPMON_RESP_TLV) and the following data containing the PTP status information.

Table 3 – PTPMON_RESP_TLV

Bits								Octets	TLV Offset
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**		
tlvType								2	0
lengthField								2	2
organizationId								3	4
organizationSubType								3	7
portState *(9.2.5*)*								1	10
reserved								1	11
parentPortAddress *(5.3.6*)*								4+N	12
parentDataset *(8.2.3*, 15.5.3.5.1*)*								32	16+N
currentDataset *(8.2.2*, 15.5.3.4.1*)*								18	48+N
timePropertiesDataset *(8.2.4, 15.5.3.6.1*)*								4	66+N
lastSyncTimestamp *(5.3.3*)*								10	70+N

**Appropriate chapter number in IEEE 1588-2008 and IEEE 1588-2019 standard*

tlvType (Enumeration16)
The tlvType shall be ORGANIZATION_EXTENSION (0x0003).

lengthField (UInteger16)
The value of the lengthField is 80+N, where N is an even number representing the length of the parentPortAddress, see 5.3.6.

organizationId (Octet[3])
The value of the organizationId shall be 0xEC4670.

organizationSubType (Enumeration24)
The value of the organizationSubType shall be 0x000002 (PTPMON_RESP_TLV).

portState (Enumeration8)
The value of portState shall be the value of portDs.portState, see 9.2.5.

parentPortAddress (PortAddress, see 5.3.6)
The parentPortAddress shall be the protocol address of the current master.

parentDataset (parentDataset, see 8.2.3)
The parentDataset shall be the value of parentDS containing information about the current master.

currentDataset (currentDataset, see 8.2.2)
The currentDataset shall be the value of currentDS containing information about the current synchronization state (stepsRemoved, offsetFromMaster, meanPathDelay).

timePropertiesDataset (timePropertiesDS, see 8.2.4)
The timePropertiesDataset shall be the value of timePropertiesDS containing general information about the clocks time (currentUtcOffset, ptpTimescale, timeSource, …).

lastSyncTimestamp (Timestamp, see 5.3.3)
The lastSyncTimestamp shall be the timestamp of the last successful synchronization.

Additionally, if an MTIE_REQ_TLV had been attached by the monitoring system, an organization extension TLV (0x0003) with organizationId 0xEC4670, organizationSubType 0x000004 (MTIE_RESP_TLV) and the following data shall be added to the Delay Reponse.

Table 4 – MTIE_RESP_TLV

Bits								Octets	TLV Offset
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**		
tlvType								2	0
lengthField								2	2
organizationId								3	4

organizationSubType	3	7
mtieValid	1	10
reserved	1	11
mtieWindowNumber	2	12
mtieWindowDuration	2	14
minOffsFromMaster *(5.3.2*)*	8	16
minOffsFromMasterAt *(5.3.3*)*	10	24
maxOffsFromMaster *(5.3.2*)*	8	34
maxOffsFromMasterAt *(5.3.3*)*	10	42

**Appropriate chapter number in IEEE 1588-2008 and IEEE 1588-2019 standard*

tlvType (Enumeration16)
The tlvType shall be ORGANIZATION_EXTENSION (0x0003).

lengthField (UInteger16)
The value of the lengthField is 52.

organizationId (Octet[3])
The value of the organizationId shall be 0xEC4670.

organizationSubType (Enumeration24)
The value of the organizationSubType shall be 0x000004 (MTIE_RESP_TLV).

mtieValid (UInteger8)
The mtieValid field indicates whether the MTIE values in this TLV have been calculated from at least one whole MTIE measurement window. The value is FALSE until the PTP port has transitioned into SLAVE state and collected data for at least *mtieWindowDuration* seconds. The value also becomes FALSE when statistics are cleared (implementation-specific).

mtieWindowNumber (UInteger16)
Rolling counter, incrementing every time the MTIE data is updated (every *mtieWindowDuration* seconds), to indicate that new data is available, when querying more often than *mtieWindowDuration* seconds.

mtieWindowDuration (UInteger16)
MTIE data collection period between updates, in seconds.

minOffsFromMaster (TimeInterval, see 5.3.2)
Minimum offset from master value within the last measurement window.

minOffsFromMasterAt (Timestamp, see 5.3.3)
Timestamp of the occurrence of the minimum offset from master, expressed in the timescale of the PTP port (timePropertiesDS dataset). The PTP T2 timestamp may be used; otherwise timestamp accuracy shall be within one PTP Sync interval.

maxOffsFromMaster (TimeInterval, see 5.3.2)
Maximum offset from master value within the last measurement window.

maxOffsFromMasterAt (Timestamp, see 5.3.3)
Timestamp of the occurrence of maximum offset from master, expressed in the timescale of the PTP port (timePropertiesDS dataset). The PTP T2 timestamp may be used; otherwise timestamp accuracy shall be within one PTP Sync interval.

