A Master–Slave Synchronization Model for Enhanced Servo Clock Design

D. Macii, D. Fontanelli, D. Petri
DISI – Department of Information Engineering and Computer Science
University of Trento
Via Sommarive 14, 38100, Trento, Italy
E–mail: {macii,fontanelli,petri}@disi.unitn.it

***Abstract*—Slave servo clocks have an essential role in hardware and software synchronization techniques based on Precision Time Protocol (PTP). The objective of servo clocks is to remove the drift between slave and master nodes, while keeping the output timing jitter within given uncertainty boundaries. Up to now, no univocal criteria exist for servo clock design. In fact, the relationship between controller design, performances and uncertainty sources is quite evanescent. In this paper, we propose a quite simple, but exhaustive linear model, which is expected to be used in the design of enhanced servo clock architectures.**

***Keywords*—Synchronization, IEEE 1588, system models, linear control systems, uncertainty.**

I. Introduction

The standard IEEE 1588 is certainly the main reference document for the synchronization of wired distributed systems and it is getting more and more important also in the field of high–end wireless applications [1], [2]. The Precision Time Protocol (PTP) described in the standard was originally conceived as an application layer protocol able to achieve better accuracy than other synchronization solutions, such as the well–known Network Time Protocol (NTP) [3]. Indeed, the synchronization uncertainty of PTP software–only implementations is in the order of hundreds of μs, but it can be decreased down to 1 μs or less, through hardware–assisted time–stamping techniques. Consequently, increasing research efforts have been focused on the design of hardware solutions for PTP, especially based on Field Programmable Gate Arrays (FPGAs) [4], [5]. Usually, the minimum synchronization uncertainty achievable with hardware–assisted PTP implementations is in the order of 100 ns. Further uncertainty reductions can be obtained either by using more stable local oscillators (e.g., Oven Compensated Crystal Oscillators – OCXOs), or by decreasing the time intervals between subsequent synchronizations [6]. While the former choice is critical in terms of costs, the latter may cause considerable increments in both network traffic and bandwidth usage, thus potentially disturbing the primary activities of the network.

An essential component for high–accuracy synchronization is the controller correcting the slave clocks. In fact, such components, usually referred to as *servo clocks*, tend to remove the slave time drifts with respect to the master. Of course, this corrective action is particularly effective if servos are implemented in hardware. However, the standard IEEE 1588 does not provide any specific information about servo clock design. Also, many existing solutions do not consider the effect of the uncertainty sources affecting the PTP–based synchronization process [4], [5], [7]. These uncertainty sources include both the *internal* jitter generated by the oscillators inside each network node and the *external* random time fluctuations related to master–slave communications. Due to such phenomena, synchronization accuracy deteriorates when multiple cascaded network elements are used. Na et al. have recently proposed a probabilistic state–space model which quantifies how uncertainties propagate through the network, in view of achieving end–to–end jitter reduction [8]. However, we expect that uncertainty propagation could be significantly mitigated if servo clocks were designed not only to track the time of the master, but also to filter out the most significant noise sources injected into the loop controlling the slave device. Usually, the problems of master clock tracking and slave jitter reduction are tackled independently. For instance, in [9] Corell et al. suggest using an elementary first–order filter just before a Proportional and Integral (PI) controller adjusting the slave clock. In [10] authors show that the accuracy of a servo clock can be considerably improved if the PI controller is preceded by a Kalman filter estimating the clock skew. In both cases, the design criteria of the controller and the filter are unclear, because the effect of the various uncertainty sources is not analyzed in depth. In this paper, we tackle this problem by proposing a model that includes all the main uncertainty contributions affecting the performance of a PTP slave clock. In particular, in Section II, after describing the model, we will show how and why this is useful for servo clock design. In Section III we will prove the correctness of the theoretical model with some time–domain simulations in two simple cases. Such results will pave the way to the design of a low–jitter servo clock which will be presented in Section IV.

II. Model Description

Let us refer to τ as the nominal duration of the time intervals between consecutive PTP *Sync* and/or *Follow-up* messages. In PTP τ is a power of 2 and it is expressed in seconds, but its actual value may change by 30% with 90% confidence to allow network nodes to complete other ongoing operations [1]. Such intrinsic period variations can be modelled with a random sequence $\delta_\tau(\cdot)$, whose values for each synchronization

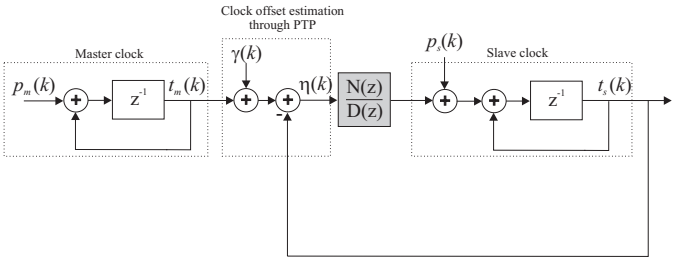

Fig. 1. Qualitative block diagram of the proposed synchronization model. The model refers just to a single master–slave link. The master and slave clocks can be represented as two discrete–time accumulators. The master–slave offset at the input of the controller (shadowed block) is estimated through the *Sync/Follow–up* and the *delay request/response* mechanisms stated in PTP specifications.

