IEEE Standards Interpretations for IEEE Std 1588™-2008 IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems

This is an interpretation of IEEE Std 1588-2008.

Interpretations are issued to explain and clarify the intent of a standard and **do not** constitute an alteration to the original standard. In addition, interpretations are not intended to supply consulting information. Permission is hereby granted to download and print one copy of this document. Individuals seeking permission to reproduce and/or distribute this document in its entirety or portions of this document must contact the IEEE Standards Department for the appropriate license. Use of the information contained in this document is at your own risk.

IEEE Standards Department Copyrights and Permissions 445 Hoes Lane, Piscataway, New Jersey 08855-1331, USA

December 2011

Interpretation Request #1
Topic: Figure 4 -- Transport Protocol **Subclause:** 6.5.4

In the text just below Figure 4 the standard states: "The end-to-end transparent clock forwards all messages just as a normal bridge, router, or repeater. However for PTP event messages, the residence time bridge, shown in Figure 4, measures the residence time of PTP event messages (the time the message takes to traverse the transparent clock). These residence times are accumulated in a special field, the correctionField, of the PTP event message or the associated follow up message (Follow_Up or Pdelay_Resp_Follow_Up). This correction is based on the difference in the timestamp generated when the event message enters and leaves the transparent clock. Any updates to checksums required by the network protocol are made.

The grandmaster sends a sync Ethernet frame to the transparent clock with the Ethernet header source MAC address fields as master's source MAC address (of course). In switch like ours, there is a CPU (and an Ethernet port) within the switch handles PTP messages, and the CPU has its own MAC address. Because TC has to alter the content of the Ethernet frame such as changing correction field and checksums, it essentially terminates the original sync Ethernet frame then generate a new one to send out, so the outgoing sync Ethernet frame will have switch's MAC address as the source MAC address instead of the grandmaster's source MAC address.

From the excerpt of spec above, it pretty much says like "The end-to-end transparent

clock forwards all messages just as a normal bridge, router, or repeater, except changing correction field and checksum". A normal bridge or router will not alter an Ethernet frame's source MAC address. So my question is: should the transparent clock keep the original grandmaster's source MAC address, or not?

Interpretation Response #1
PTP does not attempt to change the behavior of the transport protocol. Clause 10 defines the processing behavior of transparent clocks and explicitly states the PTP fields that need to be modified by transparent clocks, but is silent about modifications of the source protocol address.

PTP Annex K defines an experimental security protocol for PTP. Annex K uses the source protocol address as part of the attributes of the security association formed between sender and receiver clocks. If the source protocol address is modified by a transparent clock the security association lookup described in K.7 fails and the received PTP message would be silently discarded. Annex K K.14.6 describes the processing rules for secure transparent clocks.

PTP supports a unicast communication model assuming that the behavior of the protocol is preserved. Annex A.9.2 describes the ramifications of a unicast model on boundary and transparent clocks. In particular it addresses the issue of formation of the synchronization hierarchy. It suggests that one way to achieve the correct hierarchy is by configuration each port in advance with the unicast protocol addresses of the neighboring clocks visible from every port. However in some scenarios it is desirable to automate the discovery of neighboring clocks. For example, if only master ports are configured with addresses of the neighboring ports, slave-only clocks could potentially learn the protocol address of the master clock from the source protocol address of Sync messages. If the source protocol address is modified by transparent clocks this automatic learning process would lead to error.

We also note that implementations of transparent clocks exist that use unmanaged switch technology for which there is no appropriate source address.

In summary, PTP does not mandate that transparent clock must not override the source protocol address of PTP messages. If the source protocol address is modified, the experimental PTP security extension can not be used, and automatic discovery as detailed above or other similar features (outside the scope of PTP) that assume that transparent clocks are 'transparent' with regards to the protocol addresses must be implemented with care.

Writers of PTP profiles are encouraged to highlight any 'non-transparent' modifications of the transport layer fields performed by the transparent clocks designed to the profile.

Interpretation Request #2
Topic: clockIdentities and portIdentities **Subclause:** 7.5.2.4

The last part of 7.5.2.4 reads: "A portIdentity A of type PortIdentity with attributes clockIdentity and portNumber and a clockIdentity B of type ClockIdentity are compared as follows:

a. If A.clockIdentity is less than B.clockIdentity, then A < B. b. Otherwise, if A.clock-Identity is greater than B.clockIdentity, then A > B. <A."

Question: Should item c) state "Otherwise, A = B", which is consistent with the second sentence of Para 7.5.2.4?

Interpretation Response #2
No--IEEE Std 1588-2008 is correct as written. Note that 3 different comparison cases are defined in 7.5.2.4 to cover the possible cases involved in executing the best master clock algorithm on multiport devices. The first case is comparing two clockIdentities. The second case is comparing two portIdentities. The third case--and the subject of the question--is comparing a portIdentity with a clockIdentity. In this case the clockIdentity does NOT have a portNumber field and for purposes of this comparison a zero value for portNumber is assumed (since this will be the case for the usage of this third option). With this assumption the result is B<A as stated.

Interpretation Request #3
Topic: announceReceiptTimeout Default Value **Subclause:** 7.7.3.1

The value of portDS.announceReceiptTimeout shall specify the number of announceInterval that has to pass without receipt of an Announce message before the occurrence of the event ANNOUNCE_RECEIPT_TIMEOUT_EXPIRES; see 9.2.6.11. The range shall be 2 to 255 subject to further restrictions of a PTP profile. The minimum value should be 3.

Question: Should the last sentence state "The default value should be 3.", since the previous sentence states that the Announce Receipt Timeout range is 2 to 255?

Interpretation Response #3
IEEE Std 1588-2008 is correct. It states a recommended (hence 'should') minimum value of 3. This is a configurable attribute and the default value is left to be defined in a profile. The profiles in Annex J define the default value to be 3. Other profiles may select a different value provided it is in the range of 2 to 255. A value of 2 is permitted but is not recommended except in carefully controlled environments where more rapid reaction to missed Announce messages is needed and the number of missed messages is very low.

Interpretation Request #4
Topic: Initialization Value -- defaultDS.priority2 member **Subclauses:** 8.2.3.8; and 8.2.3.9

8.2.3.8 parentDS.grandmasterPriority1

- The value of parentDS.grandmasterPriority1 is the priority1 attribute (see 7.6.2.2) of the grandmaster clock. … The initialization value shall be the value of the defaultDS. priority1 member.

8.2.3.9 parentDS.grandmasterPriority2

- The value of parentDS.grandmasterPriority2 is the priority2 attribute (see 7.6.2.3) of the grandmaster clock… The initialization value shall be the value of the parentDS. priority2 member.

Question: Should the last sentence of Para 8.2.3.9 state "The initialization value shall be the value of the defaultDS.priority2 member." so it is consistent with Para 8.2.3.8?

Interpretation Response #4
This is clearly a typographical error and the correct statement of 8.2.3.9 should be "The initialization value shall be the value of the defaultDS.priority2 member.

Interpretation Request #5
Topic: logMessageInterval field of an announce message **Subclause:** 13.3.2.11

Subclause 13.3.2.11 says that the logMessageInterval field of an announce message must be the value of portDS.logAnnounceInterval. portDS.logAnnounceInterval is the multicast announce period. This seems odd when the message is a unicast announce message which may have been negotiated to be transmitted with a different period. Is my reading correct?

Interpretation Response #5
Subclause 13.3.2.11 says that the logMessageInterval field of an announce message must be the value of portDS.logAnnounceInterval. PortDS.logAnnounceInterval is the log (base 2) of the multicast announce period. It is possible that unicast announce messages are transmitted with a different period; however the text of the standard says that the value of the logMessageInterval field of those (unicast) messages shall equal portDS. logAnnounceInterval, the log of the multicast announce period.

Interpretation Request #6
Topic: PTP-defined static, dynamic, and configurable attributes **Subclause:** 15.5.1.1.1

The second paragraph of 15.5.1.1.1 states that "PTP-defined static, dynamic, and configurable attributes" may not be used with the SET actionField value. However, the previous paragraph states that configurable attributes may indeed be changed with the SET actionField. I suggest that the second paragraph should instead read "PTP-defined static and dynamic attributes..." Is this correct?

Interpretation Response #6
IEEE Std 1588-2008 appears to be in error as noted. The correction suggested is "PTP-defined static, and unless otherwise noted in the standard, dynamic attributes." The caveat is needed since there are some attributes that are configurable or dynamic depending on the implementation, for example those in the time properties data set depending on whether the attributes are designed to be set automatically via a GPS link, i.e. dynamic, or by management messages, i.e. configurable.

