Analysis of Clock Synchronization Approaches for Residential Ethernet

Geoffrey M. Garner (Consultant) Kees den Hollander
SAIT, Samsung Electronics
gmgarner@comcast.net, denhollander.c.j@samsung.com

Abstract

Residential Ethernet (ResE) is a new standardization activity in IEEE 802 that is considering extensions to Ethernet to allow the transport of time-sensitive traffic. Applications that ResE is expected to carry will include digital video, high-fidelity digital audio, and gaming traffic, as well as traditional non-time-sensitive traffic (e.g., data traffic). One goal of ResE is to allow a single network infrastructure in the residence to carry both time-sensitive and non-time-sensitive applications.

The Audio/Video (A/V) applications for ResE have tight jitter and wander requirements; in addition, applications where A/V content is delivered to multiple locations (e.g., gaming, digital stereo speakers) may have tight time synchronization requirements. To meet the jitter, wander, and time synchronization requirements for the applications, time synchronization must be provided to the ResE endpoints. Several approaches for providing time synchronization, all of which are based on principles used in IEEE 1588 and employ time stamps, are being considered. However, a number of algorithms for using the time stamp information have been suggested, e.g., instantaneous phase adjustments, instantaneous phase adjustments and less frequent instantaneous frequency adjustments, phase adjustments and possibly less frequent frequency adjustments with filtering, phase and frequency adjustments using phase-locked loops (PLLs), and use of transparent clocks. The jitter and wander performance and delivered time accuracy of each scheme will depend on the rate of phase and, if appropriate, frequency adjustments (which is limited by the rate at which time stamp information is transported), bandwidth and gain-peaking of the various filters and/or PLLs, quality of the node clocks (frequency tolerance, phase noise level and characteristics, etc.), and size of the network.

This paper develops simulation models to analyze several of the approaches (the remaining approaches will be considered in future work). Simulation results are obtained and compared with each other and with end-to-end application jitter and wander requirements. While this is initial work, the eventual goal is to provide input for deciding which approach is best for Residential Ethernet.

1. Introduction

Residential Ethernet (ResE) is a new standardization activity in IEEE 802 that is considering extensions to Ethernet to allow the transport of time-sensitive traffic. Applications that ResE is expected to carry will include digital video, high-fidelity digital audio, and gaming traffic, as well as traditional non-time-sensitive traffic (e.g., data traffic). One goal of ResE is to allow a single network infrastructure in the residence to carry both time-sensitive and non-time-sensitive applications. Ethernet is a ubiquitous and inexpensive network technology, and is ideally suited for this purpose.

Two key ResE requirements are (1) guaranteed QoS for time-sensitive applications, and (2) minimal administration by users. Guaranteed QoS means that application jitter, wander, time synchronization, and latency requirements are met. Minimal administration means that administration and provisioning should not be actively required on an ongoing basis. To meet these requirements, ResE will need features that include bandwidth reservation, admission control, and network synchronization. ResE also will use the priority class mechanism of current Ethernet. Finally, ResE rates will initially include 100 Mbit/s and 1 Gbit/s, with 10 Gbit/s added in the future.

The Audio/Video (A/V) applications for ResE have tight jitter and wander requirements. For example, compressed digital video delivered in the form of MPEG-2 packets has a peak-to-peak phase variation requirement that can range from 1 μs for the case where the video originates in the residence to some fraction of 50 μs for the case where the video is delivered by a service provider (the precise fraction depends on the jitter and wander budget allocations).

Digital audio has a peak-to-peak jitter requirement of 10 ns, measured with a 200 Hz measurement filter for consumer-grade interfaces. These jitter and wander requirements for A/V applications must be met by the A/V signals that are demapped from ResE packets and delivered to the respective codecs. In addition, applications where A/V content is delivered to multiple locations (e.g., gaming, digital stereo speakers) may have tight time synchronization requirements. To meet the jitter, wander, and time synchronization requirements for the applications, time synchronization must be provided to the ResE endpoints.

Several approaches are being considered for providing time synchronization, all of which are based on principles used in IEEE 1588 Precision Time Protocol (PTP) [1] and Network Time Protocol (NTP) [2], and employ time stamps. In most cases, the time-stamps are two-way and therefore analogous to the Sync and Delay_Req messages of IEEE 1588. However, a number of algorithms for using the time stamp information have been suggested, e.g., instantaneous phase adjustments, phase adjustments with filtering, instantaneous phase adjustments and less frequent instantaneous frequency adjustments, phase adjustments and less frequent frequency adjustments with filtering, phase and frequency adjustments using phase-locked loops (PLLs), and use of transparent clocks [3]. The jitter and wander performance and delivered time accuracy of each scheme will depend on the rate of phase and, if appropriate, frequency adjustments (which is limited by the rate at which time stamp information is transported), bandwidth and gain-peaking of the various filters and/or PLLs, quality of the node clocks (frequency tolerance, phase noise level and characteristics, etc.), and size of the network.

This paper develops a simulation model for the approaches based on instantaneous phase adjustments and instantaneous phase adjustments with less frequent instantaneous frequency adjustments, both with and without filtering. Simulation results are obtained for several cases that include the effects of node clock phase noise and time measurement granularity. The results are compared with the A/V application jitter and wander requirements. The present work is initial work; more detailed analysis of cases that cover ranges of parameters is planned for future work. The modeling and simulation of the other approaches (e.g., PLL filtering, transparent clocks) is also an area for future work.

The paper is organized as follows. Section 2 summarizes the jitter and wander requirements for A/V applications that are expected to be carried by ResE, and presents an example Hypothetical Reference Model (HRM) for the transport of compressed digital video. Section 3 describes the fundamental time stamp exchange process and summarizes the various approaches/schemes for using the time stamp information. Section 4 develops the simulation model for the three approaches considered here. Section 5 presents the simulation cases and results. Section 6 presents conclusions and indicates future work. The clock phase noise model used in the simulations is described in the Appendix.

2. Jitter and wander requirements for Residential Ethernet time-sensitive applications

Residential Ethernet will carry compressed digital video in the form of MPEG-2 or MPEG-4 and high-fidelity (e.g., CD quality) digital audio. In the future, ResE may carry uncompressed digital video, but this will likely require the use of 10 Gbit/s Ethernet to be practical. Note that MPEG-2 and MPEG-4 are not basic video formats; rather, they are formats for compressing and transporting uncompressed digital video that was originally produced by digitizing analog video. Detailed descriptions of digital video and digital audio applications are given in [4] and [5], respectively, and the jitter and wander requirements for these applications are discussed in [6]. The jitter and wander requirements are summarized below.

2.1. Video applications

Two classes of uncompressed digital encodings have been standardized: (1) direct sampling of a composite video signal, and (2) individual sampling of each component of a component video signal. Sampled composite NTSC and PAL signals with nominal rates of 143 Mbit/s and 177 Mbit/s, respectively, are defined in [7]. Sampled component video signals with nominal rates of 270 Mbit/s and 360 Mbit/s are defined in [7] and [8]. These 4 signals are standard definition TV (SDTV). Sampled High Definition TV (HDTV) component video signals with nominal rates of 1.485 Gbit/s and 1.485/1.001 Gbit/s are defined in [9]. Jitter, frequency offset, and frequency drift requirements for SDTV and HDTV signals are summarized in columns 2 and 3 of Table 1 [7], [9].

Compressed digital video is the initial focus for ResE because the uncompressed digital video rates are appreciable relative to or exceed the 100 Mbit/s and 1 Gbit/s Ethernet rates and the capacities of transport networks that deliver video to the residence. The MPEG-2 standard defines a scheme for compressing and packetizing digital video and audio into Packetized Elementary Streams (PESs) [10]. The PESs for a single program may be multiplexed into a Program Stream (PS), with multiple PES packets combined into larger PS packs. Alternatively, the PESs from multiple programs may be multiplexed into a Transport Stream (TS). In this case, each PES packet is mapped into one or more 188 byte TS packets (in general, the TS packet

is smaller than the PES packet). A PS contains a single program, and is therefore traceable to a single clock; it is typically used in a DVD application. A TS contains multiple programs that are traceable to different clocks and have been multiplexed. The TS multiplexing process includes a rate adjustment of the individual PESs to a TS Reference Clock.

The MPEG-2 packs or packets are mapped into Ethernet frames at the ResE ingress, transported over ResE, and demapped at the egress. However, the full reference model may also involve transport over one or more service provider networks (see Figure 1). The uncompressed digital video is encoded in MPEG-2 at the source and transported over several service provider networks. The PES and TS are created at Ref. Points B and C, respectively; note that the TS may contain additional PESs. The TS packets enter the residence, and an interworking function demaps them from the service provider transport protocol and maps them into Ethernet frames. The frames traverse the ResE, and the TS packets are demapped at Ref. Point D. The PES is then demultiplexed at Ref. Point E and input to the MPEG-2 decoder.

The accumulated jitter and wander at the ResE demapper (Ref. Point D) must be within the MPEG-2 decoder requirements, which are summarized in columns 4 and 5 of Table 1 [10]. Note that there are no jitter requirements; rather, there are requirements on peak-to-peak phase variation without any high-pass measurement filter. There actually are two requirements, one for the case where the MPEG-2 video is transported across a network, and the other for the case where there is no network transport. The former requirement applies for the case of transport over ResE and/or a service provider network to the residence (Ref. Point D and E). The latter requirement applies to Ref. Point B and C, whose jitter and wander are the same as that for Ref. Point E and D, respectively, if no networks are present.

Figure 1 also illustrates the use of ResE application time stamps to control the jitter and wander of the TS packets at the ResE egress. These time stamps are measured relative to the ResE synchronized network clocks. A time stamp is created for each TS packet when it is mapped into an Ethernet frame, relative to the network clock at the ingress; the time stamp is then used to determine when to deliver the TS packet to the MPEG-2 TS demux (Ref. Point D) using the synchronized network clock at the egress. Note that these time stamps are part of the MPEG-2 (or MPEG-4) TS to Ethernet adaptation function; they are distinct from the time stamps used within MPEG-2 or MPEG-4, and also from the IEEE 1588-like time stamps used by the ResE node clocks to for network synchronization.

2.2 High-fidelity audio applications

The primary audio applications that ResE will initially transport include digital audio that originates in the residence on a CD and analog audio that originates in the residence and is digitized prior to transport. The digital audio rate is controlled by the source clock or sampling clock. The digital audio signal is then transported to a receiver, where clock and data recovery are performed. The recovered clock is then filtered further to produce a sampling clock for D/A conversion. The additional filtering is needed because the jitter requirements for the sampling clock to produce high-fidelity audio are much more stringent than the requirements for data recovery.

The interface between the transmitter and receiver is standardized separately for consumer and professional applications [11]. The data is carried in 64-bit (128 Unit Interval (UI), with 2 UI/bit line coding) frames. The nominal basic frame rates are 44.1 kHz for consumer applications and 48 kHz for professional applications. In addition, double, quadruple, half, and quarter rates are defined. The highest rate is four times the 48 kHz rate, which corresponds to 24.576 Mbit/s (128 UI/frame).

In carrying digital audio over ResE, the audio frames are mapped into Ethernet frames at the transmitter, transported over ResE, and demapped at the receiver. The accumulated jitter and wander at the demapper must be within the requirements of the additional filter that produces the sampling clock for the D/A conversion. The requirements are summarized in columns 6 and 7 of Table 1 [11].