are common to both the master and the slave clocks. As a consequence, the duration of the $k-$th synchronization period measured by master and slave nodes is given respectively by:

$$\begin{array}{ll} \boldsymbol{p}_m(k) & =(1+\nu_m)\tau+\boldsymbol{\delta}_\tau(k)+\boldsymbol{\varepsilon}_m(k) \\ \boldsymbol{p}_s(k) & =(1+\nu_s)\tau+\boldsymbol{\delta}_\tau(k)+\boldsymbol{\varepsilon}_s(k) \end{array} \quad (1)$$

where:

- ν_m and ν_s are the non–ideal relative time drifts of the master and slave clocks caused by the systematic frequency skews of each local oscillator;
- $\boldsymbol{\varepsilon}_m(\cdot)$ and $\boldsymbol{\varepsilon}_s(\cdot)$ are the random sequences modelling the time jitter due to the accumulated power–law phase and frequency noises of master and slave clocks [11].

The former coefficients are related to clock *accuracy*, whereas the latter depend on *frequency stability*. In general, clock behavior is not stationary, as it changes over time due to aging and variable operating conditions (e.g. temperature variations). However, if clocks are periodically synchronized every few seconds, their short–term drift and stability characteristics can be assumed to be quasi stationary. As a consequence, ν_m and ν_s range from a fraction of one part per million (ppm) for high–performance OCXO and GPS–disciplined clocks to some tens of parts per million for typical crystal oscillators (XOs) [6]. Differently, the random variables $\boldsymbol{\varepsilon}_m(k)$ and $\boldsymbol{\varepsilon}_s(k)$ modelling the short–term time stability of master and slave oscillators can be assumed to be normally distributed with zero–mean and standard deviations σ_{ε_m} and σ_{ε_s} ranging from a fraction of 1 ns up to a few ns [11]–[12]. Since in PTP the grand–master is elected according to the *best master clock algorithm*, it is reasonable to assume that $|\nu_m| < |\nu_s|$ and $\sigma_{\varepsilon_m} < \sigma_{\varepsilon_s}$. However, if the grand–master is a low–cost GPS–disciplined clock, it may also happen that $|\nu_m| < |\nu_s|$, but σ_{ε_m} is in the order of 10–20 ns, i.e. larger than σ_{ε_s} [13].

In general, the time measured by the master and slave nodes, respectively, result from the accumulation of the synchronization periods defined in (1). Thus, both clocks can be simply modelled as two event–driven discrete–time integrators that are incremented anytime a new synchronization occurs. In the following, we will refer to $\boldsymbol{t}_m(\cdot)$ and $\boldsymbol{t}_s(\cdot)$ as the time sequences at the output of each accumulator. Such accumulators are combined to form a single feedback system, as shown in Fig. 1. In fact, the time offset between master and slave should be used to control (i.e. to correct) the slave clock. The time offset values result from the difference between the receiving time–stamp of a *Sync* or a *Follow–up* message measured by the slave clock and the corresponding sending time–stamp measured by the master. In general, this difference is affected by the one–way delay associated with the propagation of *Sync* and/or *Follow–up* messages. However, such a latency is usually estimated and compensated through the so–called *delay request–response mechanism* [1]. The time intervals between two subsequent delay request–response messages are usually much longer and subject to much larger random variations than the synchronization intervals. Indeed, the topology of most PTP–based networks is fixed or slowly variable over time. Therefore, frequent and timely delay request–response messages are unnecessary. For our purposes, the actual frequency of these messages is not relevant, because a new step of the servo–clock is computed anytime a new time offset is measured, regardless of how and when the one–way propagation delay is estimated. Accordingly, in the proposed model we can just assume that time offset values are affected by a sequence of uncertainty contributions $\boldsymbol{\gamma}(\cdot)$ resulting from the superimposition of the following sources, i.e.

- Imperfect compensation of the propagation delay (e.g., due to occasional propagation path asymmetries);
- Time–stamping jitter at both the transmitting and receiving ends;
- Limited clock resolution.

