New Phase-Locked Loop Design: Understanding the Impact of a Phase-Tracking Channel Detector

Jaewook Lee[1], Jaekyun Moon[2], *Fellow, IEEE*, Tong Zhang[3], and Erich F. Haratsch[4]

[1]Quantum Corporation, Irvine, CA 92617 USA
[2]Department of Electrical Engineering, KAIST, Daejeon 305-701, Republic of Korea
[3]Department of Electrical, Computer and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180 USA
[4]LSI Corporation, Allentown, PA 18109 USA

We consider design of a phase-locked loop (PLL) that runs in conjunction with a phase-compensating channel detector capable of tracking and compensating for slow-varying phase errors. The main significance of this highly nontraditional timing recovery structure is that unlike the traditional PLLs, the loop's overall tracking ability and the steady-state jitter performance can be controlled more or less separately. The effective bandwidth of the PLL is set much lower than the traditional PLL for given input phase error fluctuations. As a result, the PLL jitter is made considerably smaller than traditional systems. On the other hand, the tracking ability of the overall loop depends largely on the window parameter associated with the phase estimator operating inside the Viterbi-like trellis detector. The window parameter has only a small effect on the loop jitter. The new PLL design is tested via tracking speed, timing jitter and error rate performance comparisons against a timing recovery method based on traditional PLL design. Simulation results in a turbo equalizer setting are also presented that validate the new PLL design methodology proposed.

***Index Terms*—Frequency-tracking, intersymbol interference, jitter, phase-locked loop, timing recovery.**

I. Introduction

TRADITIONAL phase-locked loop (PLL) design focuses on achieving the best tradeoff between timing jitter and tracking ability. A small equivalent loop bandwidth indicates small timing jitter but it also means inability to track fast timing phase fluctuations. The traditional PLL loop bandwidth characteristics are largely determined by the low pass filter (LPF) that is typically placed between the phase detector (or timing error detector, as is often called) and the voltage controlled oscillator (VCO). The usual first-order loop filter is controlled by two gain parameters: α and β, known as the phase-path gain and the frequency-path gain, respectively. The overall conventional discrete-time PLL (DPLL) structure for small phase error is shown in Fig. 1. The phase error θ_k between the phase offset ϕ_k in the observation signal and its local estimate τ_k is amplified by the timing error detector (TED) gain A and corrupted by the effective noise p_k. The TED output $\hat{\theta}_k$ is then filtered before driving the VCO, whose output is to track the incoming phase ϕ_k. The additional loop latency ξ models the TED latency and/or pipeline delays in the implementation. The traditional PLL design basically amounts to finding the best combination of α and β values, given the jitter, tracking speed and stability requirements.

The requirement for a proper damping ratio, which controls the behavior of the PLL transient responses to a phase step or a frequency step in the input phase error [1], makes β considerably smaller than α. Thus, the effective loop bandwidth is largely determined by α. Accordingly, to make the loop bandwidth small, α needs to be small. However, small LPF gains in PLL slow down the tracking speed of the loop.

Manuscript received August 21, 2009; revised October 19, 2009; accepted October 26, 2009. Current version published February 18, 2010. Corresponding author: J. Moon (e-mail: jaemoon@ee.kaist.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMAG.2009.2035927

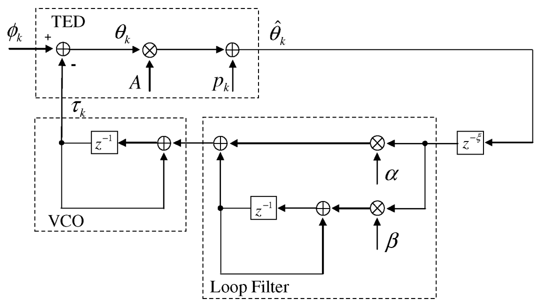

Fig. 1. Conventional discrete-time PLL model.

Another conflicting requirement that exists in traditional loop design is related to the TED latency ξ and its impact on loop's tracking speed. The TED output is typically constructed based on the symbol decisions released by the detector. High-quality but long-latency detector decisions lead to a good TED output quality but a poor tracking speed for the loop. Low-latency decisions, on the other hand, tend to be poor in quality but allows the loop to react quickly to the incoming phase error fluctuations. The designer of the traditional loop needs to strike a good balance between the TED output quality and the tracking ability of the loop.

In this paper, we present a PLL design that departs considerably from the traditional guidelines discussed above. Specifically, our PLL operates in conjunction with a channel detector that is capable of tracking phase fluctuations as well as compensating for a finite phase error in the symbol detection process, as described elsewhere [2], [3]. The tracking ability of our loop is largely dependent upon the window parameter M, the number of consecutive observation samples used in estimating the phase. It is shown that M can be set more or less independent of the traditional loop bandwidth, which determines steady-state jitter of the PLL. Remarkably, in our loop, the tracking ability and PLL jitter can be controlled almost independently. Specifically, we set M to be small enough to maintain good tracking speed while using small filter gains to achieve a small jitter. We can also make the damping ratio very large (i.e., make β negligibly small compared to the already small α). A large damping ratio creates very slow decay in the residual phase error when there exists frequency mismatches between the VCO output and the incoming signal. This does not hurt our loop, however, because of the detector's ability to compensate for finite residual phase error. Our loop also can maintain a good TED output quality with a small TED latency.