Interpretation Request #7
Topic: Version 2 clock response to NULL management message **Subclause:** 15.5.3.1.1

Are version two clocks required to respond to a null management message? In my opinion, if the actionField is COMMAND, the answer is yes and ACKNOWLEDGE message should be returned. It is a bit ambiguous for the case where the actionField is SET or GET because there is no data involved.

Interpretation Response #7
NULL management messages can have a legal value of actionField of GET, SET, or COMMAND. Hence the rules of table 38 apply. In the case of GET and SET, the structure of the management message has a lengthField of 0 indicating that the dataField is of zero length. Therefore, the TLV returned in response to a NULL with GET or SET should be this same TLV with lengthField of 0.

Interpretation Request #8
Topic: Clock Description **Subclause:** 15.5.3.1.2

Subclause 15.5.3.1.2.1 establishes the clockType field as an array of 16 Boolean values. Values at indices 0-4 represent the presence or absence of various clock types in a node, and the remaining values are reserved. Per 5.4.3, arrays of primitive types (such as Boolean) are "formatted with the member having the lowest numerical index closest to the start of the protocol data unit." This seems to indicate that the 5 bits closest to the front of the PDU would be those used for clockType data, and that the remainder would be reserved. I have encountered multiple implementations, however, that use the 5 bits farthest from the start of the PDU as the data bits, leaving the 11 bits closest to the front of the PDU reserved. Which interpretation of 15.5.3.1.2.1 is correct?

Interpretation Response #8
The data type of the clockType field is Boolean[16]. The bit corresponding to array index 0, which from Table 42 indicates whether the clock is an ordinary clock, occupies bit position 7 of the first octet in the CLOCK_DESCRIPTION management TLV data field. The remaining bits defined in Table 42 occupy bit positions 6, 5, 4, and 3 corresponding to the descriptions for boundary clock, peer-to-peer transparent clock, end-to-end transparent clock, and management node respectively. The reserved indices from Table 42 correspond to bit positions 2, 1, and 0 in the first octet and bit positions 7 through 0 in the second octet.

To illustrate, for an ordinary clock, which from Table 42 corresponds to Boolean array index 0, a portion of Table 41 is reproduced below with the clockType field (the first two octets) expanded to show the position of the bit indicating that the device is an ordinary clock.

- Table 41¾CLOCK_DESCRIPTION management TLV data field

Bits								Octets	TLV data offset
7	6	5	4	3	2	1	0		
1	0	0	0	0	0	0	0	2	0
0	0	0	0	0	0	0	0		1
physicalLayerProtocol								L	2
-remainder of fields not shown									

It is recommended that in the next revision of IEEE Std 1588, that the definition of clockType be rewritten, without changing the positions or meanings of the bits, in the same style as used in defining the flagField, see 13.3.2.6.

Interpretation Request #9
Topic: displayData field **Subclause:** 15.5.4.1.6

Subclause 15.5.4.1.6 states that the displayData field "is an optional text field." Does this mean that the displayData field can be elided entirely, or that the lengthField of the PTPText field would be set to 0 and only the textField portion would be absent? The former would result in a protocol data unit that ends at the end of the "reserved" field. The latter would have an additional two octets: a 0 for the displayData lengthField and an additional 0 to pad the PDU length to an even number of octets.

Interpretation Response #9
This is an optional field. When the sender does not wish to send displayData, the entire displayData field should be deleted from the TLV and the lengthField adjusted accordingly. In this case, the pad field is not required, i.e. has zero length. However, the transmission of the displayData field containing no text is also acceptable.

Interpretation Request #10
Topic: Hardware compatibility bit **Clause:** D.4

It seems the timeout is too short for the case of a slave sending delay requests at a rate less than announceReceiptTimeout seconds. Also it is not clear whether the behavior is the same for a master and a slave.

We implemented the timeout, but we use a timeout of 60 seconds, which is about twice

the maximum allowed delay request rate (32s). We also implemented an option to disable or enable padding without a timeout, so we can set the padding to "on", "off" or "auto". There is a problem in the spec! For robustness at least twice the worst case delay request interval would be a better choice. Longer timeouts should do no harm, because all nodes have to handle padded packets anyway.

Table D.2 states: "This padding shall be added to all transmitted PTP event messages to the receiving node, for a time duration equal to the value of portDS.announceReceiptTimeout seconds since the last PTP Announce or event message received from that node with bit 0 equal to '1.'" However, portDS.announceReceiptTimeout is not measured in seconds. Rather, it is a number of announceIntervals. Question: Should this duration instead be equal to the announceReceiptTimeoutInterval (that is, portDS.announceReceiptTimeout x 2^portDS.logAnnounceInterval), or is the value of announceReceiptTimeout truly intended to be used as a seconds duration in this instance? If the former, a profile that allows a negative value for logAnnounceInterval could result in a non-integral announceInterval (and thus potentially a non-integral announceReceiptTimeoutInterval). In such a circumstance, should the time duration be rounded up to the next integral number of seconds? An answer to this question may also apply to 9.2.6.10.

Interpretation Response #10
The timeout mechanism as described in IEEE Std 1588-2008 does not work properly and is poorly stated. As noted, it is unclear how the stated timeouts apply in the case of a master and a slave. Secondly, the timeout mechanism, if it worked, should be the interval computed based on the definition in 8.2.5.4.2. The committee recommends that in the interim, the timeout mechanism should not be implemented. Rather, when padding has been requested, padding should remain in effect until the next initialization state.

Interpretation Request #11
Topic: twoStepFlag for Follow_Up messages **Subclause:** 11.5.2.2 and 13.3.2.6

Subclause 11.5.2.2 b)6) says that a two-step TC that forwards a one-step sync-message shall generate a follow-up message, copying the flag-field to the follow-up message but with two-step-flag set to TRUE. Subclause 13.3.2.6 says that the two-step flag shall be set to TRUE in Sync- and Pdelay_Resp messages for a two-step clock, and all other messages have this flag set to FALSE. These subclauses contradict each other. Please clarify if and why 11.5.2.2 b)6) is right or wrong.

Interpretation Response #11
The error is in 11.5.2.2 b)6) which should read, "The header flagField of the received Sync message shall be copied into the flagField of the Follow_Up message but with the twoStepFlag set to FALSE." This brings this statement into agreement with Table 20 of 13.3.2.6.

Note that 11.5.2.2 a) requires that the twoStepFlag of the corresponding Sync message issued by the transparent clock to be set to TRUE indicating that a Follow_Up message

follows consistent with Table 20 of 13.3.2.6.

Interpretation Request #12
Topic: Semantics of the fault log read and reset **Subclause:** 15.5.3.1.7

Should all fault records be deleted automatically after they have been read with a FAULT_LOG management message, or should they be kept until a FAULT_LOG_RESET management message is received? IEEE Std 1588-2008 says nothing about that and it is not clear whether both interpretations are legal. There are some advantages and disadvantages for both behaviors. a) If the fault records are not removed from the fault log after having been read: Advantages: • More the one "Client" can read the fault log simultaneously • If a FAULT_LOG message response gets lost in the network, then the "Client" can try again.

Disadvantages: • If I want to delete all old fault records with a FAULT_LOG_RESET message after I have read it with a FAULT_LOG message, I lose all log records that were produced between the two messages. • If I don't reset the fault log, it grows to the maximum size. Every time the fault log is read, a maximum size FAULT_LOG message is generated, most probably containing a lot of old fault records that have already been read. This generates unnecessary network traffic and CPU load which may be a problem for weak embedded devices. • The "Clien" has to determine which fault records are new and which are not, if it does not send a FAULT_LOG_RESET messages. b) If the fault records are deleted automatically after having been read: Advantages: • The "Client" gets new fault records only. • The FAULT_LOG message is only as big as necessary. This saves network capacity. Disadvantages: • If a FAULT_LOG message response gets lost in the network, the contained fault records are lost. • Only one "Client" can read the fault log at the same time. • The FAULT_LOG_RESET message seems to be useless. [Does this imply that interpretation a) is correct?] What behavior do you expect?

Interpretation Response #12
The behavior of the FAULT_LOG_TLV with GET semantics is that it returns the fault log record in the format specified by Table 47 of 15.5.3.1.7. No statement is made concerning any deletion associated with this TLV, therefore there is no deletion associated with the GET.

The behavior of the FAULT_LOG_RESET TLV with COMMAND semantics is that it clears all entries in the fault log as specified in 15.5.3.1.8.

If more complex semantics is required for an application domain, the alternative is to create an organization specific TLV with the required semantics as described in 14.3.

It is suggested that this issue be raised at the next revision of IEEE Std 1588-2008 to address the addition of additional semantics and any issues surrounding fault log records recorded after the last read but before a reset.