We preferred concentrating all uncertainty contributions listed above in $\boldsymbol{\gamma}(\cdot)$ for two reasons. First, using a single sequence obviously makes the whole model analysis much simpler. Second, estimating the global uncertainty of each time offset value is much easier than characterizing any individual uncertainty contribution. Since the uncertainty sources listed above are mostly independent and constant over time, it is reasonable to assume that $\boldsymbol{\gamma}(\cdot)$ is a stationary sequence of normal random variables. In general, both the mean and the standard deviation of $\boldsymbol{\gamma}(\cdot)$ heavily depend on node's performance and time–stamping techniques. However, if the one–way delay is properly estimated, in a first approximation the mean value of $\boldsymbol{\gamma}(\cdot)$ can be assumed to be negligible. The standard deviation of $\boldsymbol{\gamma}(\cdot)$ instead is in the order of some tens of μs in the case of Medium Access Layer (MAC) software time–stamping or in the order of tens of ns when hardware–assisted time–stamping techniques are used [14]. In conclusion, the $k-$th estimated clock offset at the input of the servo clock is

$$\boldsymbol{\eta}(k)=\boldsymbol{t}_m(k)-\boldsymbol{t}_s(k)+\boldsymbol{\gamma}(k) \qquad k\geq 0 \quad (2)$$

The proposed model highlights that the slave servo clock not only should track the time of the master, but it should also be able to make the closed loop system robust with respect to random period variations $\boldsymbol{\delta}_\tau(\cdot)$ and to filter out the noise injected into the loop, modelled by $\boldsymbol{\gamma}(\cdot)$, $\boldsymbol{\varepsilon}_m(\cdot)$ and $\boldsymbol{\varepsilon}_s(\cdot)$. In fact, the servo represented by the shadowed block

in Fig. 1 typically consists of a PI controller preceded by a very simple filter, e.g. a first–order Finite Impulse Response (FIR) as in [9] or a Kalman filter as in [10]. Observe that such filters are placed *inside* the loop. Consequently, using high–order or non–linear filters may lead to serious stability problems. Apparently, the noise contributions $\boldsymbol{\gamma}(\cdot)$ and $\boldsymbol{\varepsilon}_m(\cdot)$ could be more effectively attenuated if the filter, referring to Fig. 1, was placed immediately after adding $\boldsymbol{\gamma}(\cdot)$, i.e. *outside* the main loop. However, this is not feasible in practice, because, as stated above, the sequence $\boldsymbol{\eta}(\cdot)$ includes not only the uncertainty contributions related to the compensation of the propagation delay, but also the jitter that is intrinsically associated with the sending and receiving time–stamps of each *Sync* or *Follow–up* message. While the former contributions can be reduced with some external low–pass filter (not shown in Fig. 1 because included in the definition of $\boldsymbol{\gamma}(\cdot)$ and not relevant for the purpose of the paper), the latter one can be filtered only after the time–stamp values are exchanged, namely after closing the loop.

In order to provide some general criteria to design the slave servo clock, let us assume that it is perfectly linear and that its transfer function is $G(z) = \frac{N(z)}{D(z)}$. In the following, the $z-$transform of any time–domain sequence will be denoted with the corresponding capital letter, i.e. $Z\{x(k)\} \xrightarrow{Z} X(z)$. Notice that the $z-$transform of the output sequence $\boldsymbol{t}_s(\cdot)$ results from the superimposition of three contributions, i.e.

$$\boldsymbol{T}_s(z) = \frac{H_1(z)}{z-1}\boldsymbol{P}_m(z) + H_1(z)\boldsymbol{\Gamma}(z) + H_2(z)\boldsymbol{P}_s(z) \quad (3)$$

where

$$\begin{aligned} \boldsymbol{P}_m(z) &= \frac{(1+\nu_m)\tau z}{z-1} + \boldsymbol{\Delta}_\tau(z) + \boldsymbol{E}_m(z) \\ \boldsymbol{P}_s(z) &= \frac{(1+\nu_s)\tau z}{z-1} + \boldsymbol{\Delta}_\tau(z) + \boldsymbol{E}_s(z) \end{aligned} \quad (4)$$

are the $z-$transforms of (1), under the assumption that τ, ν_m and ν_s are constant for $k \geq 0$, and

$$H_1(z) = \frac{N(z)}{D(z)(z-1)+N(z)} \quad (5)$$

$$H_2(z) = \frac{D(z)}{D(z)(z-1)+N(z)} \quad (6)$$

are the closed–loop transfer functions of the system with respect to $\boldsymbol{T}_m(z)$ and $\boldsymbol{P}_s(z)$, respectively. Notice that (3) can be rewritten as follows:

$$\boldsymbol{T}_s(z) = T_w(z) + \boldsymbol{R}(z) \quad (7)$$

where

$$T_w(z) = H_1(z)\frac{(1+\nu_m)\tau z}{(z-1)^2} + H_2(z)\frac{(1+\nu_s)\tau z}{z-1} \quad (8)$$

is a *deterministic term* which depends on both the nominal synchronization interval and the systematic time drifts and

$$\boldsymbol{R}(z) = H_1(z)\left[\frac{\boldsymbol{E}_m(z)+\boldsymbol{\Delta}_\tau(z)}{z-1} + \boldsymbol{\Gamma}(z)\right] + H_2(z)[\boldsymbol{E}_s(z)+\boldsymbol{\Delta}_\tau(z)] \quad (9)$$