Whereas our previous papers discussed in detail the phase-tracking/compensating channel detector and performance of some example timing loops that use such a detector [2], [3], the main contribution of the present paper is in providing loop analysis and a general design guideline for this highly nontraditional PLL structure. In particular, we describe how M, α, β and the decision delay ξ are chosen in our loop in consideration of the overall tracking speed, jitter performance and stability. The error rate, tracking speed and timing jitter performances are compared with the traditional PLL in a systematic fashion. Performance is tested in a turbo equalizer setting as well, and simulation results are presented which validate the proposed PLL and design methodology.

II. Channel Model and PLL With Phase-Compensating Detector

Assume that the baud-rate sampled observation sequence at the receiver is given by

$$c_k = \sum_i x_i f[(k-i)T + \theta_k] + n_k \tag{1}$$

where x_i is the transmitted symbol sequence, $f(t)$ is the impulse response (or dibit response for magnetic recording) of the channel, T is the symbol interval, θ_k represents the phase mismatch between the arrived signal and the local sampling device and n_k denotes the additive white noise sequence. Assuming θ_k is small and slow-varying compared to the symbol rate, $1/T$, we write

$$\begin{aligned} c_k &\approx \sum_i x_i[f\{(k-i)T\} + \theta_k f'\{(k-i)T\}] + n_k \\ &= \sum_j f_j x_{k-j} + \theta_k \sum_j g_j x_{k-j} + n_k \\ &= y_k + \theta_k \varepsilon_k + n_k, \end{aligned} \tag{2}$$

where g_j denotes the symbol-rate samples of $f'(t), y_k = \sum_j f_j x_{k-j}$ and $\varepsilon_k = \sum_j g_j x_{k-j}$. The approximate channel model in (2) is used to develop the timing algorithm but the accurate model of (1) is used in actual performance analysis and simulation runs.

We now provide a quick review on the joint phase estimation and symbol detection algorithm based on Viterbi-like trellis path search [2], [3]. Assume a trellis diagram based on state description where a state-transition gathers enough consecutive symbols so that both y_k and ε_k are completely specified given a branch that connects two successive states (note g_j is typically longer than f_j so that our trellis is larger than the standard trellis based only on f_j). See Fig. 2. In the kth stage of the trellis diagram, consider two competing paths (assuming binary input symbols) v_1 and v_2 leaving two states a_1 and a_2, respectively, on the left side and arriving at a given state b on the right side. Assume that at the beginning of the processing stage, there already exist phase estimates, $\hat{\theta}_{k-1}(a_1)$ and $\hat{\theta}_{k-1}(a_2)$, associated with the respective paths leading to a_1 and a_2, in addition to the usual path metrics, $P_{k-1}(a_1)$ and $P_{k-1}(a_2)$. In fact, assume a sequence of phase estimates exists for each survivor path leading to a_l. As will be discussed shortly, the new phase estimate associated with branch $v_l, l = 1$ or 2, is obtained as a function of $\hat{\theta}_{k-1}(a_l), c_k$ and other branch- and path-specific variables. For now, let us just call the new estimates $\hat{\theta}_k(v_1)$ and $\hat{\theta}_k(v_2)$. The branch metrics are then computed as $\lambda_k(v_l) = [c_k - y_k(v_l) - \hat{\theta}_k(v_l)\varepsilon_k(v_l)]^2$ for $l = 1, 2$. Path arbitration follows as usual, by performing the add, compare and select (ACS) operations on the updates path metrics.

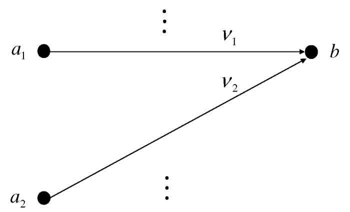

Fig. 2. Trellis diagram at kth stage.

The difference here relative to the usual Viterbi operation [4] is that the phase estimate sequence associated with the path leading to state b gets updated by appending $\hat{\theta}_k(v_m)$ to the phase estimate sequence specific to a_m. In this way, we maintain a phase estimate trajectory as well as a symbol decision sequence associated with each survivor path in every processing stage. A symbol decision is released with a delay of ω from the best survivor path to date whereas a phase estimate is released to the feedback timing loop with a delay of ξ. This method can be viewed as running a separate TED for each survivor path. Also see [5] for a timing recovery method based on running an entire PLL for each survivor path.

Now we describe how the phase estimate $\hat{\theta}_k$ is obtained for the current symbol stage. Assuming the phase error is constant within a window of M sample periods, the value of θ that minimizes the accumulated branch metrics $\sum_{i=k-M+1}^{k} [c_i - y_i - \theta\varepsilon_i]^2$ can be written in a recursive form [3]:

$$\hat{\theta}_k \approx \hat{\theta}_{k-1} + \gamma(e_k\varepsilon_k - e_{k-M}\varepsilon_{k-M}) \tag{3}$$

with $\gamma = 1/M\sigma_x^2\eta$, where $\sigma_x^2 = E\{x_k^2\}$ and $\eta = \sum_i g_i^2$. Note that $e_k\varepsilon_k$ is branch-specific and $e_{k-M}\varepsilon_{k-M}$ is path-specific (assuming the latest M values of $e_k\varepsilon_k$ are stored for each survivor path); it is easy to see how the phase estimation can be done for each branch separately, i.e., $\hat{\theta}_k(v_l) = \hat{\theta}_{k-1}(a_l) + \gamma[e_k(v_l)\varepsilon_k(v_l) - e_{k-M}(a_l)\varepsilon_{k-M}(a_l)]$ for $l = 1, 2$.