2.3 End-to-end application jitter and wander requirements expressed in terms of Maximum Time Interval Error

The requirements of Table 1 can be expressed conveniently in terms of Maximum Time Interval Error (MTIE). MTIE is peak-to-peak phase variation for an observation interval, expressed as a function of the interval length. The peak-to-peak is taken over all possible observation intervals of the given length in the measurement sample. The rigorous mathematical definition of MTIE is given in [12], along with the following expression that can be used to compute an estimate of MTIE from measured or simulated data

$$\mathrm{MTIE}(n\tau_0) = \max_{1\le k\le N-n}\left(\max_{k\le i\le k+n} x(i) - \min_{k\le i\le k+n} x(i)\right). \quad (1)$$

$n = 1, 2, ..., N-1$

Here, $x(i)$ is the i^{th} phase offset sample (out of N total samples), τ_0 is the sampling time, $n\tau_0$ is the observation interval, and $N\tau_0$ is the measurement interval.

MTIE requirements equivalent to the jitter and wander requirements of Table 1 are shown in Figure 2. MTIE for each A/V signal, at the input to the codec at the ResE egress, must not exceed the respective curve.

Detailed derivations of the masks are given in [6]. The requirement for MPEG-2 transported over networks (50 μs phase variation over approximately 600 s) is least stringent. The digital audio and uncompressed digital video jitter requirements are much more stringent (10 ns and less than 1 ns, respectively). The masks are network limits; for the case where the application is transported to the residence via service provider networks, ResE gets only an allocation of the total limit, with the remainder going to the service providers.

3. Approaches for providing ResE Synchronization

Several approaches, based on the exchange of time stamps between a master and slave clock, are being considered for ResE synchronization. The same principles are used in IEEE 1588 [1] and NTP [2]. It is envisioned that all the clocks in the ResE will be synchronized by one clock, termed the Grand Master. A slave clock will synchronize to a master which, in turn, will synchronize to its master; the synchronization chain will continue to the Grand Master.

The basic procedure for computing a correction to a slave clock is shown in Figure 3. The master and slave clocks are free-running with frequencies f_0 and f_1, respectively, and also initially indicate different times. In the k^{th} message exchange, the slave initially sends a message to the master that contains the time $T^S_{1,k}$ that the message is sent. The master notes the time it receives this message, $T^M_{2,k}$ and, at a later time, sends a message back to the slave containing the time the slave sent the first message, the time the master received that message, and the time $T^M_{3,k}$ that the master sends the current message. The slave notes the time it receives this message, $T^S_{4,k}$. Under the assumption that the propagation delays for the two messages are the same, the slave computes a correction u_k

$$u_k = \frac{\left(T^M_{2,k} - T^S_{1,k}\right) - \left(T^S_{4,k} - T^M_{3,k}\right)}{2}. \qquad (2)$$

This correction, if added to the current slave clock time, will synchronize the slave to the master clock. The error in the synchronization is of the order of the frequency difference between the master and slave multiplied by the duration of the message exchange.

[1] and [2] do not specify how the corrections should be used; a number of approaches have been suggested [17], and are summarized in Table 2. Since the master and slave clocks are free running with different frequencies, the message exchange must be repeated over time. In the simplest approach, the slave time is adjusted instantaneously at each message exchange. However, this adjustment results in an instantaneous phase step, which could result in exceeding some or all of the application jitter/wander requirements. Therefore, it is assumed that some form of filtering is performed at the network egress in all the approaches. The filtering may take the form of a digital filter running at the local egress free-running clock rate, or may be a full PLL whose frequency is adjusted based on filtered phase corrections.

The time stamp exchange described above is a two-way exchange; the master clock always responds to a message from the slave. However, the spatial extent of a typical ResE network is expected to be on the scale of a residence, with propagation delays stable and small (on the order of several μs) compared to the inter-message time (no smaller than 1 ms and likely an order of magnitude or more larger). Therefore, it is not necessary to always use a two-way time stamp exchange; instead, one-way time stamps could be used with less frequent two-way exchanges to obtain propagation delay as is done in IEEE 1588 [1] (Approach 1 in Table 2). The most recently computed propagation delay would be used by the slave when a one-way time stamp is received to compute the correction.

Approaches 2 – 5 in Table 2 concern the use of the clock corrections u_k, whether they are computed via one-way or two-way time stamps. First, the entire u_k value can be added to the current free-running slave clock time immediately to obtain the corrected time value, at each successive slave clock node (Approach 2). Second, the change in slave clock phase and master clock phase over some number of message intervals can be used to compute the frequency offset of the slave relative to the master (Approach 3). This offset can be used to obtain an improved estimate of slave clock phase, i.e., the phase the slave clock would have if it ran at the current estimate of the master clock rate. This improved slave clock phase estimate can then be used to obtain the slave clock correction u_k, which should be much smaller than the value that would be obtained if the frequency correction were not made. As in Approach 2, the u_k is added instantaneously to obtain corrected slave clock time. In Approach 4, the sequence of slave clock corrections u_k are filtered with a digital filter running at the local clock rate, and the filtered stream of corrections are added to the free running clock time. This has the effect of smoothing the instantaneous phase jumps that occur when the u_k are added directly. Note that whether or not filtering is done, the corrections must be accumulated on traversing the chain of slave clocks, because each u_k gives the correction needed to synchronize the slave to its upstream master, but what is actually desired is to synchronize the slave to the grandmaster at the beginning of the chain. If filtering is performed at the local clock rate and the filters at all nodes have the same properties (bandwidth, gain peaking, etc.), then the result is the same whether the

unfiltered corrections are accumulated and filtered at the egress or the filtered corrections are obtained at each node and accumulated, except for small differences due to the filters at the different nodes running at slightly different rates. This is because the filters are linear and run at nominally the same rate, and the u_k depend on unfiltered quantities. Finally, in Approach 5 the u_k are used in a PLL implementation to adjust the slave clock rate; this means that a particular u_k will affect future message times and future u_j ($j > k$). In this approach, placing a PLL at each intermediate node versus having a single PLL at the egress node will give different results.

Approach 5 results in a chain of PLLs. Such a scheme can produce acceptable jitter and wander accumulation with suitable PLL bandwidth, gain peaking, and noise generation (it is well-known that excessive gain peaking can result in large phase accumulation). However, depending on the actual requirements, this may result in the need for expensive oscillators. Conversely, it is not yet clear what the relative performance of Approaches 2 – 4 is relative to Approach 5. While the analysis of the present paper is limited to Approaches 2 – 4, all the approaches will be analyzed before a decision is made on which approach is most suitable for ResE.

Another scheme has been proposed [3] to reduce the need for long chains of PLLs. If synchronization is not needed at an intermediate node, it is possible to relay a message that traverses the node and measure the time the message resides in the node. This residence time may be variable, but as long as it is measured and relayed with the message, the eventual slave clock that receives the message can correct for this time. Such an intermediate node is termed a "transparent clock" [3]. If the message traverses several transparent clock nodes, the residence time can be accumulated. Since the transparent clock is free running, the correction will be in error by an amount equal to the frequency offset of the transparent clock multiplied by the residence time of the message in the node. It is possible to use Approaches 2 – 5 to adjust the phase and frequency of a transparent clock, but the same considerations indicated for slave clocks apply.

Depending on how the time stamp measurements are made (i.e., the measurements of the $T^S_{1,k}$, $T^M_{2,k}$, $T^M_{3,k}$, and $T^S_{4,k}$.), it may be much easier to make an accurate measurement if the value does not need to be sent in the current time stamp. In Approach 7, the time stamps reflect times delayed by some number of message exchanges. Finally, in Approach 8 the time stamps contain corrected slave clock times rather than the free-running, local clock values (Approach 8 is a modification of Approaches 2 – 4). This would avoid the need to accumulate the corrections, because the time stamps contain times traceable to the grandmaster.

The following sections develop a simulation model for Approaches 2 – 4 and consider several initial simulation cases. As indicated above, the other approaches will be analyzed in future work.

4. Simulation model for synchronization using Approaches 2 – 4

Figure 4 shows a chain of M+1 nodes, indexed from 0 to M. Node 0 is the grandmaster, and node i is the master of node i+1. The successive nodes exchange timestamp messages with each other, as indicated in Figure 3. Each node is timed by a clock that is free-running and has a level of phase noise. Let $y_b^i(t)$ and $n^i(t)$ be the fractional frequency offset expressed as a pure fraction and phase noise process (a random process) expressed in units of time, respectively, for the node i clock. Let $x_b^i(t)$ be the phase offset of clock i relative to UTC (ns). Then, assuming the frequency offset is constant over time and the phase offsets are zero at $t = 0$

$$x_b^i(i) = \int_0^t y_b^i(t)\,dt + n^i(t) = y_b^i t + n^i(t). \qquad (3)$$

We are primarily interested in the synchronization performance relative to the grandmaster, therefore, we can set $y_b^0(t) = n^0(t) = 0$.

Figure 5 shows the details of 3 successive message exchanges (exchanges k, k+1, and k+2) between node i+1 (slave) and node i (master). Let $T_{1,k}$' be the time the k^{th} message (i.e., part of the k^{th} message exchange) from the slave to the master leaves the slave, measured relative to the slave's free-running clock. Let $T_{1,k}$ be this same time measured relative to the master's free-running clock. The times $T_{j,k}$ and $T_{j,k}'$, for j = 2, 3, and 4, are defined similarly in Figure 5, with the primed quantities measured relative to the slave's free-running clock and the unprimed quantities measured relative to the master's free-running clock. In addition, let T_m be the nominal time between successive messages from master to slave, D the propagation delay between master and slave (it is assumed that the propagation delay is the same in both directions), and x the time offset between the receipt of the message from the slave by the master and the sending of the response message by the master to the slave. T_m, D, and x are expressed relative to UTC.

As indicated earlier, the propagation delay D is on the order of several μs for a residential network, and T_m will not be less than 1 ms. Then we may assume that $D << T_m$. The master to slave and slave to master message times are, in general, not synchronized, and x ranges from 0 to T_m. Therefore, the probability that messages from the master to slave and slave to master overlap in time is small, and is neglected here.

For each master/slave exchange, the quantity x will be initialized randomly. In addition, depending on whether the master and slave tie their message rates to their local free-running clocks or to their corrected times traceable to the grandmaster, x will vary with

time or remain constant, respectively. In the former case, the change in x over one inter-message time T_m is equal to the frequency offset between master and slave multiplied by T_m, i.e., $(y_b^{i+1} - y_b^i)T_m$. Depending on the relative signs and magnitudes of the frequency offsets, the change in x over T_m can be positive or negative and, over many inter-message times, x will reach T_m or 0, respectively. When this happens, a master to slave message "walks past" the next or previous slave to master message, and x jumps to 0 or T_m, respectively. It will be seen in the simulation results that this impacts the performance for the case where frequency adjustments are not made.

Let $x_{b,k}^i = x_b^i(kT_m)$, i.e., the phase offset of the free-running clock at node i at time kT_m, and $n_k^i = n^i(kT_m)$, i.e. the phase noise of the clock at node i at time kT_m. Then

$$x_{b,k}^i = y_k^i T_m k + n_k^i \,. \qquad (4)$$

The simulation model for Approaches 2 – 4 will compute the clock phase offsets at discrete times $t_k = kT_m$ (t_k is referred to as time step k). The following are needed:

1) Compute the phase offsets of the free-running node clocks at time step k using Eq. (4). The free-running clock frequency offsets are initialized randomly, from a uniform distribution within a frequency tolerance range, at initialization of the simulation. The clock noise is computed using the model described in the Appendix. Finite clock precision is modeled by truncating the phase offset to a granularity specified as input.