is a *noise term* comprising all the zero–mean random uncertainty contributions affecting the output of the disciplined slave clock. If (8) is decomposed into partial fractions, after some algebraic manipulations we obtain that:

$$T_w(z) = \frac{za_1}{(z-1)^2} + \frac{z(b_1+b_2)}{z-1} + \sum_{j=1}^{M}\frac{z(c_{1_j}+c_{2_j})}{z-\pi_j} \quad (10)$$

where π_j (with $|\pi_j| < 1$ for $j = 1, ..., M$) are the poles of the closed–loop transfer functions, a_1, b_1 and c_1 are the residues of the leftmost term in (8) and, similarly, b_2 and c_2 are the residues of the rightmost term in (8). From the analysis of (9) and (10), various basic design criteria can be deduced. Such criteria are shortly listed in the following.

1) The leftmost term in (10) is the $z-$transform of a linear ramp representing the time measured by the slave clock. In principle, this term depends on the *time drifts* coefficients, i.e. ν_s and ν_m. However, it can be easily proved that if the closed–loop system is stable, the order of $D(z)$ is larger than 1 and $N(z)$ has no roots in 1, then the drift between master and slave is removed. Indeed, under these assumptions $a_1 = (1+\nu_m)\tau$.
2) The middle additive term in (10) is responsible for the delay between master and slave clocks. Such a delay is proportional to the sum of b_1 and b_2. Given that $b_1 + b_2 = (\nu_s - \nu_m)\tau \lim_{z\to 1}\frac{D(z)}{N(z)}$, the output latency can be compensated if $D(z)$ has a root in 1 and $N(z)$ has no roots in 1.
3) Finally, the rightmost sum in (10) include all the unwanted higher–order modes affecting the output of the disciplined slave clock during the transient. If the closed–loop system is stable, these modes converge to zero. Thus they do not alter the expected output of the system in the steady state. As a consequence, the values of c_{1_j}, c_{2_j} and π_j for $j = 1, \ldots, M$ (or equivalently the zeros and poles of $H_1(z)$ and $H_2(z)$) can be chosen to minimize the noise term (9), while keeping the transient duration within wanted limits.

From the conditions 1) and 2) listed above, it follows that in order to have a steady state zero tracking error between the slave and the master $D(z)$ should be equal to $\bar{D}(z)(z-1)$, where $\bar{D}(z)$ is a generic polynomial function. In the simplest case, $\bar{D}(z) = 1$ and $N(z) = K(z-\alpha)$ with $\alpha \neq 1$, since $N(z)$ should not have roots in 1, at least one zero in the linear controller is needed for closed–loop stability, and no more than one zero is possible for controller causality. Under these assumptions, the controller is a PI and $N(z)$ can be rewritten as $N(z) = (K_p + K_i)z - K_p$, where K_p and K_i are the proportional and integral constants, respectively. Accordingly, the PI closed–loop transfer functions (5) and (6) become:

$$H_1(z) = \frac{(K_p+K_i)z - K_p}{z^2 + (K_p + K_i - 2)z + 1 - K_p} \quad (11)$$

and

$$H_2(z) = \frac{z-1}{z^2 + (K_p + K_i - 2)z + 1 - K_p} \quad (12)$$

Observe that the system is stable for $0<K_p<2$ and $0\leq K_i< 4-2K_p$. How such parameters affect the noise rejection and the length of the transient of the system will be clarified in the next Sections.

III. Model Analysis

In order to prove the correctness of the theoretical model described in Section II, two naïve clock controllers are now proposed and analyzed through simulations. Of course, such solutions are never used in practice, but they are helpful to understand better the implications of the model. The two simple controllers considered in the following subsections are obtained by setting:

A) $K_p=1$ and $K_i=0$;
B) $K_p=1$ and $K_i=1$.

The former configuration corresponds to a simple unit–gain proportional controller, whereas the latter is a so–called dead–beat PI controller. All simulations rely on the two following assumptions:

- The mean propagation delays are perfectly compensated;
- Time–stamping and clock adjustment are performed at the hardware level.

Consequently, $\boldsymbol{\gamma}(\cdot)$ is a sequence of normally distributed zero–mean random variables and its standard deviation is close to 30 ns. The master is assumed to be a high–performance 10 MHz disciplined GPS clock with negligible systematic drift ($\nu_m = 0$) and 1–second short–term stability $\sigma_{\varepsilon_m} \approx 100$ ps. The slave clock instead is a digital timer stimulated by a 100 MHz XO, with $\nu_s = 50$ μs, and moderate short–term stability $\sigma_{\varepsilon_s} \approx 2$ ns. This choice follows directly from recently presented accuracy analysis for PTP clock synchronization [7]. The random jitter of the XO is synthesized with a special pseudo–random generator able to accurately simulate the phase and frequency noises of real clocks [15]. In all cases, the duration of the synchronization intervals is $\tau = 1$ s with $\pm 30\%$ random variations, as described in [1].

