7 Clock Synchronization model for a bridged local area network

The clock synchronization model includes (1) synchronization services available to clients, and (2) protocols which support those services. The model defined for this standard is based on a peer calibration approach.

7.1 Peer calibration approach to network synchronization

7.1.1 Independent device clocks

Each node participating in an IEEE 802.1AS network implements a local timescale, i.e. a device clock, and references this timescale in the 802.1AS protocol messages it transmits. These device clocks are independent, and typically are derived from local free-running crystal oscillators. In contrast to some other approaches, the device clock is not adjusted or servoed to follow a master reference.

Operation of the 802.1AS protocol establishes a set of pairwise calibrated relationships among the device clocks, and uses these relationships to create a shared understanding of time throughout the network. These calibration relationships are maintained on a link-by-link basis, in parallel, without reference to any other link, to a master clock or to a preferred direction of information transfer. At this level the model is “democratic”: every device clock is equally valid, and simply defines a way of representing time which is convenient for protocol operations occurring within that device.

In addition to the free-running device timescale, a node may implement a global clock service defined in this standard; some 802.1AS protocol messages reference this global timescale as well as the device timescale. The global clock service at a node may be visualized as a variable-frequency controlled oscillator, slaved to a global master clock. The actual implementation of the global clock service may use a hardware clock servo or may be algorithmic.

7.1.2 Comparison to other synchronization approaches

The synchronization approach used in these protocols is best understood in comparison with other approaches to time distribution.

A basic model for synchronizing distributed clocks is to emit a master signal (e.g. the once-per-minute *beep* from radio station WWV) which is broadcast to all slave devices and updates them. Each slave maintains a controlled clock which “freewheels” between update signals. If the master signal propagated instantaneously and directly, this mechanism would be sufficient to maintain precise synchronization.

When propagation delay and indirect (multi-hop) communications enter the picture, two alternative concepts are often used to extend the above basic model:

1. Measurement of the propagation delays and relay delays (residence times) along the path taken by the master sync signal on its path to each slave, and compensation for this cumulative delay at the slave (e.g. Transparent Clock method in IEEE 1588v2); or
2. Implementing a slave clock at each relay node, and using that slave clock to regenerate and transmit fresh sync signals to stations on the far side of the relay (e.g. Boundary Clock method in IEEE 1588v2).

Both of these concepts can be effective, and have characteristic strengths and weaknesses. However the peer calibration approach used in this standard implements a third, distinct, concept in dealing with propagation delay and indirect communication scenarios.

7.1.3 Peer calibration mechanism

An important element of the peer calibration implementation is the "cross time stamp" or "time affiliation datum". A cross time stamp is useful where several timescales coexist; it is a statement "Time T_A on timescale A is the same instant as time T_B on timescale B." Cross timestamps allow us to make statements *about* time which are true independent of *when* the statement is made or received. This is a fundamental conceptual reorientation from methods based on precisely capturing the arrival of a sync update ("it is now 2:15 – *beep*".)

In the peer calibration approach, global time is distributed from a master station A, through intermediate nodes B and C to end station D, as follows:

1. Station A (master) initiates a clock update cycle, creating a cross-timestamp message "Master clock update: T_{MASTER} is T_A at A"
2. Station B forwards it, rewriting the cross timestamp, saying "Master clock update: T_{MASTER} is T_B at B." It can do this because it maintains a calibrated relationship between timebases B and A.
3. Station C forwards it, rewriting the cross timestamp, saying "Master clock update: T_{MASTER} is T_C at C." It can translate from T_B to T_C because it maintains a calibrated relationship between timebases C and B.
4. Station D receives it, and translates T_C to T_D, obtaining an internal representation of "Master clock update: T_{MASTER} is T_D at D". This station now has the information it requires to update its clock: the master time T_{MASTER} corresponding to a locally-known time T_D.

Note that the update information at station D refers to some instant in the past, but the information is nonetheless accurate, even with no knowledge of how long it took to relay the message from A to D. In this way it is distinct from the alternative 1 in clause 7.1.2. Also note that no intermediate slave clock is used at nodes B or C; in this way it is distinct from alternative 2 in clause 7.1.2. The principle advantage of the peer calibration approach is that it is free from the clock cascade phenomenon, in which a small error in a slave clock's estimation of master time is amplified by a gain factor as the protocol passes the master timescale from station to station. A chain of N cascaded clocks exhibits gain peaking, in which errors scale by k^N. Because an intermediate node in the peer calibration approach never uses an estimate of master time (either to regenerate a sync signal or to measure residence time) there is no cascade; errors in a chain of N stations accumulate additively rather than by power law.