In addition to the Delay Response, the device sends a unicast Sync message and – in two-step operation – a Follow Up message with the sequence ID of the Delay Request message.

5.4. Sequence Completion

The monitoring device receives the Delay Response message, the Sync (and Follow Up) message, takes the appropriate timestamps and calculates the Slave-to-Master (s2m) and Master-to-Slave (m2s) delays. In addition to that, it stores the information from the PTPMON_RESP_TLV (and, if requested, of the MTIE_RESP_TLV) so that it can be evaluated in the monitoring routines.

One big advantage of this mechanism is that the communicated offset and delay from the monitored device can be compared to the measured offset and delay values and – assuming that the monitoring system is perfectly synchronized – asymmetries can be detected.

The only required change on existing PTP implementations is, that incoming Delay Request messages with an attached PTPMON_REQ_TLV (and MTIE_REQ_TLV), are answered with a Delay Response message with PTPMON_RESP_TLV (and MTIE_RESP_TLV) and a Sync (and Follow Up) message. This must be done regardless of the current mode and port state of the monitored device.

6. Backward Compatibility

In order to allow backward compatibility to implementations of previous revisions of this specification (before OCP contribution), the following fallback mechanism shall be implemented:

If one of the following request TLVs had been attached by the monitoring system, the appropriate legacy response TLV shall be added to the Delay Response instead of the organization extensions TLVs described above.

Table 5 – PTPMON_REQ_TLV_OLD (< v0.1 Rev 6)

Bits								Octets	TLV Offset
7	6	5	4	3	2	1	0		
tlvType (0x21FE)								2	0
lengthField (0)								2	2

Table 6 – MTIE_REQ_TLV_OLD (< v0.1 Rev 6)

Bits								Octets	TLV Offset
7	6	5	4	3	2	1	0		
tlvType (0x2200)								2	0
lengthField (0)								2	2

Table 7 – PTPMON_RESP_TLV_OLD (< v0.1 Rev 6)

Bits								Octets	TLV Offset
7	6	5	4	3	2	1	0		
tlvType (0x21FF)								2	0
lengthField (74 + N)								2	2
portState *(9.2.5*)*								1	4
reserved								1	5

parentPortAddress *(5.3.6*)*	4+N	6
parentDataset *(8.2.3*, 15.5.3.5.1*)*	32	10+N
currentDataset *(8.2.2*, 15.5.3.4.1*)*	18	42+N
timePropertiesDataset *(8.2.4, 15.5.3.6.1*)*	4	60+N
lastSyncTimestamp *(5.3.3*)*	10	64+N

**Appropriate chapter number in IEEE 1588-2008 and IEEE 1588-2019 standard*

Table 8 – MTIE_RESP_TLV_OLD (< v0.1 Rev 6)

Bits								Octets	TLV Offset
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**		
tlvType (0x2201)								2	0
lengthField (46)								2	2
mtieValid								1	4
reserved								1	5
mtieWindowNumber								2	6
mtieWindowDuration								2	8
minOffsFromMaster *(5.3.2*)*								8	10
minOffsFromMasterAt *(5.3.3*)*								10	18
maxOffsFromMaster *(5.3.2*)*								8	28
maxOffsFromMasterAt *(5.3.3*)*								10	36

**Appropriate chapter number in IEEE 1588-2008 and IEEE 1588-2019 standard*

The following table shows the TLV type values for the previous revisions and their counterpart since Revision 6.

Table 9 – TLV Type comparison

TLV	**>= v0.1 Rev 6**	**<= v0.1 Rev 5**
PTPMON_REQ_TLV	0x0003, 0xEC4670, 0x000001	0x21FE
PTPMON_RESP_TLV	0x0003, 0xEC4670, 0x000002	0x21FF
MTIE_REQ_TLV	0x0003, 0xEC4670, 0x000003	0x2200
MTIE_RESP_TLV	0x0003, 0xEC4670, 0x000004	0x2201

The data of the old response TLVs (Table 7 and Table 8) shall be filled as described for the appropriate newer TLV versions (Table 3 and Table 4).