A. Unit–gain proportional controller

If $K_p=1$ and $K_i=0$, then $N(z) = D(z) = 1$ and $H_1(z) = H_2(z) = \frac{1}{z}$. Therefore, according to (7), (8) and (9), the time of the slave clock is just a delayed, noisy replica of the time of the master. In fact, neither $H_1(z)$ nor $H_2(z)$ filter out the noise injected into the loop. This behavior is confirmed by the simulation results shown in Fig. 2. Notice that the delay between the master and slave clocks is approximately $\nu_s - \nu_m = 50$ μs, as expected from the theoretical analysis. This means that the two clocks do not drift away from one another. However, the output jitter of the slave clock (bottom–right picture) is dominated by the native jitter of the slave oscillator, which indeed is unfiltered.

B. Dead–beat PI controller

If $K_p = 1$ and $K_i = 1$, the delay between the master and slave clocks after reaching the steady state is zero, because $D(z)$ has a root in 1. In this case, $H_1(z) = \frac{2z-1}{z^2}$ and $H_2(z) = \frac{z-1}{z^2}$. As known from control theory, the rate of convergence of the PI controller towards its steady state value is related to the eigenvalues of the closed–loop matrix. More precisely, the smaller the eigenvalues, the faster the convergence. Hence, if the closed–loop eigenvalues are zero (which occurs exactly when $K_p = K_i = 1$), the slave has the fastest convergence towards the reference master (see Fig. 3, top picture). Unfortunately, the fastest rate of convergence can only be obtained at detriment of noise reduction (Fig. 3, bottom pictures). In fact, the power of the noise term is amplified in this case, in accordance with (9).

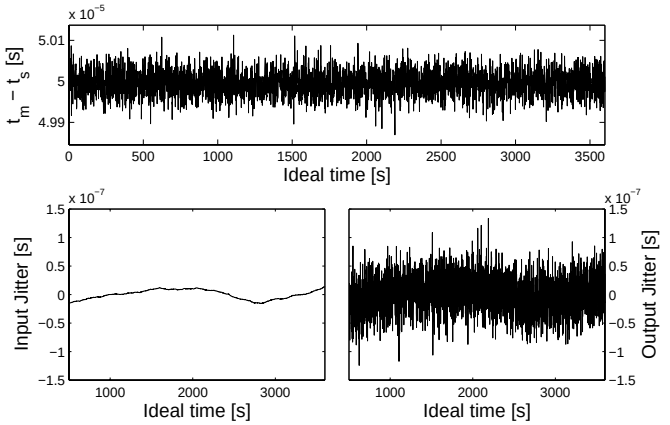

Fig. 2. Accuracy and stability performances of a servo clock disciplined by unit–gain proportional controller. The topmost figure refers to the difference between master and slave clocks. The bottom–left picture shows the native input jitter of the master, whereas in the bottom–right picture the corresponding output jitter of the slave clock is plotted.

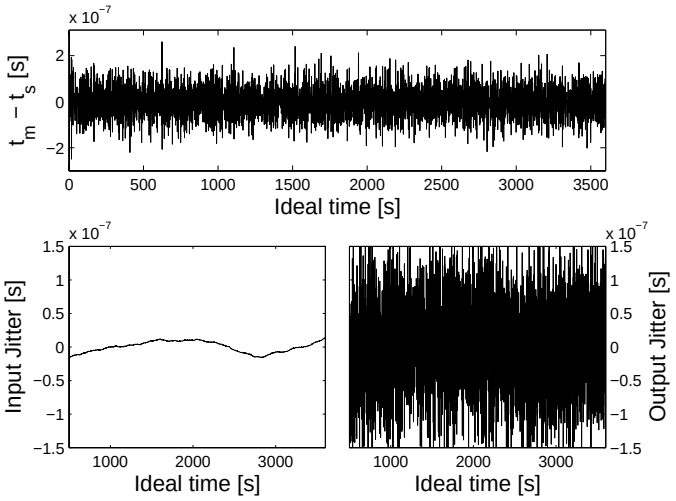

Fig. 3. Accuracy and stability performances of a servo clock disciplined by a dead-beat PI controller (i.e. with $K_p = K_i = 1$). The meaning of the various pictures is the same as in Fig. 2.

IV. Jitter–oriented Servo Clock Design

The time–domain simulations described in Section III are just two simple examples which confirm the validity of the underlying theoretical model. To better understand the trade–

off between jitter reduction and master time tracking, an analysis of the steady state influence of the noises in the slave clock is needed. Albeit such a steady state analysis does not describe the frequency relation between the system and the noises, it can be used as a guideline for the controller gain choices. More importantly, this approach is not restricted to the PI case specifically considered in the following.