7.1.4 Peer calibration protocol stack

An overall view of the services and protocols for clock synchronization in an IEEE 802.1AS system are illustrated in Figure 7-1.

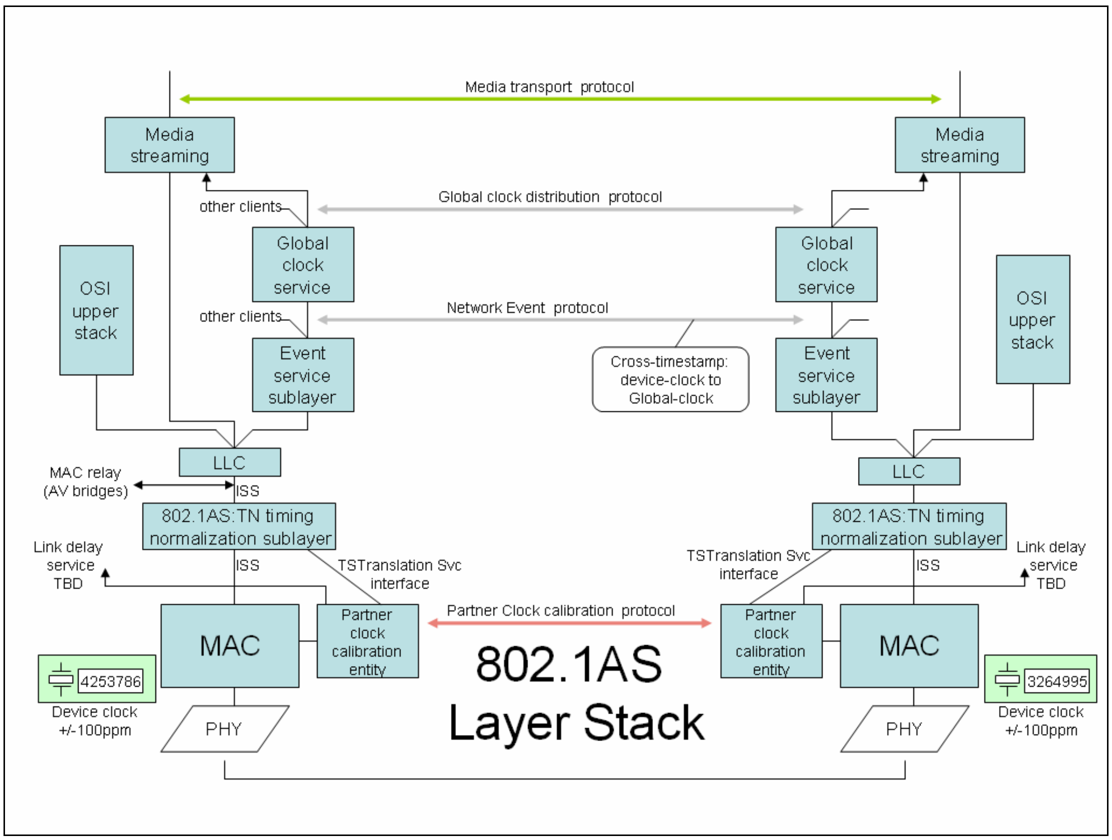

Figure 7-1. Protocol Stack

7.2 Overview of the synchronization services

This standard defines two synchronization services available to clients connected to an IEEE 802 bridged local area network: the *network event service* and the *global clock service*. The network event service is the more fundamental of the two, and assumes only that each client has a locally available timebase as described in 7.1.1. The global clock service is a derived service which provides every client access to a common timescale synchronized to a single grandmaster clock.

The successful establishment of the calibrated time relationships described in clause 7.1, within a set of connected 802.1AS-compliant nodes, renders those nodes capable of delivering the network event service. Information may be transmitted over the network event service using unicast or multicast addressing.

7.2.1 Network event service

The network event service consists of a request primitive and an indication primitive. The service interface exists in the context of a local device timescale.

The semantics of each NetworkEvent primitive is modeled on the IEEE 802.2 DL_UNITDATA primitive and consists of:

- Source MAC Address
- Destination MAC Address
- Local Event Time
- Payload
- Priority

Presentation of a request primitive at one node in an 802.1AS network results in the presentation of a corresponding indication primitive at zero or more connected nodes, as directed by the Destination MAC Address field. Each indication primitive contains the same field values as the request primitive, except that the Local Event Time field is adjusted so as to represent the identical instant of time in the timescale context of the indication.