III. Window Parameter Selection

A major design issue is how to choose the window size M for the phase estimator. The variance of the phase estimate is given by $\sigma_{\hat{\theta}}^2 = \sigma_n^2 / \sum_{i=k-M+1}^{k} E\{\varepsilon_i^2\}$, which tends to zero as M increases to infinity. However, reducing $\sigma_{\hat{\theta}}^2$ does not really play a key role in decreasing the jitter in sampling-phase τ_k or phase-mismatch θ_k because the low pass filtering effect of the PLL smoothes out the noise in the phase estimator output anyway. Reducing $\sigma_{\hat{\theta}}^2$ is useful for a different reason in our PLL;

it enhances the quality of the decisions in the detector and improves the phase tracking performance within the joint phase estimator/trellis detector.

Increasing M, on the other hand, hampers tracking performance of the joint phase estimator/symbol detector because of the inherent lag associated with the averaging operation implied in (3). A rough rule of thumb in setting the upper limit on M can be established by comparing the path metric computations performed in the presence of a phase error with and without phase estimation and phase error compensation during the branch metric computation. Assume the simple noiseless case. The conventional Viterbi detector is unaware of any phase error and computes the path metric based on

$$P_c = \sum_k (c_k - y_k)^2 \approx \sum_k \theta_k^2 \varepsilon_k^2, \tag{4}$$

where the source of errors in computing the path metric along the correct path is the unexpected phase error sequence. In contrast, the joint scheme computes

$$P_j = \sum_k \left(c_k - y_k - \hat{\theta}_k \varepsilon_k\right)^2 \approx \sum_k \left(\theta_k - \hat{\theta}_k\right)^2 \varepsilon_k^2, \tag{5}$$

where the source of inaccuracy is due to miss-estimation of the phase error sequence. It obviously makes sense to ensure that the estimation error is small enough so that P_j is closer to the true value than P_c is. The ideal value of the path metric in the absence of noise is zero, so this leads to a requirement $P_j < P_c$. Both P_j and P_c are data-dependent as ε_k is data-dependent. If we consider averaging over data, however, the data-dependency disappears, and we arrive at the equivalent condition:

$$\sum_k \left(\theta_k - \hat{\theta}_k\right)^2 < \sum_k \theta_k^2. \tag{6}$$

In the absence of any additive noise, it is easy to see that $\hat{\theta}_k$ is essentially a moving average of θ_k with window M, or roughly a delayed version of θ_k by $M/2$ sample intervals. Now assume a sinusoidal phase error with normalized frequency f_pT with some unknown timing Δ :

$$\theta_k = \cos(2\pi f_p T k - \Delta). \tag{7}$$

The phase estimate can then be expressed as

$$\hat{\theta}_k = \cos(2\pi f_p T(k - M/2) - \Delta). \tag{8}$$

Here we assume there is no DC component (i.e., any frequency offset between the incoming signal and local clock has been removed). Taking expectations on both sides of (6) with respect to Δ, which is assumed to be uniformly distributed over $[0, 2\pi]$, it is straightforward to show

$$\sin^2(\pi f_p T M/2) < 1/4 \tag{9}$$

which leads to roughly

$$\frac{1}{3M} > f_p T. \tag{10}$$

It is worthwhile at this point to mention that the traditional PLL design sets the effective loop bandwidth, commonly denoted as B_LT, above the frequency component of the phase error the PLL intends to track down, i.e.,

$$B_L T > f_p T. \tag{11}$$

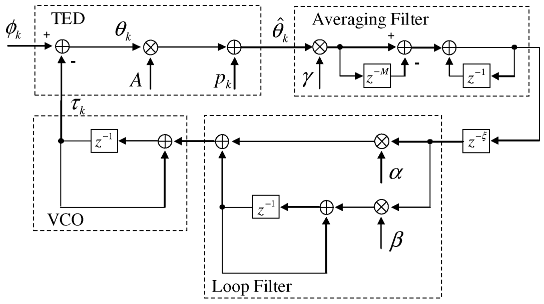

Fig. 3. Proposed discrete-time PLL model.

The comparison of (10) and (11) indicate that $1/3M$ acts like the effective loop bandwidth of the traditional PLL, as far as the tracking speed is concerned. In traditional PLL design, the same B_LT also affects jitter in τ_k (and thus in θ_k) so that the B_LT value must be chosen judiciously to balance between tracking ability and jitter performance.

As we shall show below, in our PLL design the tracking speed and the timing jitter can be separately controlled, avoiding the traditional tradeoff limitations. Namely, B_LT controls jitter in our loop just as in traditional PLL, while the window size M largely determines the tracking ability of the overall loop in our PLL, as argued above.