Interpretation Request #13
Topic: Unicast Discovery **Subclause:** 17.5.1

Subclause 17.5.1 makes explicit references to "slave" and "slave port". Would you please clarify whether this restricts the use of Unicast Discovery to the port on an ordinary clock that has defaultDS.slaveOnly set to TRUE? Or was the intent that any port configured with a UNICAST_MASTER_TABLE can use these procedures (e.g. a port on a Boundary Clock with defaultDS.slaveOnly = FALSE).

Interpretation Response #13
There are no restrictions on the use of the unicast discovery mechanism of 17.5.1 based on the value of the defaultDS.slaveOnly attribute. Any port may use this mechanism to help determine its future state and potential masters.

Interpretation Request #14
Topic: Unicast message negotiation with port in SLAVE state **Subclause:** 16.1.1

Subclause 16.1.1. states "When the grant is of Announce or Sync messages, the grantor shall transmit the messages with a mean inter-message period approximately equal to the granted inter message period." If a boundary clock (or a non-slaveOnly ordinary clock) receives a request to transmit announce and sync messages, on a port with the portDS.portState equal to SLAVE should it: a) Ignore the request and send no response (this would seem to contradict paragraph 2 of 16.1.1) b) tx grant with non-zero duration, and not transmit announce/sync messages until it transitions to MASTER c) tx grant with 0 duration d) tx grant with non-zero duration for only the announce messages, and 0 duration for the Sync messages

Interpretation Response #14
Subclause 9.2.4 specifically allows exceptions to the restrictions of Table 10 for the provisions of 16.1. There is no requirement in the standard for a port receiving such a request to grant the request. In particular there is no restriction on granting or denying a request based on state in the standard. If a port wishes to deny the request there are two acceptable ways to communicate this denial: indicate in the granting message (see 16.1.4.2) that the durationField value is 0, or if the port does not support or recognize the REQUEST_UNICAST_TRANSMISSION TLV to return the appropriate error TLV per 15.5.4. Requesting ports must account for the fact that they may be receiving multicast Announce or Sync messages from the requested or other ports and that the granting port may be in the SLAVE state. Note that there are management messages that enable a port to learn about a port granting a request, e.g. learn about its state. The standard itself specifies two uses of this mechanism: The master cluster table of 17.3 and the unicast discovery of 17.5. Other uses will no doubt be devised.

Interpretation Request #15
Topic: Unicast message negotiation with port in PASSIVE state **Subclause:** 16.1.1

As for the question above but when the portDS.portState equal to PASSIVE, should it a) Ignore the request and send no response, b) tx grant with non-zero duration, and not transmit announce/sync messages until it transitions to MASTER c) tx grant with 0 duration d) tx grant with non-zero duration for only the announce messages, and 0 duration for the Sync messages

Interpretation Response #15
The answer is the same as for the previous question.

Interpretation Request #16
Topic: Unicast message negotiation datasets **Subclause:** 16.1.1

When a port has granted a Unicast session, there is state information that must be maintained (for example grantee address, message intervals). Will future versions of the specification define these as a data set and will these datasets be retrievable through the PTP management mechanism (clause 15)?

Interpretation Response #16
We cannot predict what future revision will or will not include. However this is a topic that we recommend for discussion at the next revision.

Interpretation Request #17
Topic: Unicast message transmission rate **Subclause:** 7.7.2.3 and 8.2.5.4.3

Subclause 7.7.2.3 and 8.2.5.4.3 indicate that logSyncInterval is for multicast sync messages. What data set parameter is used to configure the rate of sync messages for unicast operation? What procedures within the specification apply to using this parameter? Would the ranges and default value for this parameter be included in a profile?

Interpretation Response #17
The data set parameters used for unicast operation are implementation specific. Any procedure in the standard that depends on the timing of Sync messages are affected notably the issuing timing of Sync and Follow_Up messages. Whether unicast parameters are included in a profile is up to the organization writing the profile. However if it is critical to an application it is a good idea.

Interpretation Request #18
Topic: Alternate masters in a unicast environment **Subclause:** 17.4.2

Subclause 17.4.2 states "A port shall transmit multicast Sync, and, if a two-step clock, Follow_Up messages subject to the restrictions in Table 87. A port transmitting Sync or Follow_Up message under the terms of 17.4 shall set alternateMasterFlag to TRUE. These messages shall be transmitted at the interval defined by ‹logAlternateMulticastSyncInterval› in Table 87." Does the terminology "transmit multicast Sync" prohibit the use of alternate master capabilities in a unicast-only environment?

Interpretation Response #18
Subclause 7.3.1 gives permission to emulate the multicast behaviour using unicast, therefore this is allowed to be used for unicast.

Interpretation Request #19
Topic: Best master clock algorithm **Subclause:** 9.3.2.3

Subclause 9.3.2.3 states "c) If port 'r' is in the SLAVE state, include the results of the previous computation of Erbest on port 'r.' However, if there is a more recent qualified Announce message received on port 'r' from the same sending port, the values from that message shall be considered instead. If an ANNOUNCE_RECEIPT_TIMEOUT_EXPIRES event occurs, see 9.2.6.11, for the clock selected during the previous computation of Erbest on port 'r,' then the previous computation of Erbest on port 'r' shall not be included." Why is the SLAVE state called out specifically in this text, does this not also apply to ports in MASTER and PASSIVE states?

Interpretation Response #19
Item 'c' referenced in the question currently applies only to the SLAVE state. Its function is to prevent thrashing due to a missed Announce message transmission and the execution of the best master clock algorithm. Inclusion of the prior computation ensures that a missed Announce message or one that came in slightly after the start of the computation due to timing uncertainty will not cause a state change. The Announce timeout is designed to handle the case where the slave ceases to receive Announce messages from its master. There are two circumstances where an Announce message may be missed with respect to the execution of the best master clock algorithm. The first is if some network failure has actually caused one of the Announce messages to be dropped or corrupted. The second is if minor differences in the time of receipt of Announce messages compared to the occurrence of state change events, see 9.2.6.8 that causes the best master clock algorithm to be executed. 9.2.6.8 specifies that a state change event occur at least once per Announce interval (which will obviously be implemented based on local clocks or timers). However if the sender of the Announce message uses a slightly longer interval, either because of differences in the clocks or timers or due to the permitted uncertainty in the actual transmission interval allowed by 9.5.8, then it is possible to have a state change interval without a receipt of an Announce message from a specific clock. This 'missed' Announce message is not lost and would presumably be examined at the next state change event unless superseded by a subsequently received Announce message from the same source. For the MASTER state this concern does not apply. Once a port is in the MASTER state there is only one non-fault or non-user directed circumstances which can cause it to change state. This circumstance is that an Announce message from a better clock is received, qualified and processed per the existing standard. The inclusion of the 'prior computation' of the referenced item 'c' cannot affect this since the result of this prior computation by definition either caused the port to enter the MASTER state or allowed it to remain in the MASTER state. The case of a port in the PASSIVE or UNCALIBRATED states is similar to a port in the SLAVE state. Ceasing to receive An-

nounce messages from the port that caused the receiving port to enter the PASSIVE or UNCALIBRATED states will be detected by the Announce receipt timeout mechanism as currently specified in the standard, 9.2.6.11. However the circumstances outlined earlier can cause an Announce message from the current master clock to be absent from an evaluation of the best master clock algorithm during a state change event thus reaching the false conclusion that the receiving clock is the best clock visible of the port. Uncorrected this can cause a port in the PASSIVE or UNCALIBRATED states to change to the PRE_MASTER and potentially the MASTER state even though the Announce timeout mechanism has not been invoked. To prevent this we recommend the following:

1. Broadening item 'c' in 9.3.2.3 to include the PASSIVE and UNCALIBRATED states, that is we recommend that item 'c' be implemented as though it read "If port 'r' is in the SLAVE, UNCALIBRATED or PASSIVE states,…"
2. Augment 9.5.3 as follows: a. In Figure 29 change the decision box currently containing the word SLAVE to contain the words SLAVE, UNCALIBRATED or PASSIVE b. In the text of 9.5.3 in the 4th paragraph first bullet change the words "…in the SLAVE state" to read "…in the SLAVE, UNCALIBRATED or PASSIVE states" c. In the text of 9.5.3 in the 4th paragraph second bullet change the words "The port N is in the SLAVE state and…" to read "The port N is in the SLAVE, UNCALIBRATED or PASSIVE states and…" d. In the last paragraph of 9.5.3 just before the note change the words "…from the current parent clock, the data…" to read"…from the current parent clock of a port in the SLAVE state, the data…" e. Add a paragraph just before the NOTE reading: "If an Announce message is received from the current parent clock of a port in the PASSIVE, or UNCALIBRATED states, the parent data set of the receiving clock shall be updated as indicated in table 15 except that the source of each field shall be the received Announce message rather than Ebest."
3. Change table 15 in 9.3.5 to include an update of the parent data set of the clock as follows: a. parentDS.parentPortIdentity updated by sourcePortIdentity of Ebest. b..parentDS.grandmasterIdentity updated by grandmasterIdentity of Ebest c. parentDS.grandmasterClockQuality updated by grandmasterClockQuality of Ebest d. parentDS.grandmasterPriority1 by grandmasterPriority1 of Ebest e. parentDS.grandmasterPriority2 by grandmasterPriority2 of Ebest.
4. In 9.2.6.11 change the words in bullet 'd' from "…indicated by a comparison of the sourcePortIdentity fields of the respective messages." To read "…indicated by a comparison of the sourcePortIdentity field of the received Announce message with the parentDS.sourcePortIdentity of the receiving clock."