2) At each time step that is a multiple of the positive integer P, i.e., $t_{kP} = kPT_m$, compute an estimate of the frequency offset of each master clock i relative to its slave i+1 using the free-running clock phase offsets. P is specified as an input parameter to the simulation, and is the number of time-stamp intervals (inter-message times) over which the frequency offset is estimated.

3) At each time step kP, calculate the cumulative frequency offset estimate of the grandmaster relative to each slave clock i.

4) Calculate an improved phase offset for each slave clock i+1, using the free-running phase offsets and the most recent cumulative frequency offset estimate relative to the grandmaster.

5) Calculate the clock correction u_k^{i+1} for each slave clock i+1 in terms of the improved phase offsets for the slave clock and its master. Note that the time stamps must contain both the free-running clock times and the improved clock times (based on the computed improved phase offsets).

6) Calculate the cumulative clock correction for each slave clock i+1 relative to the grandmaster, and add this to the improved phase offset calculated in step (5). The result is the unfiltered phase offset for the slave clock.

7) Filter the sequence of unfiltered slave clock offsets with an appropriate digital filter. The result is the filtered phase offset for the slave clock.

If frequency adjustments are not made (Approach 2), then steps (2) – (4) are skipped, and the improved phase offset is set equal to the free-running clock phase offset in step (5). If filtering is not done, then step (7) is skipped and the result of step (6) is the desired clock phase offset.

Step (1) above is performed using Eq. (4) and the phase noise model of the Appendix, along with any specified clock granularity to model finite clock precision. Steps (2) – (7) are now described in detail.

Step 2

The estimate of frequency offset of clock i relative to clock i+1 is equal to the change in time indication of clock i minus the change in time indication of clock i+1, divided by the change in time indication of clock i+1. The changes in time indication are computed over the interval between times $(k-1)PT_m$ and kPT_m. The change in time indication for clock i over this interval is $PT_m + x_{b,kP}^i - x_{b,(k-1)P}^i$. The resulting estimate of frequency offset of clock i relative to clock i+1 is

$$\tilde{y}_{kP}^i = \frac{\left(x_{b,kP}^i - x_{b,(k-1)P}^i\right) - \left(x_{b,kP}^{i+1} - x_{b,(k-1)P}^{i+1}\right)}{PT_m + x_{b,kP}^{i+1} - x_{b,(k-1)P}^{i+1}}$$

$$\tilde{y}_{kP+1}^i = \tilde{y}_{kP+2}^i = \cdots = \tilde{y}_{kP+P-1}^i = \tilde{y}_{kP}^i \qquad , (5)$$

where $\tilde{y}_{kP}^i$ is the frequency offset estimate of clock i relative to clock i+1 at time step kP.

Step 3

The cumulative frequency offset estimate of the grandmaster relative to clock i at time step k, y_k^i, is

$$y_k^i = \sum_{j=1}^{i} \tilde{y}_k^j \,. \qquad (6)$$

Step 4

The improved phase offset estimate for clock i at time step j is calculated under the assumption that the frequency offset of the clock (relative to the grandmaster) is equal to y_j^i since the most recent time step kP when the frequency offset estimate was calculated. With this assumption, the change in corrected time reading of the clock divided by the change in free-running time reading of the clock is equal to one plus the frequency offset estimate, i.e.

$$\frac{(j-kP)T_m + x_j^i - x_{kP}^i}{(j-kP)T_m + x_{b,j}^i - x_{b,kP}^i} = 1 + y_k^i . \quad (7)$$

Then

$$x_j^i = x_{kP}^i + \left(x_{b,j}^i - x_{b,kP}^i\right)\left(1 + y_k^i\right) + (j-kP)T_m y_k^i . \quad (8)$$

Step 5

The clock i+1 correction, u_k^{i+1}, is calculated using Eq. (2), with the slave and master clock times replaced by the respective improved clock phase offsets. Unfortunately, Eq. (8) provides improved phase offset estimates only at the times the slave receives the response time stamps from the master (i.e., the $T_{4,k}$'). An exact calculation of the improved phase offsets at the other time stamp times ($T_{4,k}$', $T_{2,k}$, and $T_{3,k}$) would require implementing the model as a discrete event rather than discrete time (with constant time step) model. Since the latter is computationally much more efficient and therefore desirable, the improved phase offsets at the other times are obtained via interpolation. If the free-running clock frequency offset is the only component of the phase offset (i.e., if there is no phase noise), this approach is valid because the frequency offset gives rise to linear phase variation. If phase noise is present, the approach is only approximate. However, as will be seen in the Appendix, the phase noise model consists of White Phase Modulation (WPM, i.e., white noise) at high frequencies, Flicker Phase Modulation (FPM) at intermediate frequencies, and Flicker Frequency Modulation (FFM) at low frequencies. If the overall level of phase noise in the model is chosen to correspond to the actual noise level sampled at the time stamp rate, then interpolation will give approximately the correct noise level. The result is exact for WPM because it guarantees that the discrete WPM level in the model corresponds to the aliased WPM in the real system when sampled at the time stamp rate. The result is approximate for FPM and FFM because the interpolation corresponds to a filtering of these noise components.

Let r be the fraction of T_m by which the master to slave message is offset from the slave to master message, i.e. $r = x/T_m$, where x is the offset time defined in Figure 5. Also, assume that $D << T_m$ and $D << x$, and take the limit $D \to 0$. Finally, neglect the small probability that a master to slave and slave to master message may overlap in time. Then, replacing the time values by improved phase offset values in Eq. (2), which is permissible because the time values in Eq. (2) all appear as differences, and therefore the fixed (nominal) component of time cancels out

$$\begin{aligned} T'_{4,j} &= x_j^{i+1} \\ T_{3,j} &= x^i(jT_m - D) \cong x_j^i \\ T_{2,j} &= x^i(jT_m - D - x) \cong x^i(jT_m - x) \\ T'_{1,j} &= x^{i+1}(jT_m - 2D - x) \cong x^{i+1}(jT_m - x) \end{aligned} , \quad (9)$$

where $x^i(t)$ represents the improved phase offset for clock i expressed as a function of continuous time.

Finally, the $x^i(jT_m - x)$ are calculated by interpolation

$$\begin{aligned} x^i(jT_m - x) &= x_j^i - \frac{x_j^i - x_{j-1}^i}{T_m} x \\ &= x_j^i(1-r) + x_{j-1}^i r \end{aligned} . \quad (10)$$

As indicated earlier, the time offsets x, and therefore the fractions r, are initialized randomly (from a uniform distribution within [0,1] for r) at initialization of the simulation) and are either held constant during the simulation or change over each time interval T_m by T_m multiplied by the frequency offset between the two clocks.

Inserting Eqs. (9) and (10) into Eq. (2) produces the correction for clock i+1

$$\begin{aligned} u_j^{i+1} &= (1/2)\{x_j^i(1-r) + x_{j-1}^i r \\ &\quad -[x_j^{i+1}(1-r) + x_{j-1}^{i+1}] \\ &\quad -[x_j^{i+1} - x_j^i]\} \\ &= x_j^i\left(1 - \frac{r}{2}\right) - x_j^{i+1}\left(1 - \frac{r}{2}\right) + \frac{r}{2}\left(x_{j-1}^i - x_{j-1}^{i+1}\right) \end{aligned} . \quad (11)$$

Step 6

The cumulative correction for clock i relative to the grandmaster, $u_{k,cum}{}^i$, is

$$u_{k,cum}^i = \sum_{j=1}^{i} u_k^j . \quad (12)$$

The unfiltered phase offset is equal to $u_{k,cum}{}^i + x_k^i$.

Step 7

The simulations here use a discrete implementation of a standard second order filter with 20 dB/decade roll-off and gain peaking and 3 dB bandwidth specified on input. The details of the filter and discrete representation are given in Appendix VIII of [15] and in [16]. The standard form for this filter is

$$H(s) = \frac{2\varsigma\omega_n s + \omega_n^2}{s^2 + 2\varsigma\omega_n s + \omega_n^2}$$

ω_n = undamped natural frequency

ς = damping ratio

f_n = 3 dB bandwidth

$$= (\omega_n / 2\pi)\left[(2\varsigma^2 + 1) + \sqrt{(2\varsigma^2 + 1)^2 + 1}\right]$$

H_p = gain peaking

$$= \left[1 - 2\alpha - 2\alpha^2 + 2\alpha\sqrt{2\alpha + \alpha^2}\right]^{-1/2},$$

where $\alpha = 1/(4\varsigma^2)$

. (13)

The discrete implementation is based on representing the filter using state variables and writing the filter response as a convolution integral involving the input and state transition matrix. The state transition matrix is evaluated exactly, and the convolution integral over one time step is done analytically using a trapezoidal rule approximation. The filter time step can be taken as a sub-multiple of T_m. See[15] or [16] for details.

5. Simulation cases and results

This section presents the results for six simulation cases. Cases 1 and 2 assume no clock noise and infinite clock precision; Cases 3 – 6 assume clock noise in accordance with the model described in the Appendix and finite clock precision. These are initial cases; future work will consider (among other things) a wider range of parameters and the running of multiple, independent replications for each case to obtain statistical confidence intervals for the results.

Parameters common to all simulation cases are summarized in Table 3. The current HRM for ResE contains a maximum of 7 hops (i.e., 7 Ethernet switches; note that this is not finalized in the ResE Study Group in IEEE 802). The simulation cases here use 10 slave clocks; it was felt desirable to obtain the performance for a synchronization chain that slightly exceeds the ResE HRM. The ±100 ppm frequency tolerance for the free-running slave clocks is standard for Ethernet. The 10 Hz, 0.1 dB filter was felt to be achievable at reasonable cost, and therefore reasonable for initial simulations. The simulation time step was chosen to be a small fraction of the inter-message times (as will be seen shortly, 1 ms for Cases 1 and 2 and 10 ms for Cases 3 – 6) so that phase peaks would be accurately captured for the unfiltered phase variation.

5.1 Cases 1 and 2

Parameters specific to Cases 1 and 2 are given in Table 4. These cases are intended to show the basic performance obtained with (Case 2) and without (Case 1) instantaneous frequency adjustments when no phase noise is present and the clocks have infinite precision. The nominal time between successive time stamps is 1 ms, and the nominal time between successive frequency adjustments in Case 2 is 10 ms.

Figures 6(a) – (b) show unfiltered phase variation for nodes 1 and 10, respectively. Each plot shows the results for Case 1 and Case 2 superimposed. With no frequency adjustments (solid lines), a steady state sawtooth-like phase variation is reached in a time on the order of a few ms. However, the steady-state phase variation is on the order of 70 ns for node 1 and 16 ns for node 10. The phase variation is essentially the cumulative effect of all the frequency offsets of all the clocks over the time-stamp interval (1 ms). For Case 2, there is an initial large transient for the first 10 ms, i.e., the time over which the first frequency measurement is made. However, after the first frequency measurement, a steady-state is reached where the peak-to-peak phase variation is extremely small. While not evident on the scale of the plot, the stead-state peak-to-peak unfiltered phase variation for Case 2 is 0.07 ns.

Figures 6(c) – (d) show the filtered phase variation for Cases 1 – 2. As before, there is an initial transient; however, the time for the transient to decay is now a few filter time constants (1/[2π(10 Hz)] = 16 ms). The steady-state peak-to-peak phase variation is approximately a few tenths of a ns without frequency adjustments and less than 1 ps with frequency adjustments. The extremely good synchronization performance when frequency adjustments are made is due to the fact that, when there is no phase noise and the clocks have infinite precision (zero granularity in phase), the frequency different between master and slave can be measured extremely accurately.