The payload field is of variable length and is opaque to the Network Event service itself. However this standard defines the first octet of the payload to be an Event Type identifier, assigned as follows:

Event Type value	Meaning
0x00	Global clock service protocol ver. 1 (clause 7.2.2)
0x01-0x7F	Reserved
0x80-0xFF	Unassigned (user specified)

Client applications may find user-specified payload types to be suitable for sensor reports, actuator control, alternative time distribution schemes, and other purposes outside the scope of the global clock service described in clause 7.2.2.

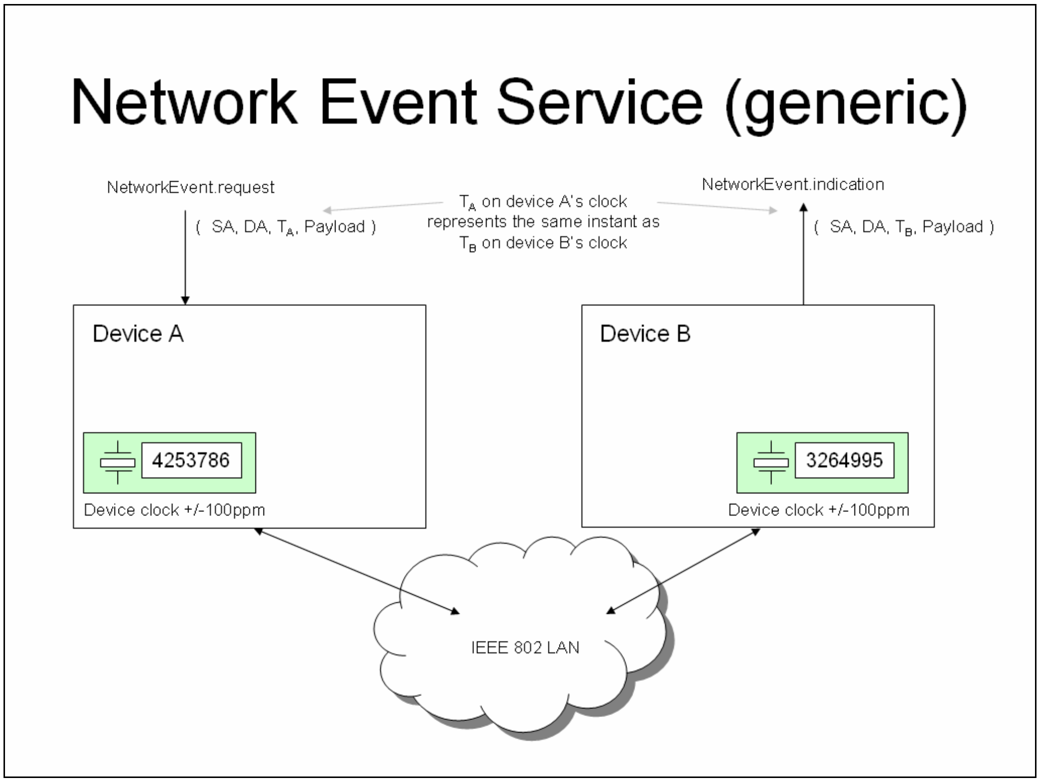

Fig. 7-2. Network Event Service

7.2.2 Global clock service

The global clock service is intended to provide a synchronized timing service throughout a bridged local area network. Its function is comparable to the NTP (network time protocol) service commonly provided in IP (internet protocol) environments.

At any time, a single node is selected as the "grandmaster clock", providing a master timescale which is distributed to all participating 802.1AS nodes in a network. As the network configuration changes (e.g. if the current grandmaster becomes disconnected), a different node may be selected to provide the grandmaster clock function.

An instance of the global clock service presents an interface by which a client may discover the current time. This interface is unaffected by whether the current grandmaster clock is located at a local or distant node in the network.

The global clock provides an "Event/Timestamp" interface. The interface consists of an event request primitive, a timestamp indication primitive, and a status parameter. The event request primitive has no internal fields: the presentation of the request primitive alone constitutes an event. Presentation of an event request primitive by the client results, a short time later, in the presenation of a timestamp indication primitive to the client, containing a numerical value representing the global time at which the event was presented.