We further recommend that this issue be discussed when the next edition of the standard is prepared.

Note that although this issue appears to be a design fault in the standard it is unlikely to produce any protocol failures in the vast majority of circumstances. Likewise the implementation of these recommendations will be completely interoperable and will produce the desired behavior for ports in the PASSIVE or UNCALIBRATED states that also implement these recommendations. The inclusion of clocks implementing this recommenda-

tion in a system including clocks that do not implement these recommendations will not produce behavior any more adverse than produced in the current situation. What the current situation will produce is for a port, say port A, in the PASSIVE or UNCALIBRATED states to intermittently change to the MASTER state and start to issue Announce, Sync and Follow_Up messages. The presence of these Announce messages will not result in the legitimate master, port B, changing its state since by definition it is a better clock (otherwise it rather than port A would have been in the PASSIVE or UNCALIBRATED state). Furthermore, port A will revert to the PASSIVE or UNCALIBRATED state as soon as it begins to evaluate the Announce messages from port B. Nor will the presence of these intermittent messages from port A normally have any effect on ports in the SLAVE state. Once a slave port, say port C, determines that port B is its master, it is then required to ensure that only Sync and Follow_Up messages from port B are used in the synchronization process, 9.5.4. The intermittent Announce messages from port A, if they make it past the foreign master qualification process of C will be evaluated but will not result in a change of state since port C like ports A and B will conclude based on Figure 27 part 1 that port B represents a better clock than that of port A. The potential point of failure is if there are a sufficient number of ports in the PASSIVE or UNCALIBRATED states falsely entering the MASTER state to the extent that the number of Announce messages present exceeds the capacity of the foreign master tables of other ports thus preventing consideration of an Announce message that in fact was from a better clock than the recipient. Since the minimum permitted capacity of the foreign master table is five, 9.3.2.4.5, this can only occur if there are in excess of 5 class 1-127 devices in the system more or less simultaneously making this inappropriate transition from PASSIVE or UNCALIBRATED to MASTER states. This design fault may as noted result in the presence of a low level of spurious (although benign as far as the protocol), network traffic.

Interpretation Request #20
Topic: Clock and Port Identity **Subclause:** 7.5.2

Subclause 7.5.2 and its subclauses describe portIdentity and clockIdentity. The portIdentity for a port is a structure that contains two members - clockIdentity and portNumber. Each port of a clock has a portIdentity, and 7.5.2.3 indicates that the values of the port numbers for the ports on a clock that has N ports are numbered 1, 2, …, N. In addition, 7.6.2.1 says that the clockIdentity of a clock shall be as specified in 7.5.2.2. It is assumed that the clockIdentity member of the portIdentity attribute of each port of a clock is the same as the clockIdentity of the clock. However, no explicit statement of this in IEEE Std 1588-2008 can be found, nor any explicit statement that the clockIdentity members of the portIdentity attributes of the respective ports can be different from each other or different from the clockIdentity of the clock. Can this point be clarified, i.e., a) Is the clockIdentity member of the portIdentity attribute of each port of a clock required to be the same as the clockIdentity of the clock? b) If the answer to a) is no, can a PTP profile require that the clockIdentity member of the portIdentity attribute of each port of a clock be the same as the clockIdentity of the clock? Related to the above, if the answer to a) is yes, then the clockIdentity member of the portIdentity attribute of each port is the same as the clockIdentity of the clock, which is stored in the default data set.

In this case, it seems unnecessary to also store this same clock identity in the portIdentity member of the port data set; the portNumber would be sufficient. Is it permissible to store only the portNumber in the port data set, and obtain the clockIdentity from the default data set?

Interpretation Response #20
There is a single value of clockIdentity, maintained in the default dataset per 8.2.1.2.2, that applies to the entire clock and therefore to the clockIdentity member of the portIdentity of each port. There are no requirements in the standard governing the internal representations of datatypes or storage schema. Therefore, it is permissible to store only the portNumber in each port dataset and then reconstruct the complete portIdentity using the portNumber from the port dataset and the clockIdentity from the default dataset when needed for computations specified in the protocol or when populating PTP messages.

Interpretation Request #21
Topic: Port Numbering **Subclause:** 7.5.2.3

Subclause 7.5.2.3 describes portNumber allocation by PTP node. "The value of the portNumber for a port on a PTP node supporting a single PTP port shall be 1. The values of the port numbers for the N ports on a PTP node supporting N PTP ports shall be 1, 2, …N, respectively." In case of a PTP node with fixed port configuration, the numbering is straightforward. However, IEEE Std 1588-2008 is being implemented in modular or flexible equipments using various blades and variable number of ports per blade, with blade that can be dynamically changed. Moreover, use of virtual interfaces, as such VLANs, would increase the potential number of PTP port per physical interface because PTP port is "a logical access point of a clock for PTP communications to the communications network." The sequential, continuous numbering required by the specification would thus be really difficult to maintain. Moreover, dynamic configuration (hardware or software) change can easily lead to instability in PTP behavior. What would be the best and easiest option to handle numbering in such equipments? One suggestion made would be "to assign port numbers ranges to each blade slot and, if numbers in that range are not used, either because the slot is empty or the blade does not fully populate the assigned number range, to consider the "unused" numbers for that slot to be port numbers on "virtual ports" that are in the disabled or faulty PTP state (and hence do not place synchronization related traffic on the network). A node in the faulty or disabled state is not supposed to put Announce, Sync, etc messages on the PTP communication path. In the disabled state it would be expected to respond to an ENABLE management message which it clearly won't if it is not present." One comment on this suggestion is that having potentially plenty of port in faulty state may not look "clean"; disabled would be better. It is always possible from an implementation viewpoint to "hide" the faulty (unused) ports, but that would be implementation specific. It just seems very confusing for ports to appear as faulty simply because they don't exist or aren't configured to run IEEE Std 1588-2008. Also, it can be assumed that, in such large implementation, (a) management port(s) would be specifically configured for this purpose and thus be enabled.

Interpretation Response #21
The standard specifies the following constraints on portNumber values and the value of defaultDS.numberPorts: 1) Ports numbers don't change dynamically -- 8.2.5.2.1 requires that the portIdentity be a static member of the dataset. If you need to renumber ports, this can only be done when a port is either offline or in the INITIALIZATION state of the state machine (see Table 10, 9.2.5). 2) The total number of ports, defaultDS.numberPorts is required by 8.2.1.2.3 to be a static member of the dataset. This value may be decreased but only when any ports affected by this change are either offline or in the INITIALIZATION state of the state machine (see Table 10, 9.2.5). The value may be increased at any time since this will not affect any ports with port numbers less than the original value. 3) Subclause 7.5.2.3 requires that port numbers have the values 1,2,3…N where N is the value of defaultDS.numberPorts. It is required by 8.2.1.2.3 for N to include ports that are in any PTP state of Table 10, and may include ports that are permanently or temporarily missing. 4) Subclause 19.2.3 requires that a conformant device be conformant not only to IEEE Std 1588-2008 but to a PTP profile. Also required is that if a PTP profile does NOT specify a particular value or option, then the device must conform to the choice made in one of the standard specified profiles in Annex J. Clause 19.2.1.2 specifically permits a profile to define a management scheme other than the scheme of 15.2, but it follows from 19.2.3 that if this is done it must be specified in a profile that meets the requirements of 19.3. Within these constraints, the management of port numbering is implementation dependent. Therefore, it is permitted for example when using the 15.2 management message structure on a blade architecture to:

1. Assign a range of port numbers to each slot with any missing ports regarded as being either faulty or disabled, or
2. Treat unpopulated port numbers as missing. This requires that appropriate responses be given to management messages querying these ports. For example an appropriate error message from Table 72, perhaps WRONG_VALUE or a profile defined value, should be returned when the PORT_DATA_SET (15.5.3.7) of a missing port is queried. Similarly, in response to a PORT_DATA_SET query directed to all-ports (Table 36), a clock should send a response for each of its ports, regardless of the port state, except for those that are missing.