Note that, for a single simulation case, the phase variation does not necessarily increase monotonically as the number of nodes increases; this is because the phase variation depends on the magnitudes and signs of the frequency offsets of the successive clocks. However, if a large number of replications of a simulation case are run and point estimates and confidence intervals for a specific quantile of peak-to-peak phase variation are obtained, it is expected that the quantile point estimate will tend to increase at least over the first few nodes. This will be investigated in more detail in future work.

5.2 Cases 3 and 4

Parameters specific to Cases 3 and 4 are given in Table 4. For these cases, the nominal time between successive time stamps is 10 ms, and the nominal time between successive frequency adjustments in Case 4 is 100 ms (both values are suggested in [4]). Clock phase noise is modeled as described in the Appendix, and the clock phase precision is 1 ns. The time offset

between master to slave and slave to master messages is initialized randomly (between 0 ant T_m) for each node and held constant through the simulation.

Results for filtered and unfiltered phase offset for nodes 1 and 10 are shown in Figures 7(a) – (d). In each figure, the corresponding results for Case 3 (no frequency adjustments) and Case 4 (frequency adjustments) are superimposed. In all cases (i.e., with and without filtering, for both nodes) the steady-state peak-to-peak phase variation for the case with frequency adjustments is much smaller than for the case without frequency adjustments. The peak-to-peak variation of the former is not discernable on the scale of the latter in these figures; therefore, more detailed views of Case 4 are shown in Figures 8(a) – (d). The effect of the noise and granularity is seen here, though the filtered and unfiltered peak-to-peak phase variations are on the order of 1 ns and 2 ns, respectively.

MTIE results for unfiltered and filtered phase variation for selected nodes are shown in Figures 9(b) – (c) (Case 3) and 9(d) – (e) (Case 4), along with the MTIE requirements of Figure 2 (the legend for these plots is given in Figure 9(a)). It is seen that with filtering, MTIE is more than an order of magnitude smaller when frequency adjustments are made than when they are not. For longer observation intervals, the former ranges from 1 – 1.5 ns, while the latter from 10 – 50 ns. For the case with frequency adjustments, the uncompressed digital video MTIE masks are slightly exceeded; without frequency adjustments, these masks and the consumer interface digital audio MTIE mask are exceeded (Figures 9(c) and 9(e)). Without filtering, MTIE ranges from approximately 160 – 600 ns without frequency adjustments and 2 – 4 ns with frequency adjustments. In the latter case, the uncompressed digital video MTIE masks are exceeded; in the former case these and the digital audio masks are exceeded.

As in Cases 1 and 2, the phase variation does not increase monotonically with the number of nodes but, as mentioned in the discussion of those cases, the behavior of MTIE point estimates and confidence intervals as the number of nodes increases must be considered.

5.2 Cases 5 and 6

Parameters specific to Cases 5 and 6 are given in Table 4. The parameters of these cases are the same as in Cases 3 and 4, except that now the time offset between master to slave and slave to master messages is initialized randomly (between 0 ant T_m) for each node and allowed to vary as the simulation proceeds (instead of being held constant). As described in Section 4, this time offset for a master/slave message exchange changes by the frequency offset between the master and slave multiplied by T_m; when the offset reaches either 0 or T_m, it is set to T_m or 0, respectively.

Results for filtered and unfiltered phase offset for nodes 1 and 10 for Case 5 (no frequency adjustments) are shown in Figures 10(a) – (d). When frequency adjustments are not made, the peak-to-peak phase variation can be very large due to phase steps that occur when the master to slave and slave to master messages "walk" past each other, i.e., when the time offset between these messages reaches either 0 or T_m and is adjusted up or down, respectively, by T_m. Comparing Figure 10(a) with 10(c) and Figure 10(b) with 10(d) indicates that the 10 Hz filter greatly attenuates the fast sawtooth due to clock corrections, but cannot reduce the wander due to the steps in master to slave and slave to master message offset time, as the frequency of these steps is very small compared to 10 Hz. In the best case, where the master and slave clock differ by 200 ppm (i.e., one is +100 ppm from nominal and the other -100 ppm from nominal, the offset between the master to slave and slave to master messages changes by $(2 \times 10^{-4})T_m$ over each inter-message time T_m. It therefore requires $T_m/[(2 \times 10^{-4})T_m] = 5000$ inter-message times for a phase step to occur. The phase step frequency is therefore $1/[5000T_m] = 0.02$ Hz, which is very small compared to the 10 Hz filter bandwidth. The peak-to-peak phase variation is on the order of hundreds of ns, with and without filtering.

Results for filtered and unfiltered phase offset for nodes 1 and 10 for Case 6 (with frequency adjustments) are shown in Figures 11(a) – (d). When frequency adjustments are made, the above phase steps do not occur because the error in phase correction due to frequency offset between the master and slave is corrected for. The results for Case 6 are very similar to those for Case 4, where the time offset between master to slave and slave to master messages is fixed throughout the simulation. MTIE results for Cases 5 and 6 are shown in Figures 12(b)-(e) (the legend for the plots is in Figure 12(a)). MTIE for Case 6, for both filtered and unfiltered phase, is very similar to Case 4 (Figures 9(d) and 9(e)). The filtered phase variation MTIE result for Case 6 slightly exceeds the uncompressed digital video mask but meets the other masks; the unfiltered phase variation MTIE result for Case 6 exceeds the uncompressed digital video mask by somewhat more but meets the other masks.

6. Conclusions and future work

In the ideal case of no clock noise and infinite clock precision (i.e., zero granularity), the peak-to-peak phase variation is extremely small. The sawtooth frequency is sufficiently large compared to the 10 Hz filter bandwidth that peak-to-peak phase variation of one or a few tenths of a ns can be achieved even if frequency adjustments are not made; if frequency

adjustments are made, the peak-to-peak phase variation can be less than 1 ps.

For the non-ideal case, with clock noise and finite precision (i.e., noise at the level of the model in the model in the Appendix and 1 ns granularity), the results indicate that filtering is necessary for the uncompressed digital video MTIE masks to be met. The masks are slightly exceeded with the 10 Hz, 0.1 dB gain peaking filter if frequency adjustments are made. In addition, these masks and the digital audio consumer interface mask are exceeded if filtering is done and frequency adjustments are not made. This indicates that the filter bandwidth will likely need to be somewhat narrower. Finally, if the master to slave and slave to master messages are sent at rates traceable to the free-running node clocks rather than at rates traceable to the same clock, the results indicate that peak-to-peak phase variation can be very large (i.e., hundreds of ns) if frequency adjustments are not made.

Additional work is planned to examine the effect of varying the various parameters of the model:

- Filter bandwidth (and possibly gain peaking)
- Time between messages
- Time between frequency adjustments
- Larger clock noise level, which bounds noise in oscillators expected to be used in ResE
- Clock precision (phase granularity)
- Consideration of errors in measurements of times the time stamps are sent and received (modeling cases where the measurements are implemented in the Ethernet PHY, MAC, and above the MAC)

In addition, future simulation cases will have multiple, independent replications run, i.e., at the completion of a run, the state of the random number generator will be saved and used to initialize the next run. This plus the use of a random number generator with a sufficiently large period will ensure that the multiple runs are independent (at least insofar as the pseudo-random samples are independent, i.e., satisfy the necessary statistical tests). The multiple runs will enable confidence intervals for MTIE and other statistics to be obtained.

Finally, additional work is planned to investigate the other approaches in Table 2.

7. References

Note: References [4] – [6], [13], and [14] are available at http://www.ieee802.org/3/re_study/public/index.html.

[1] IEEE Std™ 1588-2002, *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE, November 8, 2002.

[2] Mills, L. David, *A Kernel Model for Precision Timekeeping*, RFC 1589, March, 1994.

[3] Sven Nylund and Oyvind Holmeide, *IEEE 1588 Switch Transparency – No Need for Boundary Clocks!*, 2004 Conference on IEEE 1588, Gaithersburg, MD.

[4] Geoffrey M. Garner, *Description of ResE Video Applications and Requirements*, IEEE 802.3 ResE presentation, Austin, TX, May, 2005.

[5] Geoffrey M. Garner, *Description of ResE Audio Applications and Requirements*, IEEE 802.3 ResE presentation, Austin, TX, May, 2005.

[6] Geoffrey M. Garner, *End-to-End Jitter and Wander Requirements for ResE Applications*, IEEE 802.3 ResE presentation, Austin, TX, May, 2005.

[7] SMPTE 259M-1997, *10-Bit 4:2:2 Component and $4f_{SC}$ Composite Digital Signals – Serial Digital Interface*, 1997.

[8] ITU-R BT.601-5, *Studio Encoding Parameters of Digital Television for Standard 4:3 and Wide Screen 16:9 Aspect Ratios*, Geneva, 1995.

[9] SMPTE 292M-1998, *Bit-Serial Digital Interface for High-Definition Television Systems*, 1998.

[10] ISO/IEC 13818-1, -2, -3, -9, *Information Technology – Generic Coding of Moving Pictures and Associated Audio Information: Systems (Part 1, 2000), Video (Part 2, 2000), Audio (Part 3, 1996), Extension for Real-Time Interface for Systems Decoders (Part 9, 1996), Geneva.*

[11] IEC-60958-1, -3, -4, *Digital Audio Interface – Part 1: General (2004), - Part 3: Consumer Applications (2003), - Part 4: Professional Applications (2003)*, Geneva.

[12] ITU-T Rec. G.810, *Definitions and Terminology for Synchronization Networks*, Geneva, 1996.

[13] Geoffrey M. Garner, *Summary of Potential Choices for Synchronization Transport in Residential Ethernet*, IEEE 802.3 ResE presentation, San Francisco, CA, July, 2005.

[14] *Residential Ethernet (RE) (a working paper)*, Draft 0.136, maintained by David V. James and based on work by him and other contributors, August 10, 2005.

[15] ITU-T Rec. G.8251, *The Control of Jitter and Wander Within the Optical Transport Network (OTN)*, Geneva, November, 2001, Amendment 1, June, 2002, Corrigendum 1, June, 2002.

[16] Geoffrey M. Garner, *Jitter Analysis for Asynchronous Mapping of a Client Signal into an OcH*, Lucent Technologies Contribution to ITU-T Q 11/15 Interim Meeting, Ottawa, ON, July, 2000.

[17] *Phase Noise*, Vectron International, available at http://www.vectron.com

[18] *Jitter and Signal Noise in Frequency Sources*, Raltron Application Note, available at http://www.raltron.com.

[19] Dan H. Wolaver, *Phase-Locked Loop Circuit Design*, Prentice-Hall, 1991, Chapter 6.

[20] David W. Allan, Marc A. Weiss, and James L. Jespersen, *A Frequency Domain View of Time Domain Characterization of Clocks andTime and Frequency Distribution Systems*, Forty-Fifth Annual Symposium of Frequency Control, Los Angeles, CA, May 29 – 31, 1991, pp. 667 – 678.

[21] Stefano Bregni, *Synchronization of Digital Telecommunications Networks*, Wiley, 2002.

[22] J.A. Barnes and Stephen Jarvis, *Efficient Numerical and Analog Modeling of Flicker Noise Processes*, National Bureau of Standards, NBS Technical Note 604, June, 1971.