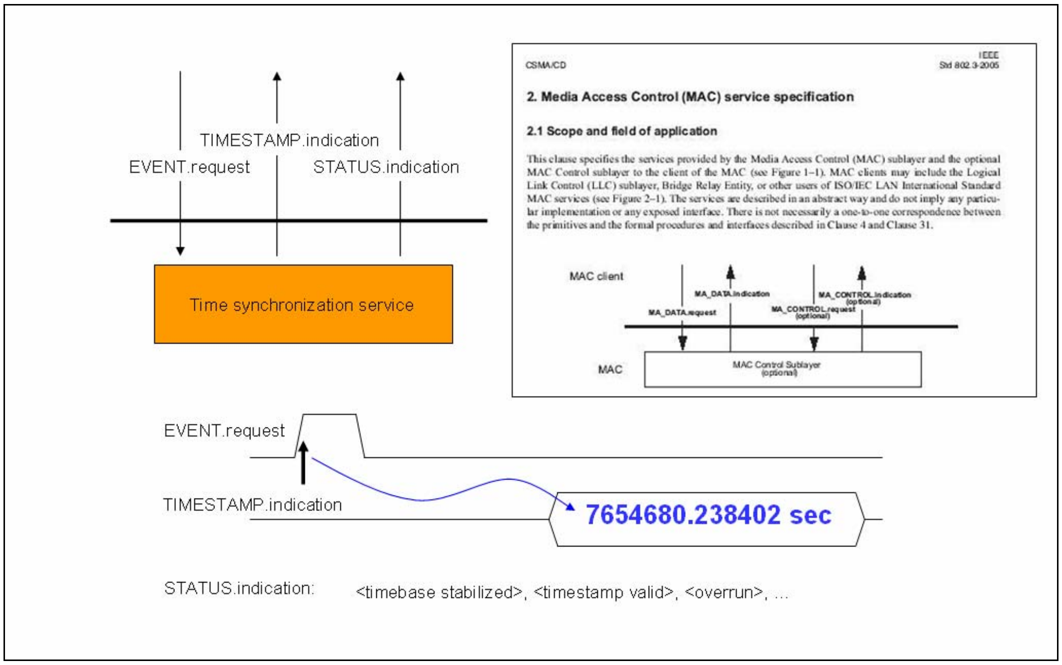

Figure 7-3. Global clock interface

7.3 Overview of the protocols

Two fundamental protocols support the network event service:

1. the partner clock calibration protocol, and
2. the network event protocol.

The global clock service is supported by the network event service, using a particular payload field. The payload contains information from the current grandmaster clock, including a current clock-time reading.

7.3.1 Partner clock calibration protocol

The partner clock calibration protocol is a symmetric protocol executed between two entities at opposite ends of a communication link. Each entity – a partner clock calibration entity – is associated with a single network port of an end device or a bridge. Each port operates in the context of a local timescale (device clock) as described in clause 7.1.

The purpose of the partner clock calibration protocol is for each entity to establish a relationship between its own timescale and the timescale of its partner entity. Based on this relationship, the partner clock calibration entity is able to convert between timescales; specifically it can translate a timestamp expressed in the distant timescale into a corresponding timestamp in its own timescale. This capability is expressed as a TTS (timestamp translation service – clause 7.3.1) interface to clients within the device protocol stack.

Typically a partner clock calibration entity will internally track its partner's device clock in terms of an *offset* term and a *rate* term, and will time-filter them to reduce noise. These terms describe the the performance of the distant clock by comparison to the local timescale only; there is no concept of a global timescale involved. It is possible to use more sophisticated clock models (e.g. those including a *rate drift* term), the result is simply to provide a more accurate timestamp translation service under certain application conditions.

The operation of the partner clock calibration entity has some similarity to a PLL (phase locked loop) slaved to the frequency and phase of the link partner's device clock. However a slave clock is only capable of answering the question "what time is it *now* on my partner's clock?", while the partner clock calibration entity answers questions of the form "what time was it when my partner's clock read 2:15?" The timestamp translation service describes the latter type of query, and the corresponding answer.

The optimum technique for performing partner clock calibration depends on the specific medium constituting the communications link and the application environment in which it operates. This standard defines one technique which must be supported on IEEE 802.3 network ports, and a second technique which must be supported on IEEE 802.11 network ports, in order for equipment to interoperate. The protocol defined for IEEE 802.3 ports is suitable when a full-duplex point-to-point link directly connects two devices implementing it. It is permissible for vendors to implement additional, perhaps proprietary techniques, and such implementations are assured to support the network event and global clock services, provided the corresponding protocol entities present a functionally compatible TTS client service meeting the error budget limits of clause 7.3.4.1.

7.3.1.1 Timestamp translation service

The TTS, or timestamp translation service, is an abstract internal service interface defined as an aid to modeling the interaction of the partner clock calibration protocol and the network event protocol. The TTS is not directly accessible to applications and there is no implication that actual device designs should or should not employ such an interface in their implementation.

The TTS interface is associated with a device port and exists in the context of a local timescale and a partner timescale. It consists of two primitives, TTS_partner_time.request and TTS_local_time.indication. Each primitive contains a single semantic element:

Device_Timestamp

Upon presentation of a TTS_partner_time.request primitive containing a device timestamp *partner-time*, the partner clock calibration entity will respond to its client with a TTS_local_time.indication primitive. The TTS_local_time.indication primitive will contain a timestamp *local-time* which is an estimate of the local device clock value corresponding to the same instant referenced by the *partner-time* value in the partner's timescale.