Likewise, it is permitted to provide management ports to handle such implementations (see 15.1.1) which specifically allows for management schemes as alternatives to the specific management message architecture of 15.2 provided they are specified in a PTP profile.

Interpretation Request #22
Topic: Definition of an Alternate BMCA

The specification of a telecom profile is under development. The profile is for frequency distribution between Ordinary Clocks acting as masters and Ordinary Clocks acting as slaves, and without any IEEE Std 1588-2008 support from the network nodes (i.e. no

Boundary Clocks or Transparent Clocks).However, to enhance network reliability in the event of possible link failures isolating individual network elements, network operators typically engineer the network such that clocks have access to multiple timing sources. In the case of IEEE Std 1588-2008, this is thought to be equivalent to having two or more master clocks active within a single domain. With this in mind, developing an alternate best master clock algorithm is under review. It is expected to define separate BMCA for an Ordinary Clock acting as slave and an Ordinary Clocks acting as master. However, the following points require clarification. IEEE Std 1588-2008, 9.3.1 specifies that: "PTP permits the use of two forms of best master clock algorithm: - By default, the mechanism specified in 9.3.2, 9.3.3, and 9.3.4 - If specified in a PTP profile, an alternate best master clock algorithm" The default BMCA as specified in the 9.3.2, 9.3.3, and 9.3.4 makes sure that only one Ordinary Clock will have its PTP port in the MASTER state within a PTP domain after the default BMCA operation. Question: Is it possible, as part of the definition of an Alternate BMCA in a PTP profile, to define a different behavior which would lead to having several Ordinary Clocks having their PTP port in the MASTER state after the Alternate BMCA operation? Or is this behavior violating some basic rules specified in IEEE Std 1588-2008? In other words: can it be specified in a PTP profile an Alternate BMCA which would elect several Ordinary Clocks as the grandmasters of the PTP domain?

Interpretation Response #22 - Introductory comment to Interpretation #22 and Interpretation #23:
The introductory remarks submitted by the questioner reflect a very restricted operational model and set of requirements. In particular, the questioner desires to operate IEEE Std 1588-2008 in an environment where:

1. There are no boundary or transparent clocks present,
2. Ordinary clocks are either slave-only or are to be operated as master clocks (and we understand that in the questioner's context, these master clocks are, for example, linked to a GPS time source and therefore will have a clockClass value less than 128 which leads the clock to be either in the master or passive states, but never the slave state),
3. PTP communication topology is to be set by configuration rather than by the use of a distributed algorithm,
4. Slave-only clocks have access to timing messages from a configured set of ordinary clocks acting as masters with the selection of timing messages to be used by the slave being under control of the slave-only clocks.

It is felt that there are several solutions to this requirement set possible while still retaining conformance to IEEE Std 1588-2008. However, since these interpretations are available to a wider audience that may or may not share these requirements, Interpretation #22, Interpretation #23 and Interpretation #24 are intended to be general in scope and, where needed, specific restrictions to the particular requirements of the questioner are given.

These three questions concern fundamental aspects of the PTP protocol as specified in the standard. For this reason, the committee calls attention to the following clauses of IEEE Std 1588-2008 to provide context for the responses to these two questions:

1. 7.1 Domains. The important points for this discussion are:
2. The domain limits the scope of operation of the protocol to the set of clocks belonging to the domain.
3. The operation of the PTP protocol in two different domains is independent.
4. The note suggests several mechanisms for implementing a domain, that is, mechanisms for ensuring that the protocol implementations in different domains do not interact. Note that it is also possible to use unicast transmissions to implement the separation of domains PROVIDED that in each domain the specifications of 7.3.1 are met, namely "… that the behavior of the protocol is preserved".
5. Note that 7.1 specifically states that a PTP device may participate in multiple domains PROVIDED that the operation of PTP in one domain does not affect the operation in another domain. To further illustrate this point, note that two clocks communicating via a unicast transmission that does not include boundary clocks in the path define a PTP domain PROVIDED that if either of the clocks participates in another domain that the operation of the PTP protocols in each domain are independent. Such a path may also include a boundary clock PROVIDED that no clocks in a second domain use the boundary clock for communication.

3.1.35 PTP port, 3.1.22 ordinary clock, 3.1.3 boundary clock. From these clauses and the clause on Domains, the following observations are relevant to the questions posed

A device may consist of multiple ordinary clocks, that is, it has multiple PTP ports EACH in a different domain. Note that these PTP ports can be implemented on a single or on multiple physical ports provided the independence of the PTP domains is supported by the communication mechanism or the use of the PTP domain number.

A device with multiple PTP ports in the same PTP domain (irrespective of the number of physical ports involved) is a boundary or transparent clock PROVIDED the device correctly implements the PTP protocol for boundary or transparent clocks respectively, as specified in the standard. If such a device does not so implement the PTP protocol, then it is out of scope of the standard and no assurance can be given on the operation of the protocol in a system containing such a device.

Response to the question:
"Is it possible, as part of the definition of an Alternate BMCA in a PTP profile, to define a different behavior which would lead having several Ordinary Clocks having their PTP port in the MASTER state after the Alternate BMCA operation? Or is this behavior violating some basic rules specified in IEEE Std 1588-2008? **Answer**: Within a domain, the normal operation of the PTP protocol requires that there is either zero or one PTP port in the MASTER state on a communication path. The use of the acceptable master table can result in more than a single master on a communication path. However, the behav-

ior of the protocol under these conditions is out of scope of the standard. **Explanation**: See 3.1.17 "master clock: In the context of a single Precision Time Protocol (PTP) communication path, a clock that is the source of time to which all other clocks on that path synchronize."For example, a system consisting of only slave-only clocks will not have any PTP port in the master state. Note that 17.6 provides a configuration option, the acceptable master table that allows an operator to restrict the set of clocks to which a port configured with this option synchronizes. Also, note the caution given in 17.1 which states that the use of this option may cause multiple ports on a communication path to be in the master state. Whether this causes failure of the protocol depends on the number of such masters and on the specific method of implementing the protocol. For example, an implementation of the foreign master filter will have finite capacity. Therefore, correct operation of this filter may depend on the normal operation of the protocol where eventually there is only a single master on the communication path. An example where the use of the acceptable master table causes multiple master ports on a single communication path is as follows: Consider three ordinary clocks A, B and C all connected via a transparent clock (or equivalently a system with no boundary or transparent clocks but only ordinary bridges supporting multicast communication). If under normal operation of the protocol the best master clock algorithm would rank these clocks in the order A, B, C then A would be the grandmaster and both B and C would be in the slave state. However, if, say, C's port was configured with an acceptable master table listing only A and clock A failed (or was otherwise removed from the system), then C would not process any Announce messages from an acceptable master and would therefore transition to the master state, however B upon receiving Announce messages from C would correctly also be in the master state (because the best master clock algorithm operating in B determines B to be better than C). **Response to the question:** "In other words, can it be specified in a PTP profile an Alternate BMCA which would elect several Ordinary Clocks as the grandmasters of the PTP domain?" **Answer**: By definition, PTP allows only a single grandmaster in a domain -- see 3.1.13. **Comment**: HOWEVER, it is believed that this question was posed in order to understand whether a PTP communication path can operate with the PTP ports of more than one Ordinary Clock (OC) or ports on a Boundary Clock (BC) acting in a way that is functionally identical to the actions of a port in the MASTER state. This is the purpose of the Alternate Master Flag and 17.4. A port that is not in the MASTER state (in this case, in the PASSIVE state) may:

- Clause 17.4 allows such a port to transmit multicast announce, sync, follow-up and delay response messages provided that the Alternate Master Flag is set.
- Clause 16.1 allows such a port to transmit unicast announce, sync , follow-up and delay response messages according to the requirements of the Unicast message negotiation. It is recommended, following 7.3.8.2, that if this unicast transmission originates from a port in the passive state, that the Alternate Master Flag be set.

The Alternate Master operation is designed to allow more than one provider of time (a potential master) to transmit timing information to the slave nodes. If this mechanism is used, rather than the standard being modified to give the same functionality, it will allow greater inter-operability.

Interpretation Request #23
Topic: Definition of an Alternate BMCA

Is it possible, as part of the definition of an Alternate BMCA in a PTP profile, to define a different behavior which would lead having no Ordinary Clock having their PTP port in the MASTER state after the Alternate BMCA operation? Or is this behavior violating some basic rules specified in IEEE Std 1588-2008? In other words: some of the Ordinary Clock PTP ports would be in the PASSIVE state, and the other Ordinary Clock PTP ports would be in the SLAVE state.