[23] James A. Barnes and Charles A. Greenhall, *Large Sample Simulation of Flicker Noise*, 19th Annual Precision

Time and Time Interval (PTTI) Applications Planning Meeting, December, 1987.
[24] Giovanni Corsini and Robert Saletti, *A* $1/f^{\prime}$ *Power Spectrum Noise Sequence Generator*, IEEE Transactions on Instrumentation and Measurement, Vol. 37, No. 4, December, 1988, pp. 615 – 619.

Appendix. Clock noise model

Clock phase noise may be modeled as a sum of random processes with power spectral density (PSD) of the form

$$S(f) = \frac{A}{f^{\alpha}}, \qquad \text{(A-1)}$$

where A is a constant and α is typically a small integer. In practice, the PSD has 3 terms [17] – [19]

- $\alpha = 0$, White Phase Modulation (WPM), i.e., white noise
- $\alpha = 1$, Flicker Phase Modulation (FPM)
- $\alpha = 3$ Flicker Frequency Modulation (FFM)

The PSD of the phase noise can be written

$$S_x(f) = \frac{A}{f^3} + \frac{B}{f} + C, \qquad \text{(A-2)}$$

where $S_x(f)$ has units ns^2/Hz. (Note that if a term with $\alpha = 2$ were present, it would be referred to as White Frequency Modulation (WFM), i.e., a random walk in phase; however, a WFM component is not included in the model here.)

Often an alternate form of the PSD is used

$$S_{\phi}(f) = (2\pi\nu_0)^2 S_x(f), \qquad \text{(A-3)}$$

where $S_{\phi}(f)$ has units rad^2/Hz. An example specification, taken from Figure 12 of [18], is shown in Figure A-1. Note that the data in [18] is given for a two-sided PSD in units of dBc/Hz (decibels relative to the carrier); the data is converted to the above one-sided PSD expressed in rad^2/Hz using the conversion

$$S(f)\big|_{2-\text{sided}}(\text{dBc/Hz}) = \frac{1}{2} S(f)\big|_{1-\text{sided}}(\text{rad}^2/\text{Hz}), \qquad \text{(A-4)}$$

Note that the data in [18] does not extend to frequencies above 10 kHz; it is conservatively assumed here that the PSD does not decrease further for higher frequencies, and instead remains flat at the 10 kHz value (i.e., the WPM region is assumed to begin at 10 kHz). Finally, the data in [18] has the piecewise linear (i.e., linear on a log scale) form given by the dotted curve; this is conservatively approximated by the solid curve, which is a fit if Eq. (A-2). The specifications provided for specific devices of [17] and [17] are below the example PSD of Figure 12 of [18] (i.e., the data that gives rise to Figure A-1 here), at least for devices for which phase noise specifications are provided.

Another measure for clock phase noise is Time Variance (TVAR), along with its square root, Time Deviation (TDEV). TVAR and TDEV are more convenient than PSD because they are time domain parameters whose estimates are easily calculated from measured or simulated phase data. TVAR is defined as 1/6 times the expectation of the square of the second difference of the phase error averaged over an interval [12]

$$\text{TVAR}(\tau) = \frac{1}{6} E\left[\left(\Delta^2 \bar{x}\right)^2\right], \qquad \text{(A-5)}$$

where $E[\cdot]$ denotes expectation, $\bar{x}$ denotes average over the integration time τ, and Δ^2 denotes second difference. TVAR may be estimated from measured or simulated data using [12]

$$\text{TVAR}(n\tau_0) = \frac{1}{6n^2(N-3n+1)} \cdot \sum_{j=1}^{N-3n+1}\left[\sum_{i=j}^{n+j-1}\left(x_{i+2n} - 2x_{i+n} + x_i\right)\right]^2, \quad n = 1, 2, ..., \text{integer part}(N/3). \qquad \text{(A-6)}$$

where τ_0 is the sampling interval and $\tau = n\tau_0$. TVAR is equal to $\tau^2/3$ multiplied by the Modified Allan Variance [12].

For power-law noises with PSD proportional to $1/f^{\alpha}$, TVAR is proportional to τ^{β}, with $\beta = \alpha - 1$ [12], [20], [21]. In addition, the magnitude of TVAR may be related to the magnitude of PSD for power-law noises [20], [21]

FFM

$$S_x(f) = \frac{A}{f^3} \qquad \text{TVAR}(\tau) = \frac{(2\pi)^2 9 \ln 2}{20} A\tau^2 \qquad \text{(A-7)}$$

FPM (result is from [20]; a more exact expression is given in [21])

$$S_x(f) = \frac{B}{f} \quad \text{TVAR}(\tau) = \frac{3.37}{3} B \quad \text{(A-8)}$$

WPM

$$S_x(f) = C \quad \text{TVAR}(\tau) = \frac{\tau_0 f_h}{\tau} C \quad \text{(A-9)}$$

$$f_h = \text{noise bandwidth}$$

The above expressions are used to obtain TDEV(τ) that is equivalent to the PSD of Figure A-1 (and represented by an expression of the form of Eq. (A-2). Each noise component is simulated separately and added to obtain the total phase noise. To verify that the noise model matches the PSD, TDEV for the simulated noise samples is evaluated and compared with the TDEV mask equivalent to the PSD. We now briefly explain how each noise component is simulated.

WPM is simulated as a sequence of independent, identically distributed random samples. The noise distribution is taken as Gaussian with zero mean. The noise variance and sampling time determine the TDEV level. The variance is chosen such that, with the given sampling time, the computed TDEV from a sample history matches the value obtained from Eq. (A-9) above. The noise bandwidth is assumed to be equal to the line rate, i.e., 100 MHz.

FPM is simulated by passing a sequence of independent, identically distributed, Gaussian random samples through a Barnes/Jarvis filter [22] – [24]. If white noise is input to a filter with frequency response $H(f) = f^{-1/2}$, the output is a random process with PSD proportional to $1/f$, i.e., FPM. The Barnes/Jarvis filter approximates an $f^{-1/2}$ frequency response using a bank of lead/lag filters. The actual frequency response of this filter resembles a "staircase." The spacing of the poles and zeros are chosen such that the average slope is –10 dB/decade. The accuracy obtained depends on the number of poles and zeros per decade of frequency; see [24] for details. The variance of the Gaussian random samples input to the filter determines the TDEV level; the variance is chosen such that the computed TDEV from the resulting filter output matches the value obtained from Eq. (A-8) above.

FFM is simulated by passing a sequence of independent, identically distributed, Gaussian random samples (with zero mean) through a Barnes/Jarvis filter followed by an integrator (accumulator). As with FPM, the variance of the Gaussian distribution determines the TDEV level. The variance is chosen such that the computed TDEV from the resulting filter output matches the value obtained from Eq. (A-7) above.

TDEV for a sample history of simulated data (using the above modeling procedures) and the TDEV mask equivalent to the PSD of Figure A-1 are shown in Figure A-2. The time step for the simulation is 0.01 ms (the same as in the simulation cases described in Section 5).

Table 1. End-to-end jitter and wander requirements for A/V applications

Requirement	Uncompressed SDTV	Uncompressed HDTV	MPEG-2, with network transport	MPEG-2, no network transport	Digital audio, consumer interface	Digital audio, professional interface
Wide-band jitter (UIpp)	0.2	1.0	50 µs peak-to-peak phase variation requirement (no measurement filter specified)	1000 ns peak-to-peak phase variation requirement (no measurement filter specified)	0.25	0.25
Wide-band jitter meas filt (Hz)	10	10			200	8000
High-band jitter (UIpp)	0.2	0.2			0.2	No requirement
High-band jitter meas filt (kHz)	1	100			400 (approx)	No requirement
Frequency offset (ppm)	±2.79365 (NTSC) ±0.225549 (PAL)	±10	±30	±30	±50 (Level 1) ±1000 (Level 2)	±1 (Grade 1) ±10 (Grade 2)
Frequency drift rate (ppm/s)	0.027937 (NTSC) 0.0225549 (PAL)	No requirement	0.000278	0.000278	No requirement	No requirement

Table 2. Summary of approaches for using slave clock corrections in performing synchronization

	Approach
1	Use one-way time stamp scheme with less frequent two-way exchange; obtain delay from two-way exchange and assume delay is fixed until next two-way exchange
2	Instantaneous phase adjustments at intermediate nodes
3	Instantaneous phase and frequency adjustments at intermediate nodes, with frequency adjustments possibly less frequent [4]
4	Filtered phase adjustments at intermediate nodes, using digital filter running at local clock rate (with or without instantaneous frequency adjustments)
5	Full phase-locked loops (PLLs) at intermediate nodes
6	Use of transparent clocks [3] a) end-to-end versus peer-to-peer b) whether or not to adjust rate of local oscillator in transparent clock and, if so, whether to do filtering
7	Time stamp reflecting current time versus delay by some number of frames
8	Time stamp reflects local free-running clock time versus latest corrected time based on most recent time stamps and possible filtering

Table 3. Parameters common to all simulation cases

Common Parameters
10 hops (grandmaster followed by chain of 10 slave clocks)
Slave clock frequency tolerance = ± 100 ppm
Filter bandwidth = 10 Hz
Filter gain peaking = 0.1 dB
Simulation time step = 0.01 ms (used small time step to ensure phase peaks were captured)

Table 4. Parameters specific to each simulation case

Case	Include phase noise	Clock precision (ns)	Instant-aneous freq-uency adjust-ments	Time offset between M/S and S/M messages	Inter-message time (ms)	Time between frequency updates (ms)
1	**No**	**0**	**No**	**Fixed**	**1**	**–**
2	**No**	**0**	**Yes**	**Fixed**	**1**	**10**
3	**Yes**	**1**	**No**	**Fixed**	**10**	**–**
4	**Yes**	**1**	**Yes**	**Fixed**	**10**	**100**
5	**Yes**	**1**	**No**	**Vary-ing**	**10**	**–**
6	**Yes**	**1**	**Yes**	**Vary-ing**	**10**	**100**

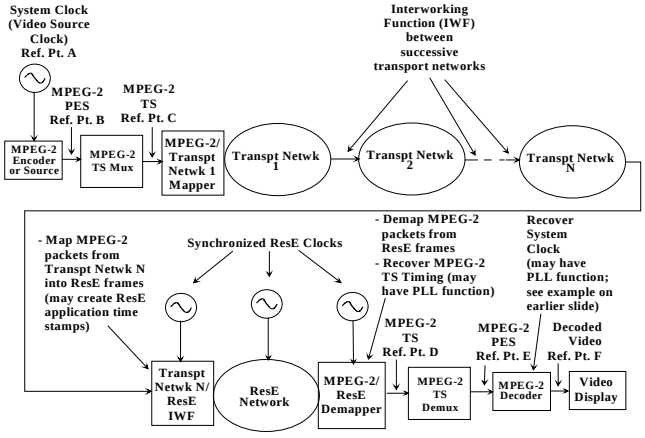

Fig. 1. Reference model for transport of MPEG-2 video over service provider networks and Residential Ethernet

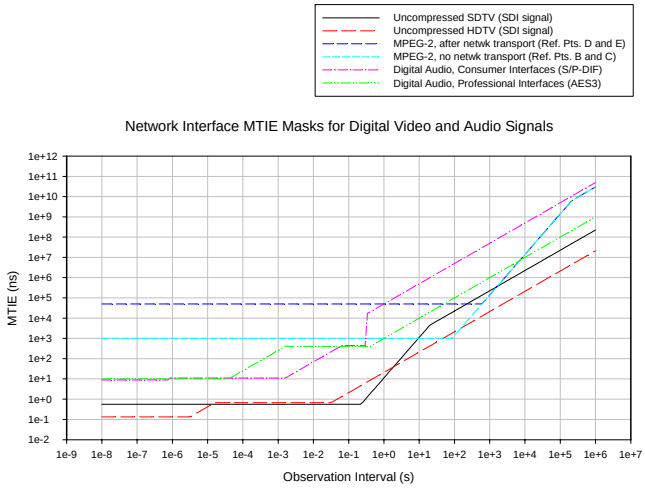

Fig. 2. Jitter and wander MTIE masks for digital video and audio applications (Reference Points are defined in Fig. 1)