Let us consider the state space realizations of the master and slave clocks and of the PI controller, i.e.,

$$\begin{array}{cc} \text{Master} & \text{Slave} \\ \begin{cases} x_m^+ = x_m + \boldsymbol{p}_m \\ \boldsymbol{t}_m = x_m \end{cases} & \begin{cases} x_s^+ = x_s + \boldsymbol{p}_s + u_s \\ \boldsymbol{t}_s = x_s \end{cases} \end{array}$$
$$\text{PI Controller}$$
$$\begin{cases} x_c^+ = x_c + \boldsymbol{t}_m + \boldsymbol{t}_s + \gamma \\ y_c = K_i x_c + (K_p + K_i)(\boldsymbol{t}_m + \boldsymbol{t}_s + \boldsymbol{\gamma}) \end{cases} \quad (13)$$

where u_s is the control input to the slave clock. Observe that any explicit reference to the event counter variable k has been removed for simplicity and x^+ stands for $x(k+1)$. To let the analysis more straightforward, we start with the derivation of the closed–loop system dynamics without including the noise sources $\boldsymbol{\varepsilon}_m$, $\boldsymbol{\varepsilon}_s$ and $\boldsymbol{\gamma}$ described in equations (1) and (2), that is, we consider the sequences $\bar{\boldsymbol{p}}_m = \boldsymbol{p}_m - \boldsymbol{\varepsilon}_m$ and $\bar{\boldsymbol{p}}_s = \boldsymbol{p}_s - \boldsymbol{\varepsilon}_s$. Notice that the output y_c of the controller is connected to the input of the slave clock u_s. Therefore, we have that:

$$\begin{bmatrix} x_s \\ x_m \\ x_c \end{bmatrix}^+ = \begin{bmatrix} 1 - K_p - K_i & K_p + K_i & K_i \\ 0 & 1 & 0 \\ -1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_s \\ x_m \\ x_c \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \bar{\boldsymbol{p}}_s + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \bar{\boldsymbol{p}}_m \quad (14)$$

Since in the absence of jitter in the master clock the objective of the controller is to minimize the difference between x_s and x_m, we define a new set of variables $\mathbf{q} = [x_s - x_m,\ x_c + (\nu_s - \nu_m)\tau/K_i,\ x_s]^T = [q_1,\ q_2,\ q_3]^T$, for which the dynamics is given by

$$\mathbf{q}^+ = \begin{bmatrix} 1 - K_p - K_i & K_i & 0 \\ -1 & 1 & 0 \\ -K_p - K_i & K_i & 1 \end{bmatrix} \mathbf{q} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \bar{\boldsymbol{p}}_s = A\mathbf{q} + B_{p_s} \bar{\boldsymbol{p}}_s \quad (15)$$

This expression highlights, again, that a stabilizing PI controller ensures null steady state tracking error [i.e., $q_1(+\infty) = 0 \Rightarrow x_s(+\infty) = x_m(+\infty) \Rightarrow q_3(+\infty) = x_m(+\infty)$]. Also, the drift difference between master and slave clocks is accumulated in the controller state variable [i.e., $q_2(+\infty) = 0 \Rightarrow x_c(+\infty) = (\nu_m - \nu_s)\tau/K_i$].

The noises affect the performance of the PI controller proportionally to their intensities. Indeed, if we include the various noise sources described in Section II it follows that:

$$\begin{aligned} \mathbf{q}^+ &= A\mathbf{q} + B_{p_s}\bar{\boldsymbol{p}}_s + \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \boldsymbol{\varepsilon}_s - \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{\varepsilon}_m + \begin{bmatrix} K_p + K_i \\ 1 \\ K_p + K_i \end{bmatrix} \boldsymbol{\gamma} \quad (16) \\ &= A\mathbf{q} + B_{p_s}\bar{\boldsymbol{p}}_s + B_{\varepsilon_s}\boldsymbol{\varepsilon}_s - B_{\varepsilon_m}\boldsymbol{\varepsilon}_m + B_\gamma \boldsymbol{\gamma}. \end{aligned}$$

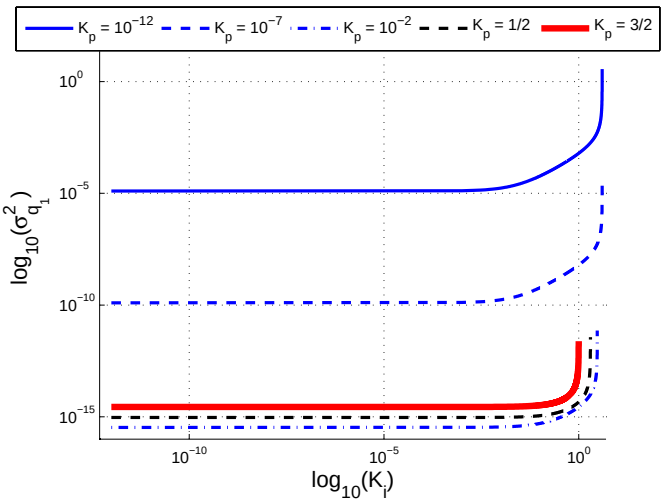

Fig. 4. Error variance $\sigma_{q_1}^2$ given a set of possible choices of the proportional parameter K_p. For each choice, $\sigma_{q_1}^2$ is plotted as a function of all the possible parameter K_i values that generate a stable closed loop system.