IV. PLL Loop Analysis

In this section, we try to understand the impact of the new parameter M on stability and jitter performance of the PLL. In steady state, our PLL can be modeled as shown in Fig. 3. The TED is again represented by a linear equation:

$$\hat{\theta}_k = A\theta_k + p_k = A(\phi_k - \tau_k) + p_k. \tag{12}$$

Notice that when $M = 1$, the PLL reduces to the traditional PLL of Fig. 1. The phase estimator of (3) can be viewed as a windowed-average operation acting on $e_k\varepsilon_k$ plus noise. This operation has a transfer function $F(Z) = \gamma(1 - Z^{-M})/(1 - Z^{-1})$. The transfer function of the loop from the incoming phase error ϕ_k to the remaining phase error θ_k at the sampler output is given by

$$\begin{aligned} H_e(z) &= \frac{(1-Z^{-1})^3}{(1-Z^{-1})^3 + A\gamma[\alpha(1-Z^{-1}) + \beta Z^{-1}](1-Z^{-M})Z^{-\xi-1}} \\ &= \frac{(Z-1)^3}{(Z-1)^3 + A\gamma[\alpha(Z-1)+\beta](1-Z^{-M})Z^{-\xi+1}}. \end{aligned} \tag{13}$$

With $M = 1$, $\gamma = 1$ and $\xi = 0$, this expression reduces to that of the classical second-order discrete-time PLL.

A. Stability Versus Window Size and Decision Delay

The stability region can be expressed as the values of α and β that make the transfer function (13) stable (i.e., keep the poles inside the unit circle in Z-domain). In [6], the effect of the decision delay ξ on stability has been investigated and it was shown that increasing decision delay has an effect of shrinking the stability region. It has been shown in [3] that increasing M also

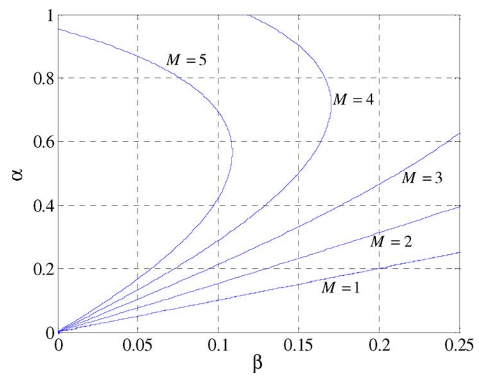

Fig. 4. Stable regions with respect to M when $\xi = 0$.

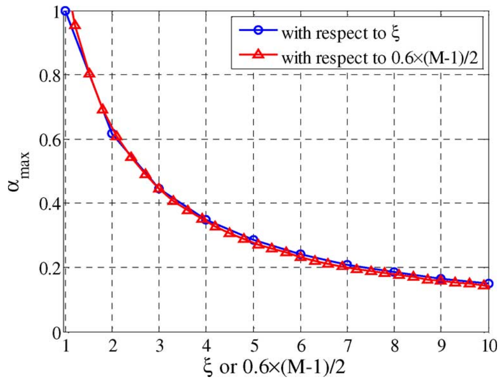

Fig. 5. $\alpha_{\max}$ of stable region.

reduces the stability region (See Fig. 4). The practical values of β are usually orders of magnitude smaller than α. Thus, the stability region in practice depends mainly on α. Thus, to observe how the stability region changes as M or ξ changes, we make a note of the maximum value of α that still retains loop stability with the β value set to zero. In Fig. 5, we show this $\alpha_{\max}$ as a function of $0.6 \times (M-1)/2$ with ξ set to zero as well as a function of ξ with M fixed to 1. The constant 0.6 was chosen empirically to create a good match between the two curves. The observation that the two curves are nearly identical suggests that as far as the stability is concerned, we can define an effective delay $\xi' = 0.6 \times (M-1)2 + \xi$, and assume that ξ' acts like ξ of the traditional DPLL, regardless of specific combination of M and ξ. In fact, although not shown, the contour plots of $\alpha_{\max}$ as functions of $0.6 \times (M-1)/2$ and ξ are near-straight lines, indicating that $0.6 \times (M-1)/2$ and ξ have nearly the same effect on $\alpha_{\max}$. Fig. 6 shows the relationship between $\alpha_{\max}$ and ξ', with β set to yield a damping factor of 1; this plot can be used to quickly check for the stability condition, once the PLL parameters α, M and ξ are chosen.

B. Loop Bandwidth and Jitter

The parameters M and ξ also have a detrimental effect (albeit small) on the timing jitter of the loop. It is straightforward to show that the transfer function of the loop from the effective

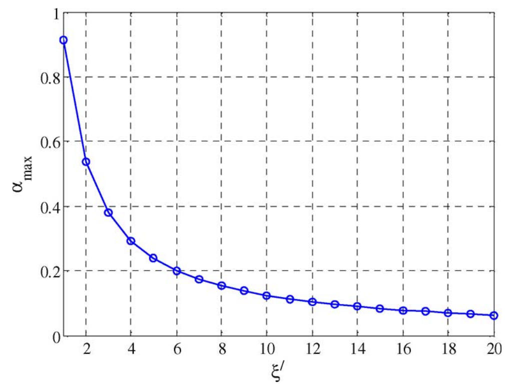

Fig. 6. Relationship between $\alpha_{\max}$ and ξ' with $\zeta = 1$.

additive noise p_k to the residual phase error θ_k, with ϕ_k set to zero, is given by

$$H_t(Z) = \frac{-[\alpha(Z-1)+\beta](1-Z^{-M})}{\gamma^{-1}(Z-1)^3 Z^{\xi-1} + A[\alpha(Z-1)+\beta](1-Z^{-M})}. \tag{14}$$