Interpretation Response #23
Response to the question: "Is it possible, as part of the definition of an Alternate BMCA in a PTP profile, to define a different behavior which would lead having no Ordinary Clock having their PTP port in the MASTER state after the Alternate BMCA operation? Or is this behavior violating some basic rules specified in IEEE Std 1588-2008?"
Answer: Clause 9.3.1 specifically permits an alternate best master clock algorithm to configure the recommended state of the port on a clock as long as all of the requirements in 9.3.1 are met. Note that the determination of the actual port state from the recommended state depends on the operation of the applicable state machine -- see Figure 23 or Figure 24. Therefore, an alternate best master clock algorithm is permitted to result in a recommended state of passive for all ordinary non-slave-only clocks as suggested. **Comment**: In the absence of boundary clocks this would mean that the best master clock algorithm has selected no clock as a master on a communication path. If this were to occur, one or more of the clocks in the passive state would eventually experience an "announce_receipt_timeout_expires" event which from the state machine of Figure 23 would result in the clock entering the master state. Note that this timeout mechanism and the state machines are not part of the best master clock algorithm and are not subject to modification by a profile. This would likely result in thrashing. If boundary clocks were present then one or more ports on the boundary clocks would be in the master state unless the profile defined the alternate BMCA operation for a boundary clock. It is strongly recommended that any profile defining an alternate BMCA also define how it operates in a boundary clock. Note that the difficulty with the occurrence of an "announce_receipt_timeout_expires" event can be overcome by having the profile specify:

1. Separate values of the announce_receipt_timeout for the listening and all other states.
2. For the listening state the value determines how long the port waits before leaving this state- normally a short time.
3. For the other states a value of infinity may be used to "turn-off" this mechanism thus preventing the condition described above.
4. Specify the value 255 (the allowed range of this attribute is 2-255) to be interpreted as infinity for devices built to the profile.

As discussed above in the response to question 1, it is possible to use the Alternate Master flag to ensure that the only difference between the operation of the best master and other nodes (that are potential masters) to be the value of the alternate master flag. In effect, this means that a communication path may operate with many devices delivering timing information that would normally only be delivered by a single master on each communication path.

Interpretation Request #24
Topic: Definition of an Alternate BMCA

Clause 17.4 of IEEE Std 1588-2008 specifying the optional "alternate master" mechanism mentions that the PTP port of an alternate master that is not the best master shall send Announce, Sync, and Follow-up messages with the Alternate Master Flag true.. However, Table 10 in 9.2.5 states: "Passive - The port shall not place any messages on its communication path except for Pdelay_Req,Pdelay_Resp, Pdelay_Resp_Follow_Up, or signaling messages, or management messages thatare a required response to another management message." So, can an Ordinary Clock having its PTP port in the PASSIVE state send Announce, Sync, Follow-up, and Delay_Resp messages with the Alternate Master Flag set to true? Moreover, is it possible, as part of the definition of an Alternate BMCA in a PTP profile, to use the "Alternate master" optional PTP mechanism specified in 17.4 in order to enable an Ordinary Clock having its PTP port in the SLAVE state to synchronize to another Ordinary Clock having its PTP port in the PASSIVE state? Indeed, this behavior is not specifically depicted in the 17.4, but nothing in the 17.4 seems to be in conflict with this behavior.

Interpretation Response #24
The following responses indicate the interpretation of the Sponsor of the standard after consideration of your request. Note that changes to the standard can only occur after those changes have been balloted by the IEEE Sponsor and approved by the IEEE-SA Standards Board. **Response to the question:** "So, can an Ordinary Clock having its PTP port in the PASSIVE state send Announce, Sync, Follow-up, and Delay_Resp messages with the Alternate Master Flag set to true?": **Answer:** The interpretation of the Sponsor is that 17.4 should be followed. Note that in 9.2.4, "The behavior of the states of a port associated with the state machines of Figure 23 and Figure 24 shall be as defined in Table 10 with the exception of the provisions for unicast messages specified in 16.1." This clause should also include the exception for the provisions of 17.4. **Response to the question**: "Moreover, is it possible, as part of the definition of an Alternate BMCA in a PTP profile, to use the "Alternate master" optional PTP mechanism specified in 17.4 in order to enable an Ordinary Clock having its PTP port in the SLAVE state to synchronize to another Ordinary Clock having its PTP port in the PASSIVE state?" **Answer**: If specified in a profile, a slave port may synchronize based on any set of received timing messages irrespective of the value of the alternate master flag in these messages.

Interpretation Request #25
Topic: Delay asymmetry correction

Clause 11.6.3 states that "For a boundary or ordinary clock, prior to transmission on an egress port the correctionField of the transmitted Delay_Req message shall be modified by subtracting the value of the egress path delayAsymmetry from the correctionField of the transmitted Delay_Req message."

For boundary and ordinary clocks, the correctionField is initially set to 0, according to 11.3.2. The correctionField, as defined in 13.3.2.7, cannot have a negative value.

Thus, the question is: how is the delayAsymmetry subtracted from the correctionField in this case?

Note: the same question applies also to 11.6.4 for Pdelay_Req.

Interpretation Response #25
Clause 11.3.2 Specifications for Delay_Req messages should be interpreted as an ordered list of actions thus:

- 11.3.2-b-3 requires the correctionField to be initialized to 0 FOLLOWED BY applying any asymmetry correction per 11.6.3

Clause 11.4.3 Specifications for Pdelay_Req messages should be interpreted as an ordered list of actions thus:

- 11.4.3-a-1 requires the correctionField to be initialized to 0 FOLLOWED BYapplying any asymmetry correction per 11.6.4

Clause 13.3.2.7 Specifies the details of the correctionField as follows:

- It is an Integer64 hence can represent both positive and negative values
- There is a distinguished value indicating that the correction is too big to be represented. This distinguished value is 0x7FFFFFFFFFFFFFFF, i.e. one in all bits except the most significant.

With this interpretation of the specifications there is no difficulty in executing the required subtraction in generating Delay_Req or Pdelay_Req messages.

Interpretation Request #26
Topic: Message timestamp point **Subclause:** 7.3.4.1

Where precisely is the message timestamp point with PTP over IEEE Std 802.3 /Ethernet 100Base X?

Clause 7.3.4.1 states that "… the message timestamp point for an event message shall be the beginning of the first symbol after the Start of Frame (SOF) delimiter." Presumably the timestamp point is thus the beginning of the first bit of this first symbol. However, 100Base-X uses NRZI, wherein where a polarity transition represents a logical ONE, and the absence of a polarity transition denotes a logical ZERO. Is the start of the first bit the point where a polarity transition would occur should there be one, or midway between the point where a polarity transition would occur in the first bit should there be one and the transition of the previous bit?

This is important as the time difference between these two points is 4 ns, much more the PTP synchronization accuracy better than 1 ns claimed in the scope.

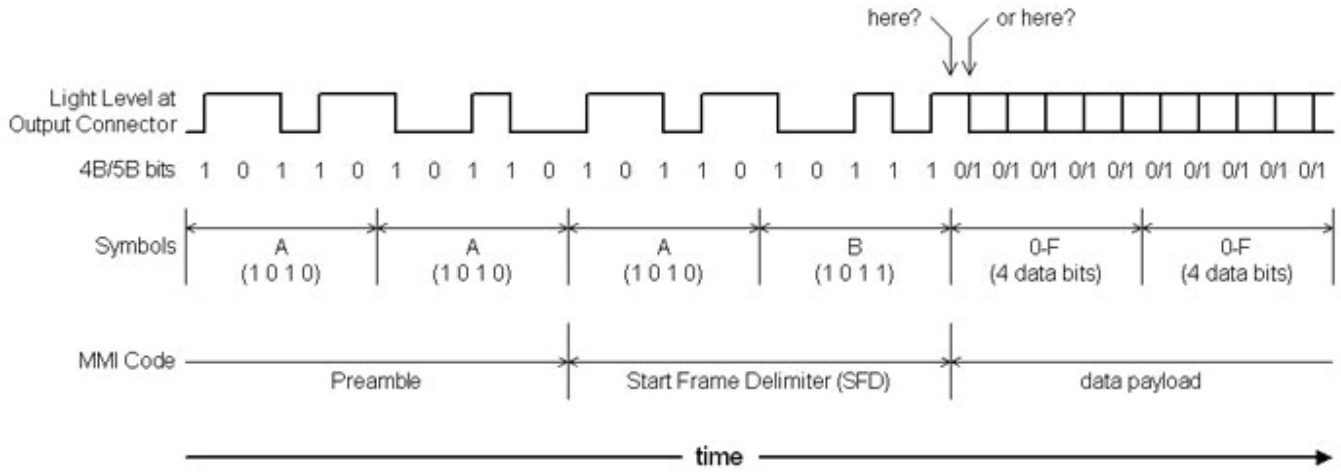

Figure 1 - Message timestamp point on a 100Base-FX frame

Interpretation Response #26
Message timestamp point for 100Base-X NRZI encoding with a specific example provided for 100Base-FX. Reference 7.3.4.1

First, note that the scope does NOT state that PTP achieves 1 ns accuracy. The statement in 1.1 is that it "permits" accuracies better than 1 ns. The basis for this is the ability to carry fractional ns time stamps. However, it is true that to achieve ns level accuracy that the timestamp point must be identical in all PTP nodes to avoid introducing asymmetry errors that lead to a bias in the synchronization.