Considering the covariance matrix $P(k) = \mathrm{E}\left\{\mathbf{q}(k)\mathbf{q}(k)^T\right\}$, with $P(0) = \mathrm{E}\left\{\mathbf{q}(0)\mathbf{q}(0)^T\right\}$, we obtain

$$P(k+1) = AP(k)A^T + B_{\varepsilon_s}B_{\varepsilon_s}^T\sigma_{\varepsilon_s}^2 + B_{\varepsilon_m}B_{\varepsilon_m}^T\sigma_{\varepsilon_m}^2 + B_\gamma B_\gamma^T\sigma_\gamma^2 \quad (17)$$

From (17), reminding that the steady state variance $\sigma_{q_1}^2$ of $q_1(+\infty)$ is the element in position $[1,1]$ of the covariance matrix $P(+\infty)$, we have

$$\sigma_{q_1}^2 = \frac{(2K_p^2 + K_pK_i + 2K_i)\sigma_\gamma^2 + 2(\sigma_{\varepsilon_m}^2 + \sigma_{\varepsilon_s}^2)}{K_p(4 - K_i - 2K_p)}. \quad (18)$$

The trend of (18) is depicted in Fig. 4, after setting the values of σ_γ, σ_{ε_m} and σ_{ε_s} as stated in Section III. Observe that when $K_i \to 0$ (but with $K_i > 0$) the overall noise variance is minimum. This condition corresponds to two closed–loop complex conjugated eigenvalues close to the unitary circle. Since the transient length is directly related to the modulus of the eigenvalues (the lower the modulus, the faster the transient), this choice corresponds to very long transients, but the noise rejection is maximum.

As a consequence of these considerations, as a last example we propose an alternative servo clock PI that trades the convergence rate of the controller for noise rejection. In the case considered, a transient length of no more than 400 s is imposed, which corresponds to 400 steps of the discrete system. The steady state is assumed reached if the error between the master and the slave in the ideal case is reduced by a factor of 10^{-5} with respect to the initial conditions (in the dead–beat case this value is reached after 2 steps). This corresponds to have two eigenvalues with modulus of at most $|\lambda_{max}| = 0.973$ (since $|\lambda_{max}|^{400} \approx 1.6 \cdot 10^{-5}$). The choice of the PI gains $K_p = 0.05413$ and $K_i = 0.0027$ satisfies such a condition. In fact, the closed–loop eigenvalues are $\lambda_i = 0.9716 \pm 0.0435j$, with modulus $0.9726 < |\lambda_{max}|$. The time–domain simulation results related to this example

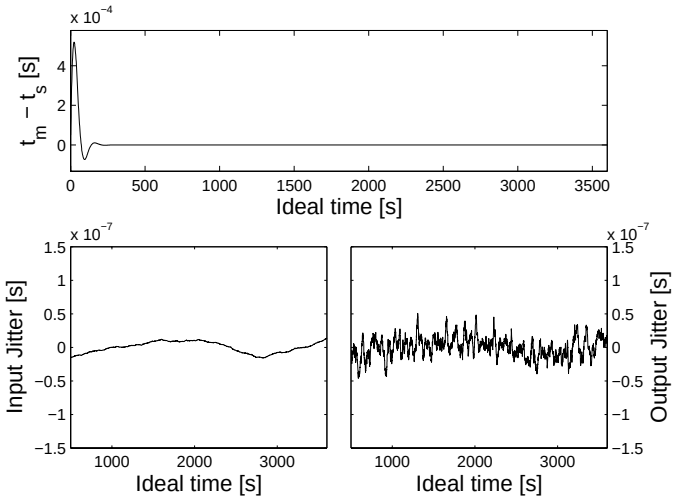

Fig. 5. Delay and stability performances of PI servo clock with $K_p = 0.05413$ and $K_i = 0.0027$. The meaning of the various pictures is the same as in Fig. 3.

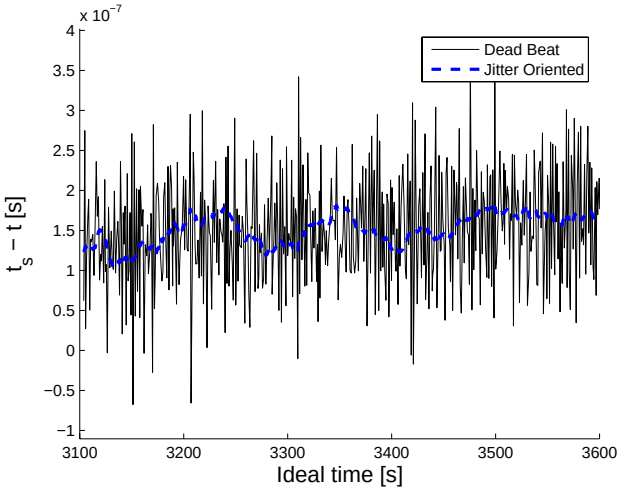

Fig. 6. Error between the controlled slave clock and the ideal time-scale. Both the behaviors of the dead–beat and the jitter oriented PI are depicted.

are shown in Fig. 5. Compared with the dead–beat case, the output noise is considerably reduced, although the influence of the local oscillator jitter is still significant. Such a reduction is more evident in Fig. 6 where the timing errors between the controlled slave clock and the ideal time–scale are plotted in the same picture for both the dead–beat and the jitter–oriented PI controller.