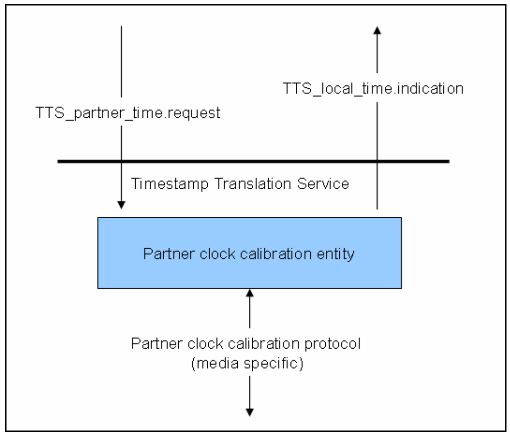

Figure 7-4.

7.3.1.2 Required 802.3 partner clock calibration protocol

The defined 802.3 partner clock calibration protocol is a symmetrical protocol, in which each link endpoint periodically transmits a LocalSync frame to the other. The transmission times of the two entities need not be synchronized; in fact the protocol operates correctly if the two partner entities transmit at substantially different rates.

For each LocalSync frame transmitted or received, the port registers a timestamp representing the instant at which the frame passed a particular reference point. The protocol assumes that the transit delay from the sender's reference point to the receiver's reference point is fixed (or very slowly varying), and that the transit delay is equal in both directions.

The measured timestamps at each link partner are shared with the other partner by means of LocalSync messages. On this basis, each partner can calculate an estimate of the other partner's clock behavior.

The LocalSync frame contains 7 fields:

Field	Size octets	Function
Source Address	6	Source MAC
Destination Address	6	Assigned link-local MAC
Sequence	1	Message sequence number
ET_1	6	Egress timestamp, message 1
IT_1	6	Ingress timestamp, message 1
ET_2	6	Egress timestamp, message 2

FCS	2	Frame Check Sequence
	33	

The timestamps in the message are reported in the context of the link partners' device timescales. At the issuer of the LocalSync frame, "message 1" refers to the most recently *received* LocalSync frame for which the issuer has both egress and ingress timestamps available. Being a received frame, the ingress timestamp for message 1 has been generated locally, while the egress timestamp is known by reception of an earlier LocalSync message. "Message 2" refers to the most recently *transmitted* LocalSync frame, i.e. the frame with a Sequence field one less than the current frame.

Upon receiving a LocalSync frame *i*, the peer clock calibration entity may confirm that the received sequence number has advanced by one (i.e. no packets were lost) and retrieve the previous ingress timestamp, which is associated with message 2 and is identified as IT_2. It may then compute as follows:

$$\textit{local-point}_i = (ET_1 + IT_2)/2$$

$$\textit{partner-point}_i = (ET_2 + IT_1)/2$$

These two timestamps represent the same instant in the local and partner device timescales, respectively, assuming symmetrical link propagation delay.

Using a previous LocalSync frame $i\text{-}N$, N steps prior to frame *i*, the peer clock calibration entity may compute the frequency ratio between its own device clock and its partner's:

$$\textit{rate}_i = (\textit{local-point}_i - \textit{local-point}_{i-N}) / (\textit{partner-point}_i - \textit{partner-point}_{i-N})$$

In order to reduce the effect of measurement jitter, the peer clock calibration entity will perform temporal low-pass filtering on the *local-point*, *partner-point*, and *rate* parameters. The timestamp translation service may then convert *partner-time* to *local-time* according to:

$$\textit{local-time} = (\textit{partner-time} - \textit{partner-point}) \cdot \textit{rate} + \textit{local-point}$$

As an example, a single-pole discrete-time IIR (infinite impulse response) low-pass filter with *z*-transform response

$$F(z) = 0.02 + 0.98 \cdot z^{-1}$$

may be applied identically to the sequences $\textit{local-point}_i$, $\textit{partner-point}_i$, and $\textit{rate}_i$.