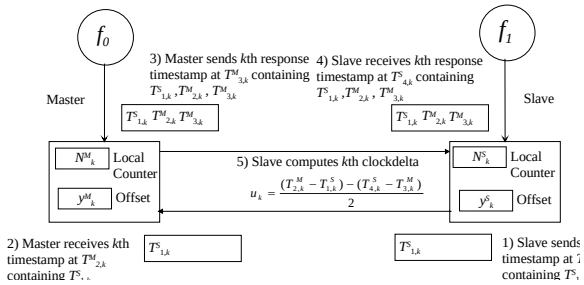

Fig. 3. Illustration of time stamp exchange between master and slave clock and computation of correction for slave

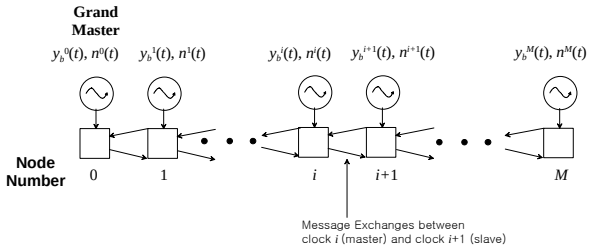

Fig. 4. Model for synchronization of a chain of slave clocks, using Approaches 2 – 4 of Table 2

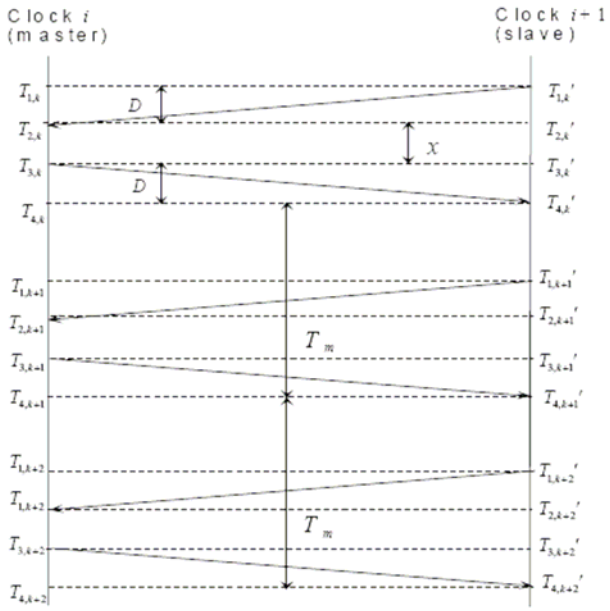

Fig. 5. Details of 3 successive message exchanges between a slave and master clock