The normalized loop bandwidth B_LT is defined as [7]

$$B_LT \triangleq \frac{T}{|H_t(Z=1)|^2} \int_0^{1/2T} |H_t(Z=e^{j2\pi fT})|^2 df. \tag{15}$$

Denoting $S_p(Z)$ as the power spectral density of p_k, the variance of the jitter in θ_k is obtained as

$$\begin{aligned}\sigma_\theta^2 &= \int_{-1/2T}^{1/2T} S_p(Z=e^{j2\pi fT})|H_t(Z=e^{j2\pi fT})|^2 df \\ &= 2A^{-1}\sigma_n^2 B_LT \end{aligned} \tag{16}$$

where the last equality results from the assumption that $S_p(Z) = \sigma_n^2\sigma_x^2\eta$. The jitter variance is proportional to B_LT, and we need to make B_LT small to maintain low jitter. The loop bandwidth B_LT is clearly a function of $\{\alpha, \beta, M, \xi\}$. For continuous-time PLL, which obviously has no dependency on M and ξ, it is well-known that $B_LT = \alpha/2 + \beta/2\alpha$ [8]. Assuming the typical scenario of $\alpha >> \beta$, the continuous-time PLL's bandwidth is then $B_LT \approx \alpha/2$. We observe that the numerically computed B_LT of a discrete-time PLL as a function of $\alpha/2$ also exhibits a clear linear behavior.

Now to understand the impact of M and ξ on B_LT, we plot B_LT as a function of $0.8 \times (M-1)/2$ (with ξ fixed to 0) as well as of ξ (with M fixed to 1) in Fig. 7 while setting $\alpha = 0.01$ and $\beta = \alpha^2/2$. The coefficient 0.8 was found empirically. The two curves roughly match, indicating that $0.8 \times (M-1)/2$ has a similar impact on jitter as ξ has. The contours of B_LT values plotted against $0.8 \times (M-1)/2$ and ξ indicate that B_LT is indeed proportional to $\xi'' = 0.8 \times (M-1)/2 + \xi$.

Based on the contour plots of B_LT given as functions of $\alpha/2$ and ξ'', we found the following empirical relationship:

$$B_LT \approx \frac{\alpha}{2} + c \cdot 10^{-5} \cdot \xi'' - 3 \times 10^{-4} \tag{17}$$

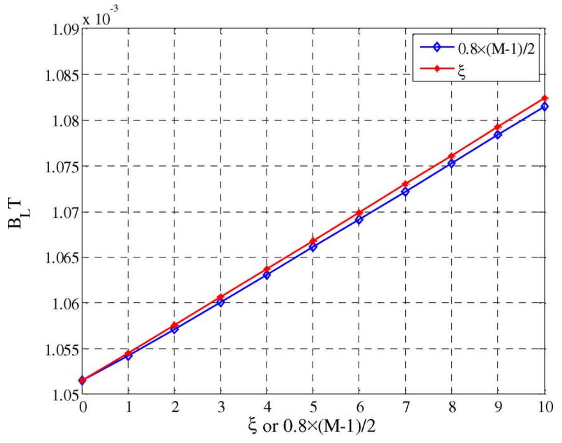

Fig. 7. B_LT of a discrete-time PLL.

with $c = 2.5$, where B_LT depends mostly on $\alpha/2$ but also is a function (albeit weaker) of ξ''.

Note that the empirically found constants in (17) and in the expressions for ξ' and ξ'' (the former of which led to the $\alpha_{\max}$ plot in Fig. 6) depend on γ, which in turn is a function of the channel-dependent parameter $\eta = \sum_i g_i^2$. This implies that the set of constants would be different in general for different channel responses. The particular set of constants obtained thus far reflect the extended partial response class IV (EPR4) channel, as will be revisited in Section VI.

V. New PLL Design Guideline

Based on the above analysis and observations, we now put together an overall design guideline for our PLL. Assume that we know the normalized target frequency f_pT component we want our PLL to track in the incoming phase error. We first set M such that (10) is satisfied. At the same time, in order to ensure a good phase tracking capability of the detector, M should also be reasonably larger than 1. Our experience indicates M should be well above 10.

Once M is determined, we set ξ to be a fairly small value. Unlike in traditional DPLL, where ξ must be large enough to guarantee high quality decisions, in our scheme a very small value (e.g., $\xi = 1$) provides adequate quality of the phase estimate that drives the loop filter. This is because the delayed-phase estimate chosen out of the best up-to-date path provides reliable phase estimates even with small delays. Once M and ξ are both chosen, compute the effective decision delays $\xi' = 0.6\times(M-1)/2+\xi$ and $\xi'' = 0.8\times(M-1)/2+\xi$, which will later be used to check the stability condition and jitter performance, respectively.

Now we discuss selecting B_LT that will mainly control the jitter performance. It is clear that we should choose α as small as possible to minimize jitter. Again we emphasize that selecting a very small value of α is possible in our case, as we can control the tracking speed largely based on the parameter M. This would not be possible in traditional PLL design. The lower bound on α comes into the picture from the stability standpoint. Obviously setting $\alpha = 0$ will make the PLL unstable. Thus, we choose α close to zero with some safety margin. In our numerical examples, we often set α to be one-tenth of the B_LT value chosen in the standard DPLL.