Clause 7.3.4.1 as noted defines the message timestamp point as the beginning of the first symbol after the start of frame delimiter. Clause 7.3.4.2 states that the timestamp is referenced to the point where the message timestamp point passes the clock reference plane that marks the boundary between the clock and the network and illustrated in Figure 19. For Ethernet this reference plane should be interpreted as a point on the physical media side of the PHY and therefore the question of symbol and bit boundaries in the on-the-wire encoding is important. These encodings are not specified by IEEE Std

1588-2008 but by the standards relevant to the transmission protocol used.

In the case of Ethernet IEEE Std 802.3, 24.1.4.3 states that for 100Base-X the on-the-wire format uses the FDDI signaling defined in ISO/IEC 9314-3:1990 and ANSI X3.263-1995 (TP-PMD). It further notes that this is NRZI encoding.

In IEEE Std 802.3, 1.4.235 notes that NRZI encoding specifies that a polarity transition represents a logical ONE and the absence of a polarity transition denotes a logical ZERO.

The transition point for a bit value of 1 defines the bit boundaries (which of course for 0s must be inferred from the last 1 transition). These bit boundaries are to be used in determining the message timestamp point.

Therefore in the figure provided by the questioner the message timestamp point is the choice on the right.

This definition of bit boundaries is consistent with diagrams found by searching for NRZI on the internet.

Interpretation Request #27
Topic: Math on maximum value of correctionField **Subclause:** 13.3.2.7

It seems that the IEEE Std 1588-2008 should have explicitly specified that before modifying the correctionField a clock must check that the its value is not the special 'too-big' value, and if it is, leave the correctionField alone.

(The whole arithmetic of the correctionField near the 'too-big' value is not clear either, i.e. it doesn't seem correct to wrap around, as you might understand from 5.2, rather it should be 'stuck' at the maximum 'too big' value. However, for all practical reasons this seemingly larger problem is non-existent and hence solving is not recommended).

Interpretation Response #27
Clause 13.3.2.7 defines the data type of the correctionField as an Integer64 and it represents a correction in nanoseconds scaled by 2**16. A value of all bits set to '1' is reserved to indicate that the correction is too big to be represented. A reading of clause 16 shows that the correctionField is intended to allow accumulation of residence times within transparent clocks along the path. If at some point the addition of an error is too big to be represented then the value all bits 1 is to be used. (This condition is very unlikely to occur in any real network as the maximum value is 2**(63-16) ns = 2**47 ns or about 40 hours). Once this condition occurs it is not possible to infer anything other than 'too big', for example even if the next correction would reduce the actual accumulation it is impossible to determine this from the 'too big' value.
The correct mathematics is therefore: tooBig + or - any value is still tooBig.

How a slave handles a message with a correctionField value of 'tooBig' is implementation

or profile specific and is outside of the scope of IEEE Std 1588-2008.

Interpretation Request #28
Topic: clockAccuracy **Subclause:** 3.1.1, 7.6.2.5, Table 6

Accuracy is defined in IEEE Std 1588-2008, 3.1.1 as "The mean of the time or frequency error between the clock under test and a perfect reference clock, over an ensemble of measurements…" Precision, per the same definition, "…is a measure of the deviation of the error from the mean."

IEEE Std 1588-2008, 7.6.2.5 states that "The clockAccuracy characterizes a clock for the purpose of the best master clock (BMC) algorithm. The value of clockAccuracy shall be taken from the enumeration in Table 6. The value of this attribute shall be estimated by the clock to a precision consistent with the value of the selected enumeration, e.g., for 2316 a precision of plus or minus 0.5 _s."

IEEE Std 1588-2008, 7.6.2.5 also states that "The clockAccuracy indicates the expected accuracy of a clock when it is the grandmaster, or in the event it becomes the grandmaster."

Should clockAccuracy be interpreted as the expected accuracy (mean time error), precision (deviation from mean value) or the sum of the expected accuracy and precision?

Interpretation Response #28
The clockAccuracy indicates the expected accuracy of a clock when it is the grandmaster, or in the event it becomes the grandmaster. The clock accuracy indicates how far the mean value of the time of the clock, when it is the grandmaster, departs from the timescale of the domain, that is, if the timescale is PTP, how accurate is it compared with TAI. This estimate is based on the timeSource attribute, the elapsed time since last synchronized to this time source, and the holdover specifications of the clock.
The offsetScaledLogVariance captures the inherent stability and precision of the clock.

Therefore the clock accuracy is to be interpreted as the expected accuracy (mean time error).

Interpretation Request #29
Topic: Data field in Management messages of GET, RESPONSE, COMMAND, or ACKNOWLEDGE types **Subclause:** 15, Table 38

In IEEE Std 1588-2008, 15, there is no statement about what happens to the dataField for management messages based on the verb used. For example - if a SET management message is sent, a RESPONSE message must be generated. But what goes in the response's dataField portion? Should it be filled in with the same material just set? If so, does this also hold true for GET messages - when asking for a clock's Parent Data Set, for example, should the dataField of the message be filled in with one's own? Or, are

these fields truncated where they are unnecessary and omitted?

Interpretation Response #29
For each of the management messages defined in 15, the contents of the dataField are specified. See for example IEEE Std 1588-2008, 15.5.3.1.2 CLOCK_DESCRIPTION, Table 41; or in IEEE Std 1588-2008, 15.5.3.1.4 SAVE_IN_NON_VOLATILE_STORAGE "The data field is of zero length". In IEEE Std 1588-2008, 15.5.2.3, Table 40 enumerates the possible action fields for each of the defined management messages.

An examination of messages for which the action is COMMAND or ACKNOWLEDGE indicates that the dataField for these messages is of zero length.

For messages with action GET or SET the contents of the dataField is specified in clause defining the message. Table 38 under 'RESPONSE' clearly states that the contents of the RESPONSE message to a successful GET or SET is to contain the current values of the data defined by the dataField of the GET or SET message.

In the case of an error in the execution of a management message, a management error status TLV is to be returned as defined in 15.5.4.

Interpretation Request #30
Topic: Ambiguity of CANCEL_UNICAST_TRANSMISSION TLVs
Clause, Subclause, Annex, Figure, or Table: Subclause 16.1.1

If using unicast negotiation, and a port has two grants for the same peer: one as grantee (the port is receiving PTP messages from the peer), and one as grantor (the port is sending PTP messages to the peer), then it is not possible to determine which of the two grants is intended to be cancelled when the port receives a CANCEL_UNICAST_TRANSMISSION TLV from the peer.

Should the port cancel both grants, pick one of the grants (if so, how should it decide which?), or neither?

When using the default BMCA as described in the standard, cancelling both grants can lead to infinite port state flaps, as outlined below:

1. Consider two PTP ports connected over a single link, port 1 and port 2. Port 1's clock is of higher quality than port 2's clock. Both ports start in Master state, and request Announce message grants from the other port.
2. Port 2 qualifies the clock received from port 1, and so moves into Slave state. As ports in Slave state should not send Announce messages, port 2 sends a CANCEL_UNICAST_TRANSMISSION TLV to port 1, indicating that it is no longer going to send Announce messages.
3. Port 1 receives the CANCEL_UNICAST_TRANSMISSION TLV. It cannot determine which grant should be cancelled, and so cancels both. It neither sends Announce

messages to port 2, nor expects to receive Announce messages from port 2.
4. As port 2 is no longer receiving Announce messages from port 1, an announce timeout is raised. The clock from port 1 is no longer qualified. At this point, the BMCA indicates that port 2 should move back into Master state.
5. We are now back in the starting state, where both ports are in Master state. They request Announce grants from each other and the cycle continues.

Interpretation Response #30
If a PTP port receives a CANCEL_UNICAST_TRANSMISSION TLV and the port is both a grantor and a grantee of the identified messageType, the port must interpret the TLV both as an indication that the grant it has received will no longer be honored by the remote PTP port and the grant that it has given to that remote port is no longer required.

