Iterative Timing Recovery with the Split-Preamble Strategy for Coded Partial Response Channels

Article *in* IEICE Transactions on Electronics · March 2011

DOI: 10.1587/transele.E94.C.368 · Source: DBLP

3 authors:

Chanon Warisarn
King Mongkut's Institute of Technology Ladkrabang
68 PUBLICATIONS **77** CITATIONS

SEE PROFILE

Piya Kovintavewat
Nakhon Pathom Rajabhat University
108 PUBLICATIONS **275** CITATIONS

SEE PROFILE

Pornchai Supnithi
King Mongkut's Institute of Technology Ladkrabang
140 PUBLICATIONS **411** CITATIONS

SEE PROFILE

IEICE TRANS. ELECTRON., VOL.E94–C, NO.3 MARCH 2011

PAPER

Iterative Timing Recovery with the Split-Preamble Strategy for Coded Partial Response Channels

Chanon WARISARN[†a)], ***Nonmember***, **Piya KOVINTAVEWAT**[††], ***and*** **Pornchai SUPNITHI**[†], ***Members***

SUMMARY This paper proposes a modified per-survivor iterative timing recovery scheme, which exploits a new split-preamble strategy in conjunction with a per-survivor processing soft-output Viterbi algorithm (PSP-SOVA). The conventional split-preamble strategy places a preamble at the beginning of a data sector and uses it to run a phase-locked loop during acquisition to find an initial phase/frequency offset. However, the proposed scheme splits the preamble into two parts. The first part is placed at the beginning of the data sector, whereas the second part is divided into small clusters, each of which is then embedded uniformly within the data stream. This split preamble is utilized to adjust the branch metric calculation in PSP-SOVA to ensure that the survivor path occurs in a correct direction. Results indicate that the proposed scheme yields a better performance than a conventional receiver with separate timing recovery and turbo equalization, and the iterative timing recovery scheme proposed in [1], [2], especially when the timing jitter is large. In addition, we also show that the proposed scheme can automatically correct a cycle slip much more efficiently than the others.

key words: *iterative timing recovery, per-survivor iterative timing recovery, split-preamble strategy, timing acquisition*

1. Introduction

Timing recovery is the process of synchronizing the sampler with the received analog signal. Sampling at the right times is critical to achieving good overall performance. Therefore, the quality of synchronization is very important for all applications. Generally, conventional timing recovery employs a 2nd-order phase-locked loop (PLL), which consists of a timing error detector (TED), a loop filter, and a voltage controlled oscillator (VCO). A widely used TED is the Mueller and Müller (M&M) TED [3]. In practice, the M&M TED performs well at high signal-to-noise ratio (SNR). However, at low SNR, the system is plagued by cycle slips and severe timing jitter. To solve this problem, the MAP-based timing recovery has been proposed in [4], which uses a new TED algorithm. This TED employs maximum *a-posteriori* and minimum mean-squared error techniques to estimate timing information. Results have indicated that it can reduce a cycle-slip rate if compared to conventional timing recovery at the expense of increased computation complexity. It has been shown in [4] that the complexity of this new TED is about 4 times higher than that of the M&M TED.

Practically, timing recovery performs in two modes, namely, acquisition and tracking modes [5]. The acquisition mode is performed to acquire the initial timing offset with an aid of a preamble (or a training sequence of known symbols), whereas the tracking mode is performed to refine the timing estimates based on a user data sequence (unknown symbols). To improve the performance of acquisition mode, Hwang et al. [6] proposed an extended Kalman filter (EKF) to estimate and correct possibly large initial timing errors. This proposed method converges faster than a conventional timing recovery scheme, thus allowing us to reduce the length of the preamble used during acquisition mode.

It should be noted that all timing recovery architectures mentioned above performs timing recovery and maximum-likelihood (ML) equalization separately, thus performing unreliably when operating at low SNR and severe timing jitter. To improve their performances, Kovintavewat et al. [7] proposed a per-survivor timing recovery architecture, which performs timing recovery and ML equalization *jointly*. Furthermore, Zeng et al. [8] also proposed a trellis-based optimal baud-rate timing recovery loop based on a Markov model, whose structure is similar to [7]. This scheme has been shown to perform well at the expense of high complexity. Nonetheless, both schemes have been proposed for partial-response (PR) channels without error-correction codes (ECCs).