As for choosing β, while forcing $\beta = 0$ does not cause a stability issue, it is not desirable in standard PLL design. This is because setting $\beta = 0$ will hamper the PLL's ability to drive the phase error (the TED output) to zero, when there is a frequency mismatch between the incoming signal and the local clock. In our PLL, however, this does not cause a difficulty in principle, since any finite phase error can be dealt with in the joint phase estimator and symbol detector. When the residual phase error is large enough in practice, however, it can make the first-order approximation in (2) we use in modeling the phase-error effect on the signal amplitude less accurate. For this reason, it is still beneficial to try to eliminate phase error in the presence of a frequency-step in the incoming signal. This means we would still want to maintain a small positive value for β. Making β increasingly smaller relative to α has an effect of increasing the damping ratio. In terms of the transient phase response of the loop to a step-frequency input, a very large damping ratio is manifested as a phase error that may be small but dies out very slowly. Since any residual phase error in the PLL degrades the decision quality of the detector, this is not acceptable in a conventional PLL. In our PLL, however, the damping ratio can be allowed very large, because of the phase-tracking ability of our detector.

To summarize the new PLL design guidelines given above, let B_LT_{TPLL} be the loop bandwidth of the traditional PLL, which is typically chosen to be above the highest targeted frequency component of ϕ_k. We then choose M and α values of our loop such that

$$\frac{\alpha}{2} < B_LT_{\mathrm{TPLL}} < \frac{1}{3M}. \tag{18}$$

In this way, we have higher tracking speed while our jitter is lower, relative to the traditional PLL. We then make β much smaller than α. For example, if $\alpha = 0.002$, then setting $\beta = 1.0\times10^{-6}$ gives a damping ratio of $\zeta = \alpha/2\sqrt{\beta} = 1$. The choice $\beta = 2.5\times10^{-7}$ yields $\zeta = 2$. Note that in traditional PLL design, the damping ratio is typically kept between 0.5 and 1. Once M is chosen, select ξ to be something small, say, $\xi = 1$. As a final step, check the stability condition and the overall jitter amount against the chosen value of $\xi' = 0.6\times(M-1)/2+\xi$ and $\xi'' = 0.8\times(M-1)/2+\xi$, using the stability plot of Fig. 6 and the B_LT in (17).

VI. Numerical Results

For the numerical results, the channel impulse response is assumed to be EPR4, modeling the high density tape channel, where $f(t) = p_q(t) + p_q(t-T) - p_q(t-2T) - p_q(t-3T)$ with $p_q(t)$ denoting a zero-ISI pulse with 0% excess band. The length of the "derivative" response $\{g_j\}$ used to construct the expanded trellis, which is needed for applying the branch-specific phase-estimating scheme, is six, counting only significant terms. This means the size of our trellis is 32 as opposed to 8 in the normal Viterbi detector matched to the EPR4 channel. There are also extra computation (see (3)) and storage required for branch-specific phase estimation but this effect is minimal. The main issue is the increased size of the trellis, which represents roughly a $4\times$ increase in the detector complexity. This is basically the price that is paid in return for the excellent performance and robustness against timing errors. We note, however, that standard trellis-reduction strategies like local decision feedback [9] can help reduce complexity. Using the design guideline developed above, we present numerical examples based on two different phase error models.

A. Ramp Plus Sinusoid Phase Error

Assume that the incoming phase error is modeled as

$$\phi_k = kf_eT + p_eT\sin(2\pi kf_pT) \tag{19}$$

where f_eT represents the normalized constant frequency offset, p_eT is the normalized peak phase fluctuation and f_pT is the normalized frequency of the phase fluctuation. While there is no explicit random noise component in (19), the additive channel noise n_k effectively injects via p_k a random noise component in the phase-domain model of the loop.

Assume $f_eT = 0.001, p_eT = 0.1$ and $f_pT = 0.001$. For traditional PLL, we set the loop bandwidth B_LT to $4 \times f_pT$ (so that $\alpha/2$ is roughly 0.004 or $\alpha = 0.008$). Increasing α beyond this value improves the tracking speed but results in a larger jitter. Choosing a smaller value makes tracking difficult within a few thousand bits. This choice of B_LT sets the guideline for our loop parameters:

$$\frac{\alpha}{2} < 0.004 < \frac{1}{3M}. \tag{20}$$

We try $\alpha/2 = 0.002$ for our loop, one-half the value for the traditional PLL, which sets $\alpha = 0.004$. We also set $M = 41$ so that $1/3M$ is roughly twice as large as the traditional loop's B_LT of 0.004. We subsequently fix $\xi = 1$, which gives $\xi'' = 17$. With $\beta = 1.0 \times 10^{-6}$, we get the damping ratio of $\zeta = 2$.

For the traditional PLL based on Mueller-Muller (MM) TED [10] and a standard Viterbi detector, we set the decision delay to $\xi = 11$ and $\alpha = 0.008$. To achieve the popular damping ratio of $1/\sqrt{2} \approx 0.707$, we choose $\beta = \alpha^2/2 = 3.2 \times 10^{-5}$.