IEEE Std 1588-2008, 6.1.1 says

A grantor receiving a CANCEL_UNICAST _TRANSMISSION TLV shall always respond with an ACKNOWLEDGE_CANCEL_UNICAST_TRANSMISSION TLV and **may immediately cease to provide the indicated service.**

A grantee receiving a CANCEL_UNICAST_TRANSMISSION TLV shall always respond with an ACKNOWLEDGE_CANCEL_UNICAST_TRANSMISSION TLV and **should immediately cease to use the indicated service.**

The symmetrical case described in the question affects only Announce messages.

A possible solution to the problem that may be immediately implemented in the requestor's node is for a PTP port to not advertise a grandmaster when the node's connection to that grandmaster is via the same PTP port. This operation is not specified in the standard, but the standard never requires a PTP port to advertise in such a manner. The situation described is not strictly permanent because of the requirement in IEEE Std 1588-2008, 9.3.2.5 that stepsRemoved does not exceed 255. It is, however, long lasting.

Adding a feature to identify which grant should be canceled should be considered in the next revision. Note that the obvious resource to use for these corrections are the 8 and 4 bit reserved field of the GRANT_UNICAST_TRANSMISSION TLV and the CANCEL_UNICAST_TRANSMISSION TLV and the ACKNOWLEDGE_CANCEL_UNICAST_TRANSMISSION TLV of Tables 74, 75 and 76 respectively to contain either a key indicating whether the contract referenced in the received TLV is one with the receiver as a grantor or a grantee, or alternatively by a contract identification guaranteed to be unique between the grantor and grantee. Note that the values of the reserved fields are currently zero by definition.

Interpretation Request #31
Topic: Meaning of the clockAccuracy field when operating with the ARB timescale

It is unclear what value the clockAccuracy field should be set to when operating with an arbitrary timescale. For instance, take the case where a PTP master is locked to an atomic frequency source, such as a telecom grade Primary Reference Clock, but is not synchronized to a source of time that is traceable to UTC. The settings in this case could be:

- clockClass = 13 (i.e. an application-specific source of time)
- timescale = ARB (because there is no relationship to the PTP timescale)
- timeSource enumeration = ATOMIC_CLOCK
- frequencyTraceable = TRUE
- timeTraceable = FALSE

The text of IEEE Std 1588-2008 does not make it clear to what the clockAccuracy field indicates accuracy. It could be interpreted as the accuracy of the clock relative to the PTP timescale, or to the arbitrary timescale maintained by the PTP master clock.

If clockAccuracy is accuracy to PTP time, then presumably the correct setting for clockAccuracy is UNKNOWN (0xFE). A possible alternative setting might be "> 10s" (0x31).

If clockAccuracy is accuracy to the ARB timescale being maintained by the PTP master clock, it could be dependent on the implementation of the clock. For example, if the PTP master clock is sufficiently accurate it might be set to a value such as "< 250ns" (0x22).

The guidance listed in Note 1 of Table 7 suggests the values for clockClass, clockAccuracy and timeSource should be consistent; however, the example quoted is not helpful in this scenario, since it relates to the situation where the master clock is hand-set to the PTP timescale.

The purpose of the clockAccuracy field is to be used in the Best Master Clock Algorithm. In the case of a system based on the G.8265.1 Telecom Profile, this is replaced by an algorithm based on profile-specific clockClass values and a locally assigned priority value. Therefore, the clockAccuracy is not used in this mechanism, although some PTP slave clocks report a warning when the clockAccuracy is stated to be UNKNOWN.

However, if the standard BMCA is used, the best master clock is decided on the basis of priority1 first, then clockClass, then clockAccuracy. If the clockAccuracy is set by one implementation to UNKNOWN, this clock will be rejected by the BMCA in favor of another implementation using a finite value for clockAccuracy, even if its source of frequency is of lower quality.

This question arises because we have observed some PTP clock implementations claiming a high degree of clockAccuracy even when not synchronized to a source of time, while others claim an unknown clockAccuracy. This can cause a distortion in the operation of the BMCA, plus cause warnings to be generated by some PTP slave clock implementations about the unknown clockAccuracy.

To summarize, the specific questions to be answered are:

- When operating with a source of accurate frequency (e.g. an atomic frequency source) but there is no synchronization to a source of time (i.e. not even hand set), must the timescale be set to ARB?
- When operating with the ARB timescale, does the clockAccuracy parameter indicate accuracy to the PTP timescale, or to the ARB timescale?
  - If it is PTP timescale, must the clockAccuracy parameter be set to UNKNOWN, or is the more vague association of "> 10s" acceptable?
  - If it is the ARB timescale, can the clockAccuracy parameter be set to any value consistent with the accuracy of the clock's implementation?

Interpretation Response #31
It is correct that the purpose of the clockAccuracy attribute is tied to the Best Master Clock algorithm, BMCA. Specifically, the precedence order for clock selection from Figure 27 of IEEE Std 1588-2008, 9.3.4 is: Priority1, clockClass, clockAccuracy, variance, Priority2, and the tie-breaker clockIdentity. The BMCA was designed with precise traceable time transfer as the target application and therefore favors high quality clocks synchronized with high accuracy to a clock supporting TAI -- the international time standard. However, provision was made for other applications scenarios as described below; in particular, arbitrary timescales and frequency-only applications.

IEEE Std 1588-2008, 7.6.2.4 defines the clockClass. An examination of this attribute reflects the preference for grandmasters, GMs, with time traceable to international standards over all other clocks. Note 1 of this clause makes it clear that the expectation was that for systems transferring frequency only, that an alternate profile would specify clockClass values appropriate to frequency-only transfer.

IEEE Std 1588-2008, 7.6.2.5 defines the clockAccuracy. Note that the primary context of the clockClass attribute is time transfer. The clockAccuracy attribute should therefore be taken as the accuracy of timescale of the GM clock compared to whatever time source the GM is synchronized. Again, the note in this clause makes it clear that the expectation was that a profile would use the values 80 to FD to define accuracy measures appropriate for frequency-only transfer in conjunction with similar frequency-only transfer values of clockClass.

Regarding an ambiguity in the interpretation of Table 6 of IEEE Std 1588-2008,7.6.2.5, the following should be noted. The issue is that the relationship of the value 0x31, i.e. >10s and 0xFE, i.e. UNKNOWN. 0x31, >10s, is supposed to be used for time transfer when the GM, for whatever reason, is only accurate to the stated timescale to >10s but presumably still sufficiently accurate for the application domain and with 0xFE, UNKNOWN, used when this is not the case. It would have been clearer if 0x31 was stated as an upper bound either a specific value or 'designated in a profile'.

It also should be pointed that the system posed is one where frequency-only transfer is the intent but without recourse to the expected use of profile specific values for clockClass and clockAccuracy. Given this, it is incumbent on system designers to ensure that only clocks with clockClassand clockAccuracy values that in conjunction with the BMCA will produce the desired clock ordering are included in the operating domain.

RESPONSE to specific questions:

1. IEEE Std 1588-2008, 7.2.1 clearly states that there are only two possible timescales used in the PTP protocol: PTP and ARB. The PTP timescale is defined as having the epoch defined as a specific TAI time as noted in 7.2.2. All other situations, including those **where no timescale epoch is defined**, must therefore be considered as the ARB timescale.
2. The clockAccuracy attribute **always** defines the accuracy of the GM with respect to the timescale in use, i.e. either PTP or ARB. Therefore, when using the ARB timescale, the clockAccuracy must reflect the accuracy of the GM to whatever epoch is appropriate for the ARB timescale in use.
   - Logically, this question perhaps should be a subparagraph of question 1 however the answer is as follows. With the PTP timescale, the clockAccuracy values 20-31 of 7.6.2.5, Table 6 are appropriate. The value 31 (>10s) is appropriate only if the actual accuracy is known to be acceptable for the application otherwise UNKNOWN should be used.
   - For the ARB timescale (and for that matter the PTP timescale), the clockAccuracy must always reflect the accuracy of the GM implementation in enforcing the epoch of the timescale in use.
     - If the epoch is defined, for example, as 0 at the launch of a rocket, then the clockAccuracy should reflect the GM timescale accuracy with respect to this specific epoch and the values 20-31 would be appropriate. The value 31 (>10s) is appropriate only if the actual accuracy is known to be acceptable for the application. Otherwise, UNKNOWN should be used.
     - If no epoch is defined and enforced by the GM for the application and the profile specific values for clockClass and clockAccuracy are not used, then the clockAccuracy value should be UNKNOWN, i.e. FE.
     - If the clockAccuracy (and presumably the clockClass) attributes are defined in a profile for frequency-only transfer and no epoch is specified, then the timescale is ARB and the clockClass and clockAccuracy follow the definitions in the profile.