The large coding gains of iterative ECCs enable reliable communication at very low SNR. This means that timing recovery must be performed at an SNR lower than ever before. A conventional receiver performs timing recovery and turbo equalization [9] *separately*. Specifically, conventional timing recovery ignores the presence of ECCs and thus fails to work properly when the SNR is low enough. To improve the performance of the conventional receiver, Kovintavewat et al. [1] proposed a full-complexity per-survivor iterative timing recovery (Full PS-ITR) scheme, which *jointly* performs timing recovery, equalization, and error-correction decoding. It is realized by first applying the per-survivor processing (PSP) technique [10], to the Bahl, Cocke, Jelinek, and Raviv (BCJR) algorithm [11], resulting in a per-survivor BCJR equalizer, denoted as "PSP-BCJR" [1]. Then, PSP-BCJR iteratively exchanges soft information with a soft-in soft-out (SISO) decoder. Because of the PSP-BCJR has very high complexity, a reduced-complexity per-survivor iterative timing recovery (PS-ITR) scheme has

Manuscript received January 12, 2010.
Manuscript revised September 26, 2010.

[†]The authors are with the Faculty of Engineering and I/UCRC in Data Storage Technology and Applications, King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand.

[††]The author is with the Data Storage Technology Research Unit, Nakhon Pathom Rajabhat University, Nakhon Pathom 73000, Thailand.

a) E-mail: s9060053@kmitl.ac.th

DOI: 10.1587/transele.E94.C.368

been proposed [2], where a soft-output Viterbi algorithm (SOVA) [12] is used to instead of BCJR, resulting in a per-survivor SOVA equalizer, denoted as "PSP-SOVA." As investigated in [2], at low to moderate complexity, the PS-ITR performs better than both the Full PS-ITR and the conventional receiver. This is because the PS-ITR can automatically correct a cycle slip [5] with only a small number of turbo iterations.

Generally, to have good timing estimates before the user data sequence starts, the conventional split-preamble strategy places all known symbols at the beginning of a data sector [13]. Nonetheless, in [14], [15], the known symbols are arranged in small contiguous clusters and are placed periodically in the data stream, subject to the power constraint on training. Additionally, Nayak et al. [16] proposed the optimal training symbol placement strategy to minimize a Cramér-Rao bound by splitting the known symbols into two halves and placing them at the beginning and at the end of a data sector. This split-preamble arrangement leads to a reduced frequency estimation error variance and greatly reduces the occurrence of cycle slips.

In general, the PS-ITR uses the conventional split-preamble strategy, which places a preamble at the beginning of a data sector and uses it to run a PLL [5] during acquisition to find the initial timing offset. To further improve the performance of the PS-ITR, we propose a *modified* per-survivor iterative timing recovery (MPS-ITR) scheme. In essence, the MPS-ITR performs same as the PS-ITR, except that a *modified* PSP-SOVA is utilized instead of PSP-SOVA. This modified PSP-SOVA employs a new split-preamble strategy, which splits a preamble into two parts. The first part is placed at the beginning of a data sector, while the second part is divided into many small clusters, each of which is therefore embedded uniformly within the user data sequence. This split preamble is utilized to adjust the branch metric calculation in PSP-SOVA to guarantee that the survivor path occurs in a correct direction.

This paper is organized as follows. After explaining the system model in Sect. 2, Sect. 3 proposes the modified PSP-SOVA and describes how it works. The complexity comparison of different iterative timing recovery schemes is given in Sect. 4. Section 5 compares the performance of different iterative timing recovery schemes. Finally, Sect. 6 concludes this paper.

2. System Description

Consider the coded PR channel in Fig. 1, where $H(D) = \sum_{k=0}^{\nu-1} h_k D^k = 1 + 2D + D^2$ is a PR2 channel, D is the delay operator, and ν is channel memory. The message $x_k \in \{0, 1\}$ is encoded by an ECC encoder and is mapped to a binary sequence $a_k \in \{\pm 1\}$ of length L. Next, a preamble is inserted in a sequence a_k to obtain a sequence $b_k \in \{\pm 1\}$. The readback signal can then be written as