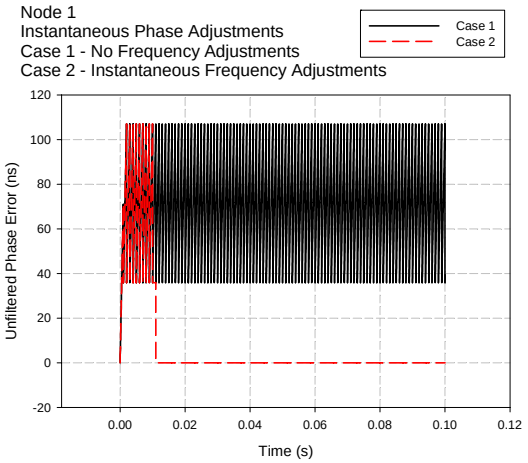

Fig. 6(a). Unfiltered phase offset for Cases 1 and 2, node 1

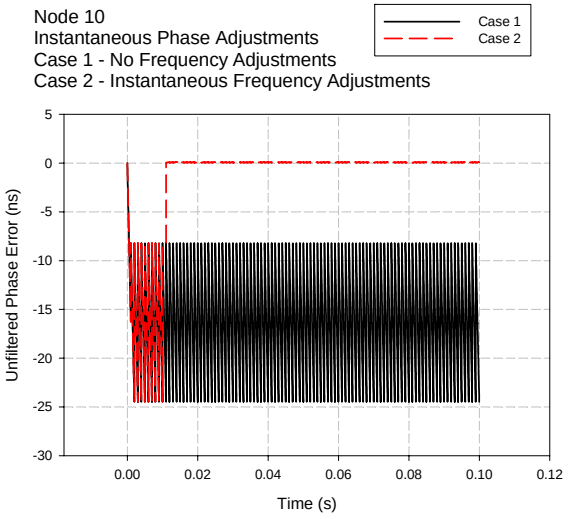

Fig. 6(b). Unfiltered phase offset for Cases 1 and 2, node 10

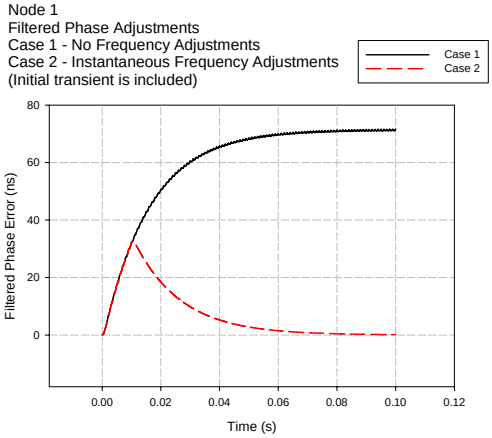

Fig. 6(c). Filtered phase offset for Cases 1 and 2, node 1

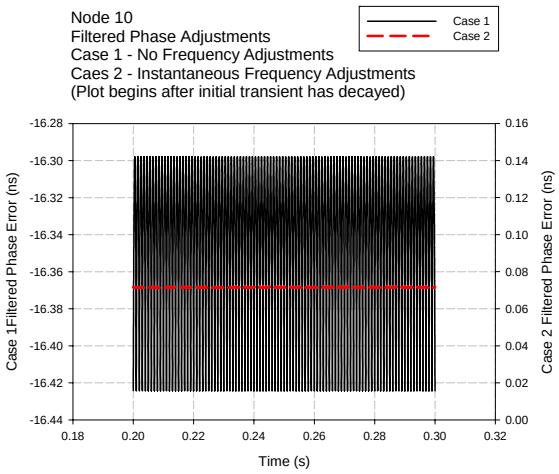

Fig. 6(d). Filtered phase offset for Cases 1 and 2, node 10

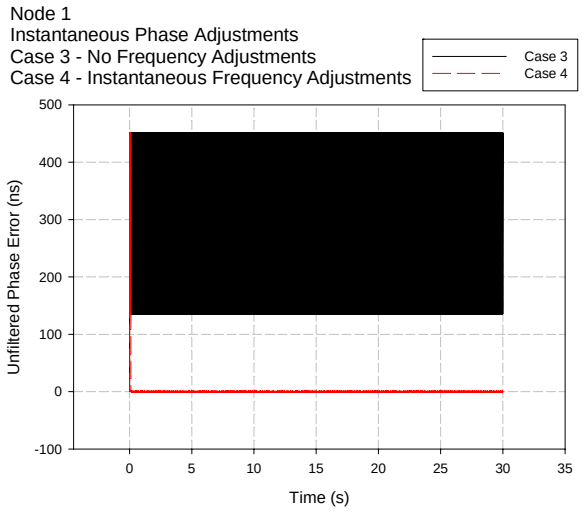

Fig. 7(a). Unfiltered phase offset for Cases 3 and 4, node 1

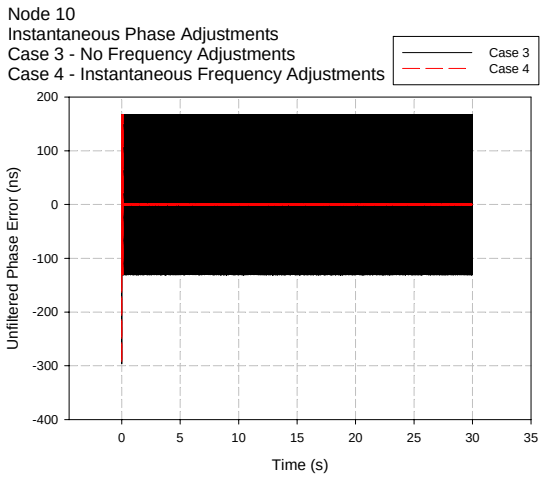

Fig. 7(b). Unfiltered phase offset for Cases 3 and 4, node 10

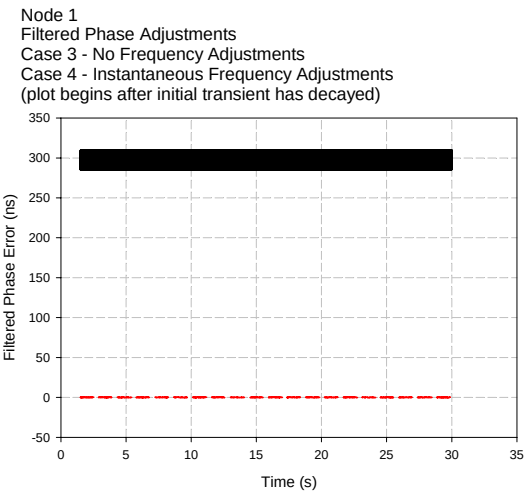

Fig. 7(c). Filtered phase offset for Cases 3 and 4, node 1

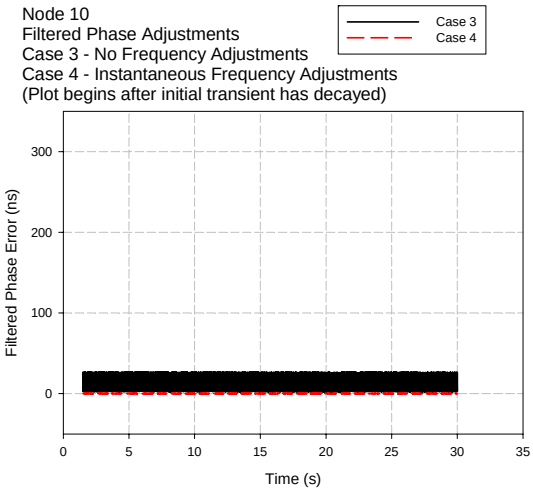

Fig. 7(d). Filtered phase offset for Cases 3 and 4, node 10

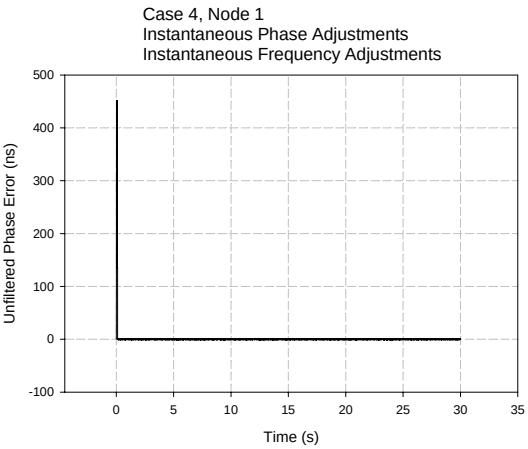

Fig. 8(a). Detailed view of unfiltered phase offset for Case 4, node 1

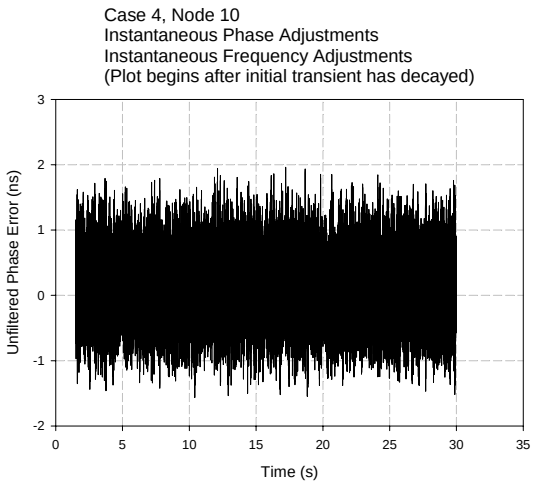

Fig. 8(b). Detailed view of unfiltered phase offset for Case 4, node 10

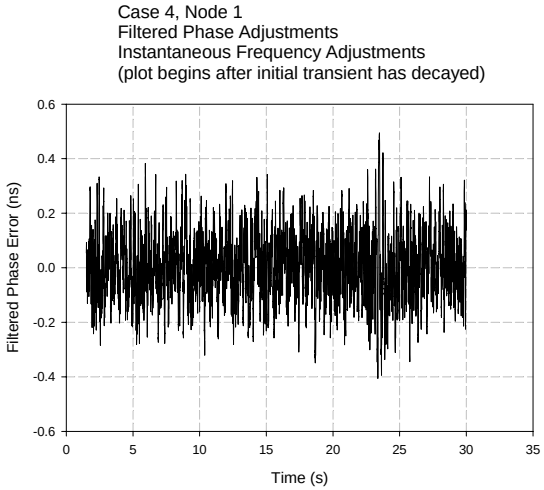

Fig. 8(c). Detailed view of filtered phase offset for Case 4, node 1

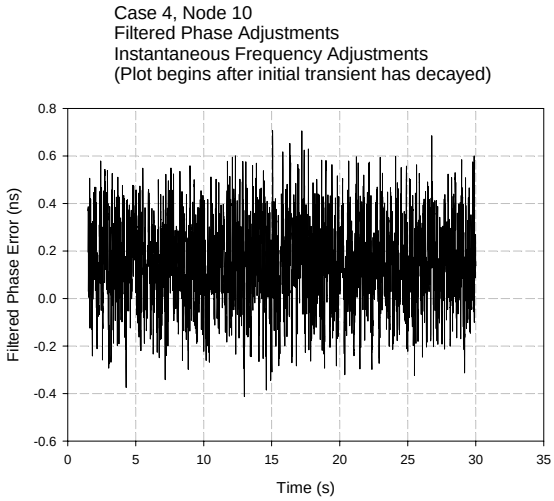

Fig. 8(d). Detailed view of filtered phase offset for Case 4, node 10

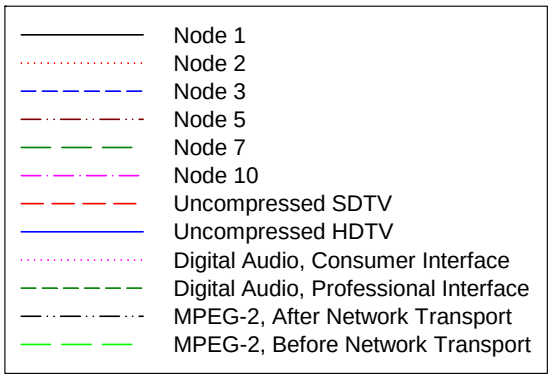

Fig. 9(a). Legend for Figs. 9(b) – (e)