Fig. 8 shows a simulated tracking speed comparison of the two systems. Our loop achieves a zero-frequency error much earlier than the traditional loop. The residual phase error in the traditional loop goes to zero after 1200 samples, but the phase error in the proposed loop decays very slowly due to the large damping ratio. However, the residual phase error does not matter in the proposed loop because the joint detector can compensate for the finite residual phase error during data detection. We can reduce the tracking time in the traditional loop by increasing the loop bandwidth, but this also increases the bit-error and the cycle-slip probabilities as well as jitter.

Table I shows simulated jitter variances as a function of SNR, after both schemes track down the initial frequency offset. The SNR is defined as $\Sigma_i f_i^2/2\sigma_n^2$, where f_i denotes the channel response and σ_n^2 the noise variance. The jitter variance of the proposed loop is consistently better than that of the traditional loop. We safely conclude that for a given conventional PLL setting, our loop can be set so that the latter has a better tracking speed while at the same time yielding a smaller steady-state jitter.

B. Random Walk Phase Error

We now use a random walk phase error model in comparing frame error rates (FERs):

$$\phi_k = \sigma_w \sum_{i=0}^{k} w_i \tag{21}$$

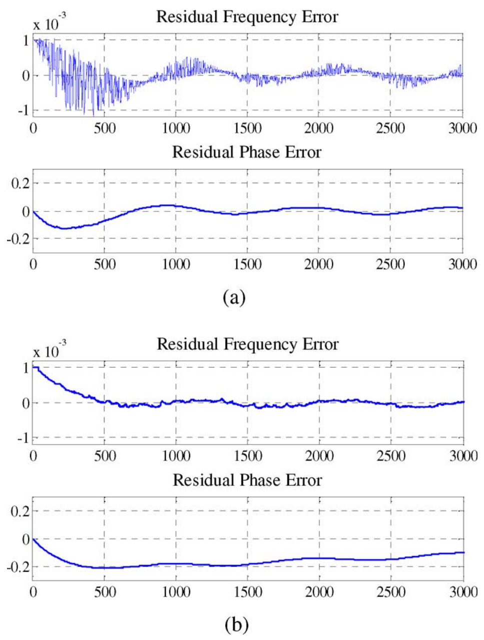

Fig. 8. Tracking speed comparison. (a) Traditional loop. (b) Proposed loop.

TABLE I
JITTER VARIANCE COMPARISON ($\times 10^{-4}$)

	SNR(dB)				
Loop Type	6	7	8	9	10
Traditional	6.12	5.87	5.31	4.81	4.04
Proposed	3.12	2.26	1.94	1.82	1.51

where w_i is a unit-variance white Gaussian noise process and σ_w is an adjustable parameter that controls the overall phase error variance. For our simulation, we fix $\sigma_w = 0.0075T$.

For the traditional PLL, we set the decision delay to $\xi = 11, \alpha = 0.02$ and $\beta = \alpha^2/2 = 2.0 \times 10^{-4}$, which correspond to the lowest FER at SNR $= 10$ dB. For our loop, we set $\alpha = 0.002$ and $M = 26$. We subsequently fix $\xi = 1$, which gives $\xi'' = 11$. With $\beta = 2.5 \times 10^{-7}$, we get the damping ratio of $\zeta = 2$. Fig. 9 compares the FER performance between the two loops. Each frame contains 4096 bits, and a frame error is declared if the number of bit errors in a frame is greater than 40 bits (assuming a reasonable error correction coding capability). As can be seen in the FER comparison, the proposed loop shows a significant performance improvement over the traditional loop. The conventional loop suffers from frequent cycle-slips and many bit errors due to the large loop bandwidth and inaccurate decisions in the Viterbi detector. Making the bandwidth smaller does not help either because in that case the reduced tracking ability starts to create issues.

C. Turbo Equalizer Setting and Performance

The new PLL structure can be applied to turbo equalization by incorporating the joint timing and data estimator into the BCJR algorithm of [11]. One possible way to incorporate the joint timing and data recovery scheme in the BCJR algorithm is to compute the phase estimate $\hat{\theta}_k(\nu_l)$ according to (3) for each branch in the beginning of every cycle during forward recursion. To use the phase update equation of (3), then a "survivor" path

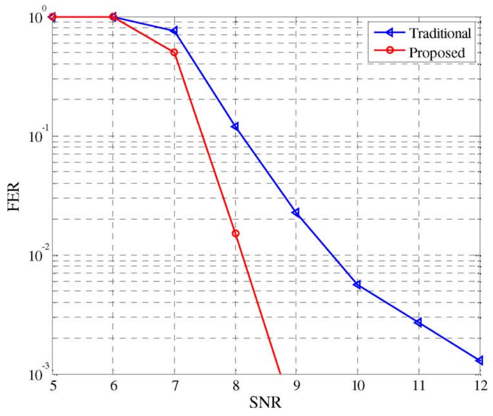

Fig. 9. FER comparison with random walk phase error.

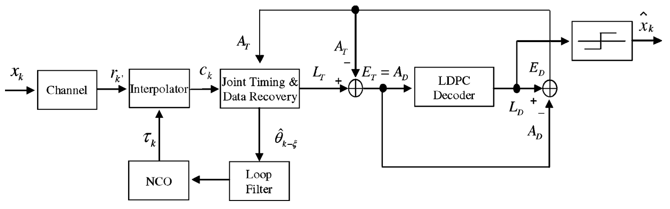

Fig. 10. Block diagram of the turbo equalizer setting.

needs to be maintained for each state, as in the standard Viterbi detector. This can be achieved by performing, during the forward recursion, an additional Viterbi-like path arbitration via path metric updates and selection of one path per state.