<<need modeling to optimize coefficients for jitter/wander vs lag>>

<<we could also compute link delay, but it is not needed for this protocol. Also could make provision for a priori delay asymmetry correction e.g. with management parameter.>>

<<so far this is suitable for point to point. could probably be extended to work with shared media by paying attention to Source Address field>>

<<might be interesting to extend this with granularity and repetition-rate control fields: "my timestamp granularity is T", "please send at rate X", "I can't handle more than rate Y". This would allow "smart" devices to control error budget even when interoperating with "dumb" ones >>

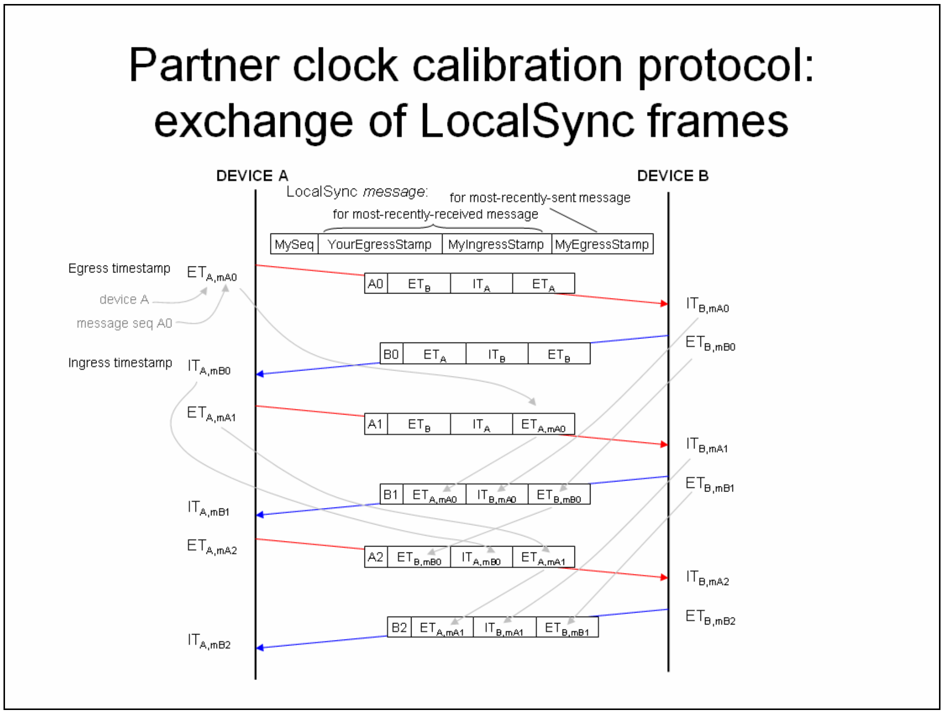

Figure 7-5.

7.3.1.3 Required 802.11v partner clock calibraton protocol

The 802.11 partner clock calibration protocol is based on the presence request messages defined in IEEE 802.11v.

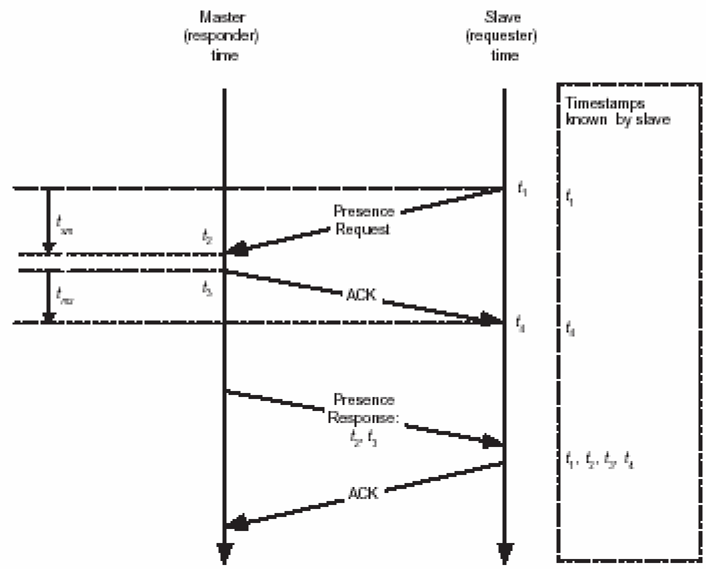

Figure 7-10—Synchronization message exchange for a master and slave that are directly connected via an 802.11 wireless link

The message exchange pattern is:

a) The requester sends a Presence Request message to the responder and notes the time, t_1, at which it was sent

b) The responder receives the Presence Request message and notes the time of reception, t_2

c) The responder returns an ACK message to the requester, and notes the time, t_3, at which it was sent

d) The requester receives the ACK message and notes the time of reception, t_4

e) The responder communicates the times t_2 and t_3 to the requester in a Presence Response message

f) The requester sends an ACK message to the responder to acknowledge that it received the Presence Response message

At the conclusion of this exchange of messages, the requester possesses all four timestamps (t_1, t_2, t_3, and t_4). The requester's peer clock calibration entity may then compute parameters for the current exchange *i* as follows:

$$local\text{-}point_i = (t_1 + t_4)/2$$

$$partner\text{-}point_i = (t_2 + t_3)/2$$

These two timestamps represent the same instant in the local and partner device timescales, respectively, assuming symmetrical link propagation delay.