7. Existing Implementations

In order to add support for v0.1 Rev 6 and later, an existing implementation needs to apply the following changes:

Incoming messages:
For each TLV, in addition to checking for the TLV types used by v0.1 Rev 5 and earlier, the implementation needs to check if the TLV is an organization extension TLV and if it has the correct organizationId and one of the supported organizationSubTypes.

Outgoing messages:
The response TLVs have to be set up using the old TLV types in case the request TLV was using the old types, too. An organization extension request TLV according to v0.1 Rev 6 and later shall be answered using the corresponding organization extension response TLVs.

A monitoring system shall always try to use the new v0.1 Rev 6+ request TLVs first and only if it does not receive a response, try the old ones instead. If it receives a response when using the old TLVs, but not when it uses the new TLVs, it shall send any further requests to that specific monitored node using the old TLVs. Whenever there is a restart of the monitoring system or a reconfiguration concerning the monitored node on the monitoring system, it should retry to use the new mechanism again to check if the monitored node has been updated to support v0.1 Rev 6+ in the meantime.

Appendix A - Proposal for future additional TLVs

In order to enable the monitoring system to determine possible disparities between the PTP clock of a slave and other clocks (i.e. the system clock of the operating system), a third requesting TLV can be attached to the Delay Request, signaling, that the monitored slave shall attach an appropriate response TLV containing the offset to the PTP clock for each additional clock it supports.

This TLV shall be an organization extension TLV (0x0003) with organizationId 0xEC4670, organizationSubType 0x000005 (CLOCK_OFFS_REQ_TLV) and no dataField. The lengthField must be set to the size of organizationId and organizationSubType.

Table 10 - CLOCK_OFFS_REQ_TLV

Bits								Octets	TLV Offset
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**		
tlvType								2	0
lengthField								2	2
organizationId								3	4
organizationSubType								3	7

If a CLOCK_OFFS_REQ_TLV has been found in the Delay Request, the monitored device shall add an organization extension TLV (0x0003) with organizationId 0xEC4670, organizationSubType 0x000006 (CLOCK_OFFS_RESP_TLV) and the following data for each supported clock.

Table 11 – CLOCK_OFFS_RESP_TLV

Bits								Octets	TLV Offset
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**		
tlvType								2	0
lengthField								2	2
organizationId								3	4
organizationSubType								3	7

clock	1	10
clockStatus	1	11
refClock	1	12
reserved	1	13
clockOffs *(5.3.2*)*	8	14
reserved	3	22
clockDesc *(5.3.9*)*	1+N	25

**Appropriate chapter number in IEEE 1588-2008 and IEEE 1588-2019 standard*

tlvType (Enumeration16)
The tlvType shall be ORGANIZATION_EXTENSION (0x0003).

lengthField (UInteger16)
The value of the lengthField is 26 + N, where N is an even number representing the length of the clockDesc, see 5.3.9.

organizationId (Octet[3])
The value of the organizationId shall be 0xEC4670.

organizationSubType (Enumeration24)
The value of the organizationSubType shall be 0x000006 (CLOCK_OFFS_RESP_TLV).

clock (Enumeration8)
The value of clock shall be a unique ID for each supported clock, see table 12.

clockStatus (Enumeration8)
The value of clockStatus shall be the current synchronization status of the clock, see table 13.

refClock (Enumeration8)
The value of refClock shall be the unique ID of the reference clock of this clock, see table 12.

clockOffs (TimeInterval, see 5.3.2)
The value of clockOffs shall be the offset between this clock and its reference clock.

clockDesc (PTPText, see 5.3.9)
The value of clockDesc shall be an optional description of the clock. The number of bytes used (N) shall be an even number.

Table 12 – Clock enumeration

Name	Value
PTPClock	0
OperatingSystemClock	1
Reserved for specified clocks	2 - 63
User defined clocks	64 - 255

Table 13 – clockStatus enumeration

Name	Value
Locked	0
Synchronising	1
Holdover	2
Freerun	3
Reserved for specified clock stati	4 - 15
User defined clock stati	16 - 252
Faulty	253
Disabled	254
Unknown clock status	255