V. Conclusions

This paper deals with an analytical model describing the master–slave PTP synchronization process based on the use of servo clocks. The proposed model includes the main uncertainty contributions affecting synchronization accuracy and it is useful for two reasons. On one hand, it provides some general criteria for servo clock design. On the other, it can be used to compare the performances of different servo clock architectures. Up to now, the model has been used to analyze the behavior of PI servo clocks. Further research activities are currently ongoing to design more effective structures.

References

[1] *IEEE 1588:2008, Precision clock synchronization protocol for networked measurement and control systems*, New York, USA, Jul. 2008.

[2] T. Cooklev, J. Eidson, and A. Pakdaman, "An implementation of IEEE 1588 over IEEE 802.11b for synchronization of wireless local area network nodes," *IEEE Trans. on Instrumentation and Measurement*, vol. 56, no. 5, pp. 1632–1639, Oct. 2007.

[3] D. L. Mills, "Precision synchronization of computer network clocks," *ACM Computer Communication Review*, vol. 24, no. 2, pp. 28–43, Apr. 1994.

[4] R. Holler, T. Sauter, and N. Kero, "Embedded synUTC and IEEE 1588 clock synchronization for industrial ethernet," in *Proceedings of IEEE Conf. on Emerging Technologies and Factory Automation (ETFA)*, Lisbon, Portugal, Sep. 2003, pp. 422–426.

[5] S. Meier, H. Weibel, and K. Weber, "IEEE 1588 syntonization and synchronization functions completely realized in hardware," in *Proc. of Int. Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS)*, Ann Arbor, Michigan, USA, Sep. 2008, pp. 1–4.

[6] P. Loschmidt, R. Exel, A. Nagy, and G. Gaderer, "Limits of synchronization accuracy using hardware support in IEEE 1588," in *Proc. of Int. Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS)*, Ann Arbor, Michigan, USA, Sep. 2008, pp. 12–16.

[7] R. L. Scheiterer, C. Na, D. Obradovic, and G. Steindl, "Synchronization performance of the precision time protocol in industrial automation networks," *IEEE Trans. on Instrumentation and Measurement*, vol. 58, no. 6, pp. 1849–1857, Jun. 2009.

[8] C. Na, D. Obradovic, and R. L. Scheiterer, "Probabilistic model for clock synchronization of cascaded network elements," in *Proc. of International Instrumentation and Measurement Conference (I2MTC)*, Singapore, May 2009, pp. 1594–1598.

[9] K. Corell, N. Berendt, and M. Branicky, "Design considerations for software only implementations of the IEEE 1588 precision time protocol," in *Proc. of Conf. on IEEE–1588 Standard for a Precise Clock Synchronization Protocol for Networked Measurement and Control Systems*, Winterthur, Switzerland, Oct. 2005, pp. 1–6.

[10] H. Abubakari and S. Sastry, "IEEE 1588 style synchronization over wireless link," in *Proc. of Int. Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS)*, Ann Arbor, Michigan, USA, Sep. 2008, pp. 127–130.

[11] S. Bregni, "Clock stability characterization and measurement in telecommunications," *IEEE Trans. on Instrumentation and Measurement*, vol. 46, no. 6, pp. 1284–1294, Dec. 1997.

[12] M. A. Lombardi, *NISTIR 6610 - NIST Frequency Measurement and Analysis System: Operators Manual*, National Institue of Standards and Technology (NIST), Aug. 2001.

[13] L. Gasparini, O. Zadedyurina, G. Fontana, D. Macii, A. Boni, and Y. Ofek, "A digital circuit for jitter reduction of gps–disciplined 1–pps synchronization signals," in *Proc. of IEEE Int. Workshop on Advanced Methods for Uncertainty Estimation in Measurement (AMUEM)*, Trento, Italy, Jul. 2007, pp. 84–88.

[14] H. Weibel and D. Bechaz, "IEEE 1588 - implementation and performance of time stamping techniques," in *Proc. of Int. Conf. on IEEE 1588, Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, Gaithersburg, Maryland, USA, Sep. 2004, pp. 24–39.

[15] N. J. Kasdin and T. Walter, "Discrete simulation of power law noise," in *Proc. of IEEE Frequency Control Symposium*, Hershey, USA, May 1992, pp. 274–283.