$$p(t) = \sum\nolimits_k r_k q(t - kT - \tau_k) + n(t), \tag{1}$$

where $r_k = b_k * h_k \in \{0, \pm 2, \pm 4\}$ is the noiseless channel output, $*$ is the convolution operator, $q(t) = \sin(\pi t/T)/(\pi t/T)$ is an ideal zero-excess-bandwidth Nyquist pulse, T is a bit period, and $n(t)$ is an additive white Gaussian noise with a two-sided power spectral density $N_0/2$. The uncertainty in the timing is captured by the timing offset τ_k, which is modeled as a random walk [17] according to $\tau_{k+1} = \tau_k + \mathcal{N}\left(0, \sigma_w^2\right)$, where σ_w controls the severity of the timing jitter. The random walk model is chosen because of its simplicity and its ability to represent a variety of channels by changing only one parameter.

At the receiver, the readback signal $p(t)$ is filtered by an ideal low-pass filter (LPF), whose impulse response is $q(t)/T$, to eliminate the out-of-band noise, and is then sampled at time $kT + \hat{\tau}_k$, creating

$$\begin{aligned} y_k &= y(kT + \hat{\tau}_k) \\ &= \sum\nolimits_i r_i q(kT + \hat{\tau}_k - iT - \tau_i) + n_k, \end{aligned} \tag{2}$$

where $\hat{\tau}_k$ is the receiver's estimate of τ_k, and n_k is i.i.d. zero-mean Gaussian random variable with variance $\sigma_n^2 = N_0/(2T)$, i.e., $n_k \sim \mathcal{N}\left(0, \sigma_n^2\right)$.

Conventional timing recovery is based on a 2nd-order PLL [5], which consists of a timing error detector (TED), a loop filter, and a voltage-controlled oscillator (VCO). A decision-directed TED computes the receiver's estimate of the timing error $\varepsilon_k = \tau_k - \hat{\tau}_k$ using the well-known M&M TED algorithm [3] according to

$$\hat{\varepsilon}_k = \frac{6T}{40}\left\{y_k \hat{r}_{k-1} - y_{k-1}\hat{r}_k\right\}, \tag{3}$$

where $\hat{r}_k$ is an estimate of r_k, and the constant $6T/40$ [18] is used to normalize the timing function of the M&M TED in (3) to have unit slope at origin [5]. For simplicity, we consider the M&M TED algorithm because its complexity is lower than the TED proposed in [4]. We also assume no frequency offset in the system. Thus, the next sampling phase offset can be updated by a 1st-order PLL according to

$$\hat{\tau}_{k+1} = \hat{\tau}_k + \alpha\hat{\varepsilon}_k, \tag{4}$$

where α is a PLL gain parameter [5].

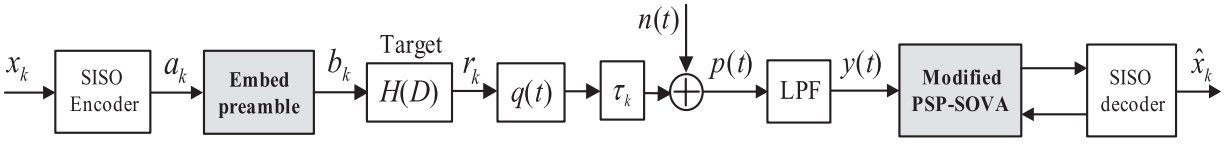

Fig. 1 A channel model with the modified per-survivor iterative timing recovery scheme.

In a conventional setting, conventional timing recovery is followed by a turbo equalizer [9], which iteratively exchanges soft information between the SISO equalizer for the PR2 channel and the SISO decoder for the outer code.

3. Modified PSP-SOVA

The modified PSP-SOVA is developed based on the PSP-SOVA [2] with an aid of the new split-preamble strategy. Specifically, the PSP-SOVA uses the conventional preamble arrangement, which places all C known bits at the beginning of the data sector as shown in Fig. 2(a). We propose a new split-preamble strategy, which splits a C-bit preamble into two parts. The first part of $C/2$ bits is placed at the beginning of the data sector, and the second part of $C/2$ bits is divided into $C/(2m)$ clusters (e.g., m = 1, 2, or 4 bits), each of which is then embedded uniformly within the user data stream as illustrated in Fig. 2(b). This split preamble is utilized to adjust the branch metric calculation in PSP-SOVA to ensure that the survivor path occurs in a correct direction. Based on extensive simulation search, we found that modified PSP-SOVA with the one-bit split-preamble arrangement (i.e., m = 1) provides the best performance.