The block diagram of a turbo equalizer setting is shown in Fig. 10, where the global timing recovery loop is composed of an interpolator, a joint timing and data recovery block, a loop filter, and a numerically-controlled oscillator (NCO), operating on the over-sampled channel output $r_{k'}$ and soft information A_T generated from a low density parity check (LDPC) code decoder. For the LDPC, the quasi-cyclic low density parity check (QC-LDPC) code of [12] is chosen with parameters: code length $N = 4572$, code rate $R = 0.92$, row weight 3 and column weight 36. The message-passing LDPC decoder runs with soft information A_D from the BCJR detector and then passes back its outputs to the BCJR algorithm; this iterative process continues until the number of iterations reaches a certain number (5 being chosen for the BER simulations). We run LDPC internal iterations up to 25.

The normalized phase fluctuation model of (19) is used. The peak phase p_eT is set to 0.2, and the period f_pT to 0.001. The frequency offset f_eT is set to 0 to avoid the need for initial tracking. The loop parameters are adjusted to best track the phase fluctuation $f_pT = 0.001 : \xi = 1, \alpha = 0.002, \beta = 2.5\times10^{-7}$, and $M = 56$. The BER simulation results are shown in Fig. 11. Our loop achieves the performance of ideal timing within a fraction of dB.

VII. Conclusion

We provided analytical tools to systematically design the recently proposed PLL that incorporates the phase-tracking detector of [2], [3]. A design guideline has been constructed for this PLL so that it can achieve both good tracking and low jitter,

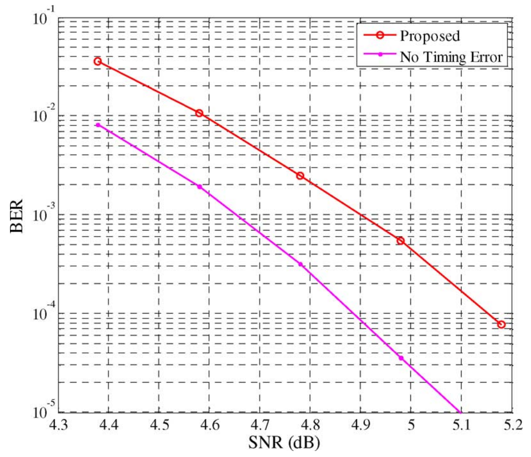

Fig. 11. BER performance under turbo equalizer setting.

a feat that is not possible with traditional PLLs. Tracking speed simulation and error rate simulation under traditional hard-decision processing as well as a turbo equalization environment validate the new PLL and the design guideline presented in this paper.

Acknowledgment

This work was supported in part by the National Science Foundation under Grant IHCS/GOALI 0701946. The earlier work on phase-compensating detector was supported in part by INSIC Tape Program and Samsung. The authors are grateful to S. Oelcer, who reviewed an early draft and provided valuable feedback. The authors also acknowledge Hui Jin's participation during the very early phase of the PLL work.

References

[1] J. W. M. Bergmans, *Digital Baseband Transmission and Recording*. Boston, MA: Kluwer, 1996, pp. 591–624.
[2] J. Moon and J. Lee, "Joint timing recovery and data detection with applications to magnetic recording," *IEEE Trans. Magn.*, vol. 42, no. 10, pp. 2576–2578, Oct. 2006.
[3] J. Moon and J. Lee, "Timing recovery in conjunction with maximum likelihood sequence detection in the presence of intersymbol interference," *IEEE Trans. Circuits Syst. I*, vol. 55, no. 10, pp. 2884–2897, Oct. 2008.
[4] G. G. Forney Jr., "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.
[5] P. Kovintavewat, J. R. Barry, M. F. Erden, and E. M. Kurtas, "Per-survivor timing recovery for uncoded partial response channels," *IEEE ICC*, vol. 27, no. 1, pp. 2715–2719, Jun. 2004.
[6] J. W. M. Bergmans, "Effect of loop delay on stability of discrete-time PLL," *IEEE Trans. Circuits Syst. I*, vol. CAS-42, no. 4, pp. 229–231, Apr. 1995.
[7] P. M. Aziz and S. Surendran, "Symbol rate timing recovery for higher order partial response channels," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 4, pp. 635–648, Apr. 2001.
[8] F. M. Gardner, *Phaselock Techniques*. New York: Wiley, 1966.
[9] M. V. Eyuboglu and S. U. H. Qureshi, "Reduced-state sequence estimation with set partitioning and decision feedback," *IEEE Trans. Commun.*, vol. 36, no. 1, pp. 13–20, Jan. 1988.
[10] R. D. Cideciyan, F. Dolivo, R. Hermann, W. Hirt, and W. Schott, "A PRML system for digital magnetic recording," *IEEE J. Sel. Areas Commun.*, vol. 10, no. 1, pp. 38–56, Jan. 1992.
[11] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. IT-20, no. 2, pp. 248–287, Mar. 1974.
[12] R. M. Tanner, D. Sridhara, A. Sridharan, T. E. Fuja, and D. J. Costello Jr., "LDPC block and convolutional codes based on circulant matrices," *IEEE Trans. Inf. Theory*, vol. IT-50, no. 12, pp. 2966–2984, Dec. 2004.