Using a previous presence message exchange *i-N*, N steps prior to message *i*, the peer clock calibration entity may compute the frequency ratio between its own device clock and its partner's:

$$rate_i = (local\text{-}point_i - local\text{-}point_{i-N}) / (partner\text{-}point_i - partner\text{-}point_{i-N})$$

In order to reduce the effect of measurement jitter, the peer clock calibration entity will perform temporal low-pass filtering on the *local-point*, *partner-point*, and *rate* parameters. The timestamp translation service may then convert *partner-time* to *local-time* according to:

$$local\text{-}time = (partner\text{-}time - partner\text{-}point) \cdot rate + local\text{-}point$$

As an example, a single-pole discrete-time IIR (infinite impulse response) low-pass filter with *z*-transform response

$$F(z) = 0.02 + 0.98 \cdot z^{-1}$$

may be applied identically to the sequences $local\text{-}point_i$, $partner\text{-}point_i$, and $rate_i$.
<<need modeling to optimize coefficients for jitter/wander vs lag>>

The 802.11v presence message exchange is not symmetrical, so only the requester is able to perform these computations. The presence exchange is therefore executed periodically in both directions, with the link partners adopting alternative roles as requester and responder. By this means, both peer clock calibration entities are enabled to offer the timestamp translation service within their respective devices' protocol stacks.

7.3.2 Network event protocol

The network event protocol consists of NetworkEvent frames passed through the network from the issuer of the network event request primitive to the recipients of the network event indication primitive. The NetworkEvent frame consists of six fields:

Source Address
Destination Address
Ethertype/subtype
Event Timestamp
Payload
FCS (Frame Check Sequence)

The Event Timestamp field is valid in the context of the device transmitting the frame. A device receiving such a frame addressed to itself must translate this timestamp into its own device timescale before presenting the semantic content of the frame as a network event indication. The timestamp translation is provided by the partner clock calibration entity (clause 7.3.1).

A bridge device is responsible for forwarding network event frames to its attached ports in accordance with the model of IEEE 802.1D and IEEE 802.1Q specifications. However an 802.1AS bridge is also responsible for adjusting the Event Timestamp field in each outgoing NetworkEvent frame so that it is valid in the device timescale context of the bridge itself. This may be achieved by applying the timestamp translation service at the ingress port, then forwarding the corrected frames through the bridge fabric as specified by existing 802.1 standards.