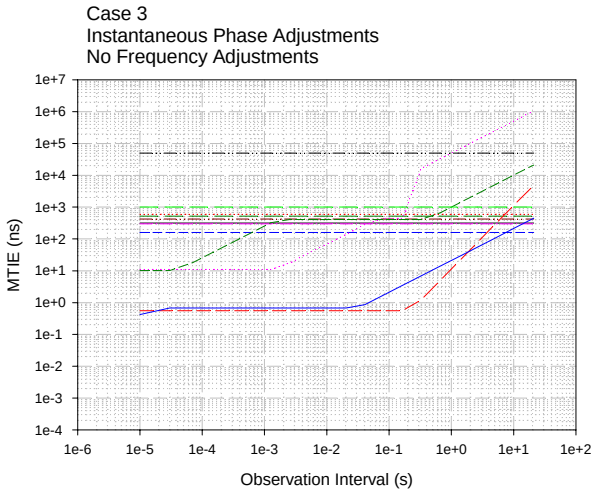

Fig. 9(b). MTIE, Case 3, unfiltered phase variation

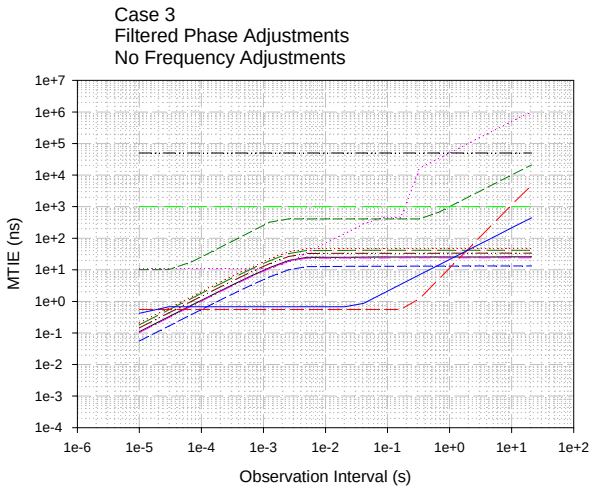

Fig. 9(c). MTIE, Case 3, filtered phase variation

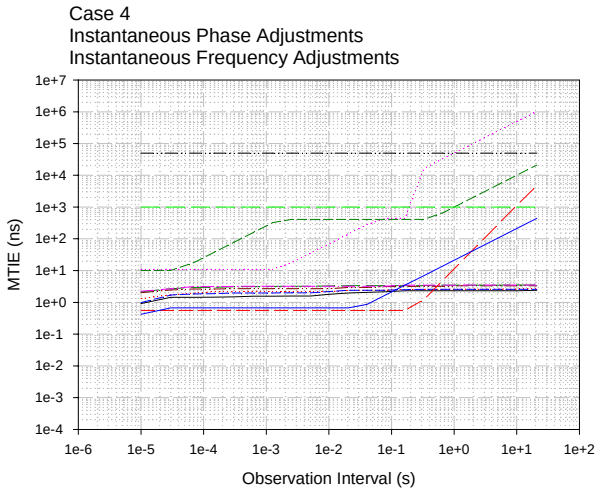

Fig. 9(d). MTIE, Case 4, unfiltered phase variation

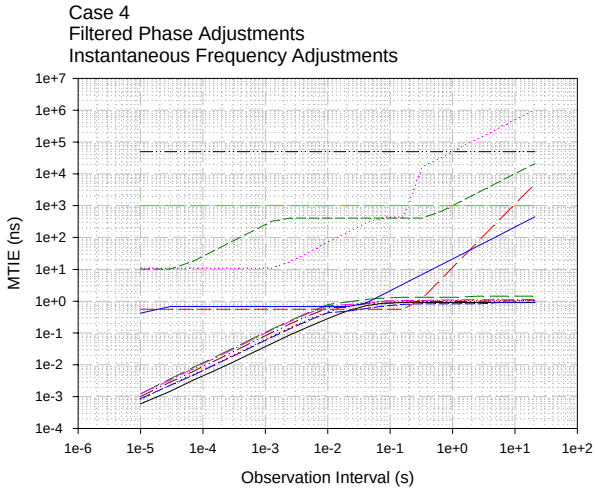

Fig. 9(e). MTIE, Case 4, filtered phase variation

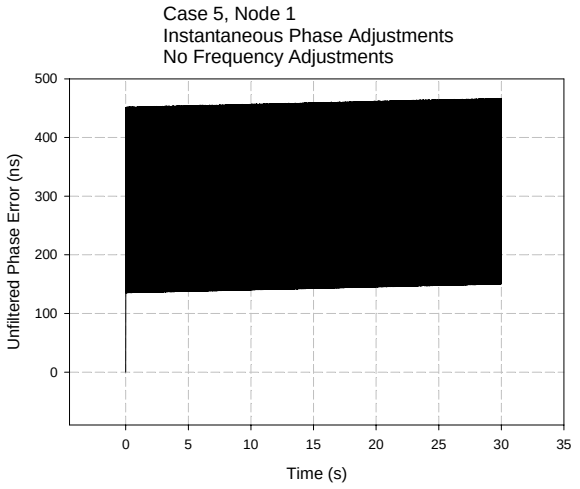

Fig. 10(a). Unfiltered phase offset for Case 5, node 1

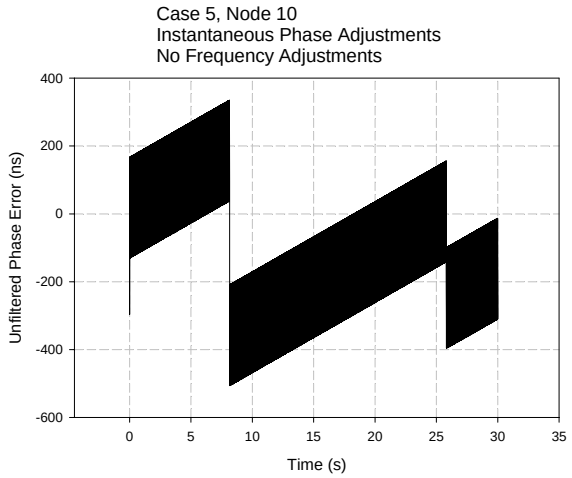

Fig. 10(b). Unfiltered phase offset for Case 5, node 10

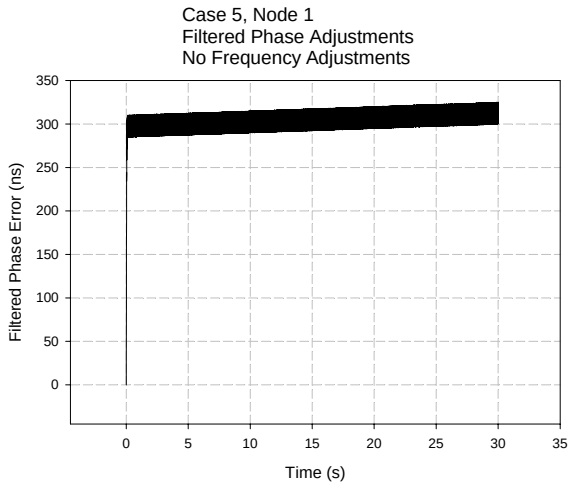

Fig. 10(c). Filtered phase offset for Case 5, node 1

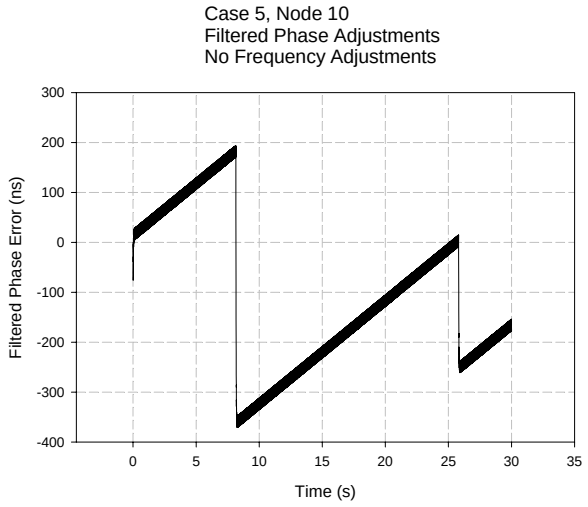

Fig. 10(d). Filtered phase offset for Case 5, node 1

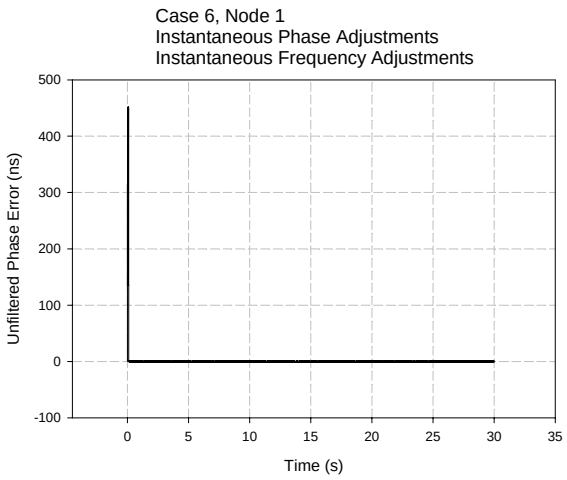

Fig. 11(a). Unfiltered phase offset for Case 6, node 1

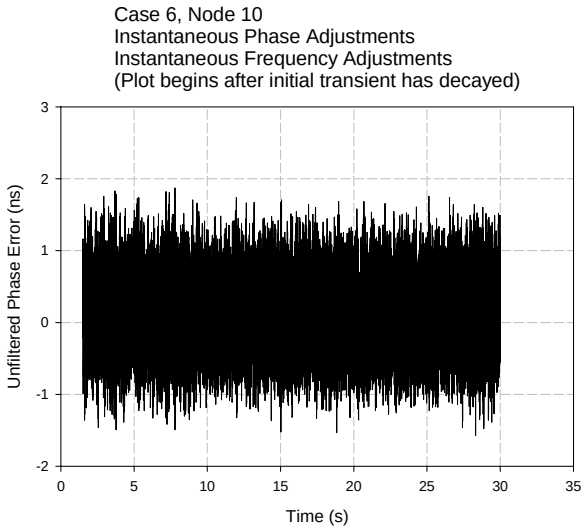

Fig. 11(b). Unfiltered phase offset for Case 6, node 10

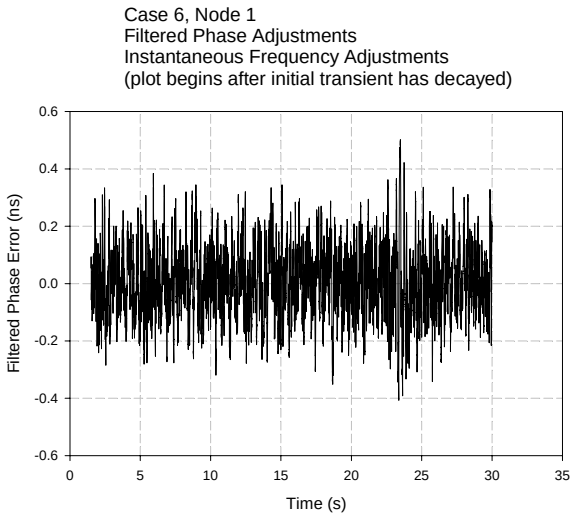

Fig. 11(c). Filtered phase offset for Case 6, node 1

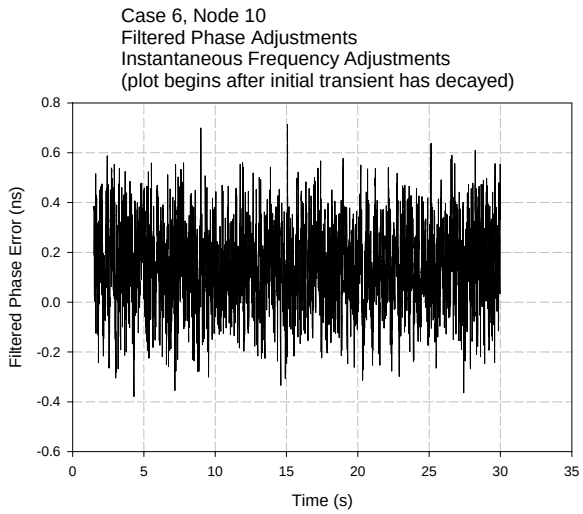

Fig. 11(d). Filtered phase offset for Case 6, node 10

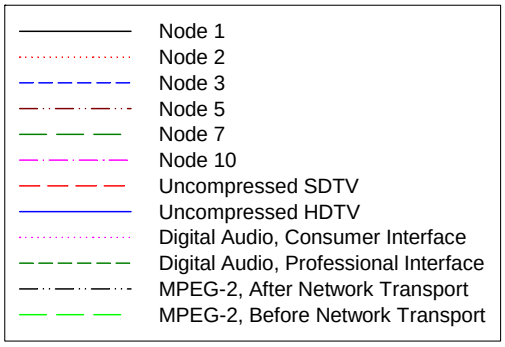

Fig. 12(a). Legend for Figs. 12(b) – (e)

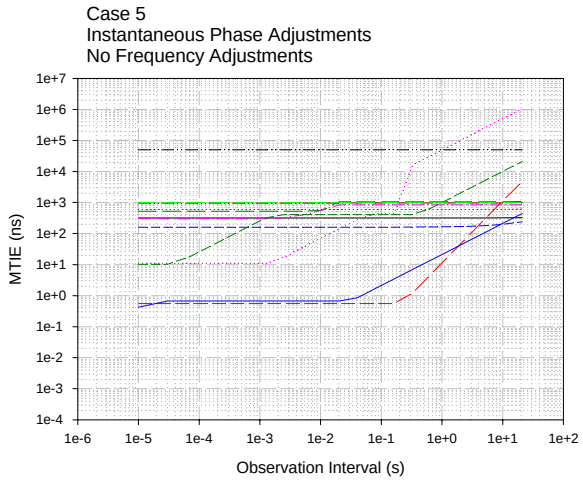

Fig. 12(b). MTIE, Case 5, unfiltered phase variation

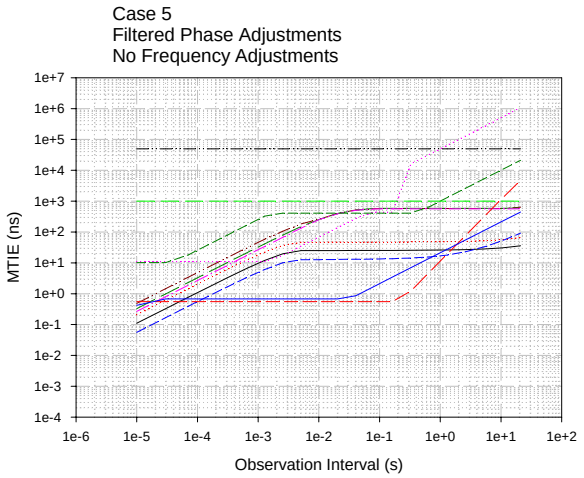

Fig. 12(c). MTIE, Case 5, filtered phase variation

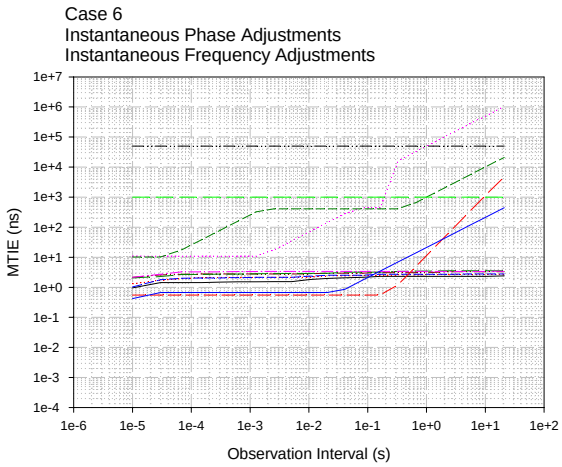

Fig. 12(d). MTIE, Case 6, unfiltered phase variation

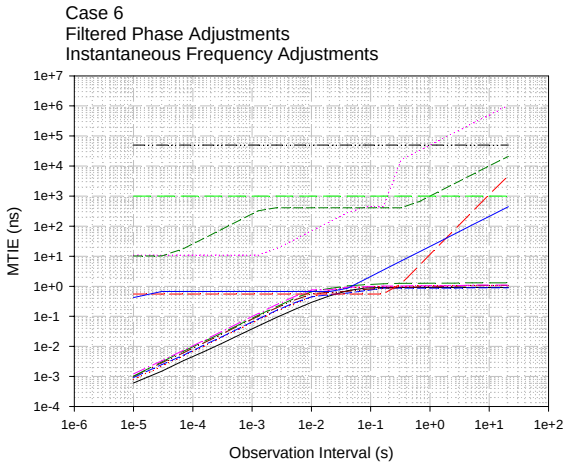

Fig. 12(e). MTIE, Case 6, filtered phase variation

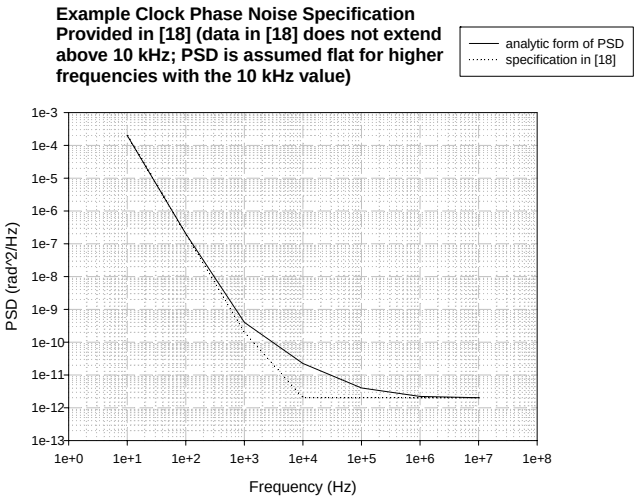

Fig. A-1. Example clock phase noise specification, taken from [18]. Note that data in [18] is given in dBc/Hz, and has been converted to rad^2/Hz.

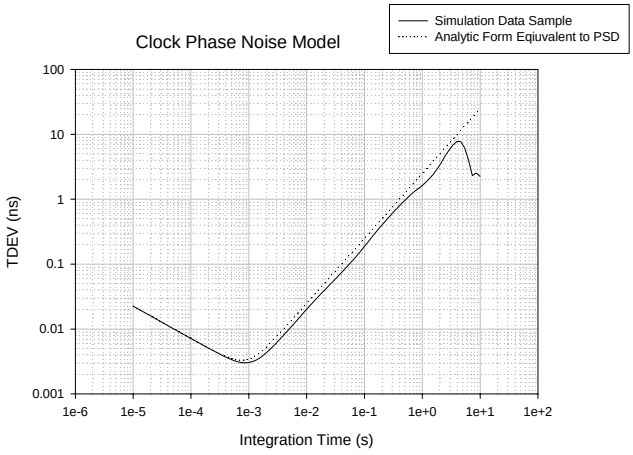

Fig. A-2. TDEV equivalent to PSD of Figure A-1, and TDEV computed from equivalent simulation model data.