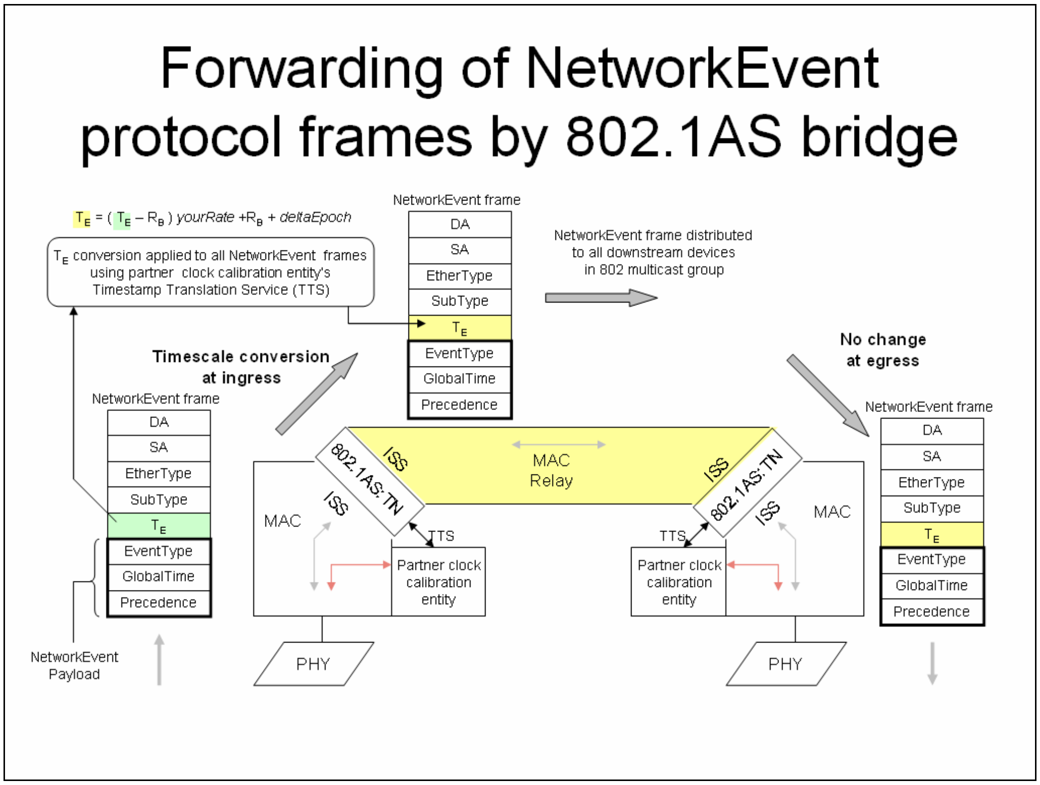

Figure 7-7

7.3.3 Global clock protocol

The global clock protocol is layered above the network event protocol, and uses a specific form of network event primitives to implement a grandmaster clock distribution system within a bridged local area network.

Selection of a single "best clock" to serve as grandmaster in a network is performed automatically on the basis of a Grandmaster precedence parameter assigned to each clock entity. The Grandmaster precedence parameter encodes accuracy and other features of the clock in such a way that a lower value of the parameter represents a superior choice as master clock.

7.3.3.1 Network event payload

The Global clock protocol defines a specific structure for the payload field of network event primitives exchanged in the protocol. That structure is:

Event Type = 0x00
Grandmaster precedence
Grandmaster time

7.3.3.2 Clock entity behavior

When a global clock entity receives a network event indication message, it processes the contents as follows:

1. If the Event Type does not have the assigned value 0x00, the message is ignored.
2. If the Grandmaster precedence field in the message is inferior to the clock entity's own, the message is ignored.
3. Otherwise (i.e. if the Grandmaster precedence field in the message is superior to the clock entity's own), the message is "qualified"; the clock entity is updated, treating the message as a cross time stamp affiliating the Event Time field of the message with the Grandmaster time field of the payload.

If a global clock entity does not receive any qualifying network event indications for a timeout period P (currently specified at 300msec), it initiates a random backoff period (currently specified 0-100msec). The backoff period is intended to minimize the threat of a "multicast storm" when a current grandmaster goes off-line. If the clock entity still has not received a qualifying network event at the expiration of the backoff period, it becomes grandmaster and begins to issue network event request primitives at regular intervals (currently specified at 100msec).

7.3.4 Selection of the protocol parameters

The two primary protocol parameters are

1. the repetition rate for peer clock calibraton protocol messages, and
2. the repetition rate for global clock update messages.

In combination with the device-dependent characteristics (i.e. timestamp granularity, clock phase noise, clock drift), these parameters determine the achievable system performance in terms of synchronization accuracy, jitter/wander, and settling time. Reaching the achievable performance also depends on the implementation of suitable temporal filters at the peer-clock and global-clock levels.

<<this outlines a design methodology; we need to run the numbers and simulate the results>>

7.3.4.1 Peer clock calibration parameters

The design methodology for this protocol is to assign an error budget to each peer clock calibration entity such that a 7-hop chain will accumulate a predictable maximum error; then global clock entities can be designed to meet particular application requirements based on this maximum error in their update message stream. Also other applications making use of the network event service experience predictable behavior. The peer clock error budget is expressed as an MTIE mask <<TBD>>. The design of the peer clock calibration message rate and filtering algorithm to meet this error budget will rely on the device-dependent characteristics noted above, viz. timestamp granularity, phase noise, clock drift. A standard message rate is associated with the 802.3 and 802.11v protocols defined in this standard <<TBD>>. Nonetheless it is permissible for vendor-specific protocols to deviate from this rate or to negotiate the rate dynamically depending on application requirements, e.g. for power-up initialization, transient recovery, or enhanced accuracy.

7.3.4.2 Global clock update operational parameters

The optimal repetition rate for global clock update messages is independent of the device and media characteristics, provided the error budget assigned at the timestamp translation service is met. Thus a single global clock update rate used throughout a mixed-link-type network is appropriate. The rate specified in this standard <<100msec TBD>> is intended to support media clock reconstruction to the jitter/wander requirements of consumer and professional audiovisual applications.

In some alternative time distribution protocols, increasing the update rate for comparable messages (e.g. 1588 Sync messages) is expected to improve accuracy and dynamic response. Each such message is associated with some timing uncertainty, dependent on the hardware implementation and operating environment; by averaging the effect a large number of messages over a short period the uncertainty can be reduced.
In contrast, in the peer calibration approach, the system timing uncertainty comes primarily from calibration errors in the timestamp translation model, and increasing the rate of global clock messages (i.e. repeating the translation operation more often) has little effect on the accuracy of time transfer.