Figure 3 shows the modified PSP-SOVA algorithm, where (A-7) starting with * is an additional step beyond the PSP-SOVA algorithm, and a constant $6T/40$ in (A-11) is only for the PR2 channel, which can be included in the PLL gain parameter. It should be noted that the modified PSP-SOVA works in a *same* manner as PSP-SOVA [2] does, except that the modified PSP-SOVA has an *extra* step to account for the split preamble, which can be briefly described as follows.

Denote $f_k = \{\pm 1\}$ as a $C/2$-bit preamble that is embedded uniformly within the user data stream at the i-th position. In other words, each m-bit preamble is inserted at every i-th data bit, where $i = \lfloor 8190\,\mathrm{m/C} \rfloor$ is the lowest integer close to 8190 m/C. Following the notations in [2], during the branch metric calculation at each k-th stage, we check the condition in (A-7) to adjust the branch metrics $\rho_k(p,q)$. Specifically, at the k-th stage, if $\hat{b}_k(p,q) \neq f_i$, where $\hat{b}_k(p,q) \in \{\pm 1\}$ is the data bit that corresponds to the state transition from state p to state q, and f_i is the preamble bit at the i-th position, we then set $\rho_k(p,q) = \Delta$, where Δ is a large number to guarantee that the modified SOVA will not choose this branch as part of a survivor path.

In a practical receiver, a preamble is used not only for timing recovery, but also for automatic gain control (AGC) [19]. This AGC is utilized to adjust the amplitude of the readback signal such that its amplitude remains constant throughout the data packet. From the simulation result on conventional timing recovery (not shown here), based on the same algorithm as in [19], but for PR2 signals, we found that the number of required preamble bits for convergence of the gain coefficient is less than 50 bits, and well below 128 preamble bits proposed in this work. Therefore, a shorter preamble in this case does not affect the overall performance of timing recovery system.

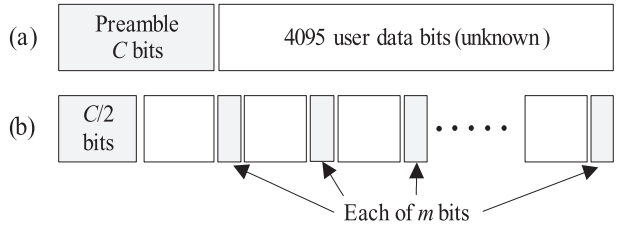

Fig. 2 (a) the conventional preamble arrangement, and (b) the proposed split-preamble strategy.

(A-1) Initialize $\mathbf{\Phi}_0(p) = 0$ for $\forall p$
(A-2) Initialize $\hat{\tau}_0(p) = 0$ for $\forall p$
(A-3) For $k = 0,1,...,L+\upsilon-1$
(A-4) For $q = 0,1,...,Q-1$
(A-5) $y_k(p) = y_k\left(kT + \hat{\tau}_k(p)\right)$ for $\forall p$
(A-6) $\rho_k(p,q) = \left|y_k(p) - \hat{r}(p,q)\right|^2$ for $\forall p$
*(A-7) If $k = i$ then [i is the preamble position]
For $p = 0,1,...,Q-1$
If $\hat{b}(p,q) \neq f_i$ then
$\rho_k(p,q) = \Delta$ [Δ is a large number]
End
End
(A-8) $\pi_{k+1}(q) = \arg\min_p \left\{\mathbf{\Phi}_k(p) + \rho_k(p,q)\right\}$
(A-9) $\mathbf{\Phi}_{k+1}(q) = \mathbf{\Phi}_k\left(\pi_{k+1}(q)\right) + \rho_k\left(\pi_{k+1}(q), q\right)$
(A-10) $\mathbf{S}_{k+1}(q) = \left[\mathbf{S}_k\left(\pi_{k+1}(q)\right) \middle| \pi_{k+1}(q)\right]$
(A-11) $\hat{\varepsilon} = \frac{6T}{40}\Big\{y_k\left(\pi_{k+1}(q)\right)\hat{r}\left(\pi_k\left(\pi_{k+1}(q)\right), \pi_{k+1}(q)\right)$
$- y_{k-1}\left(\pi_k\left(\pi_{k+1}(q)\right)\right)\hat{r}\left(\pi_{k+1}(q), q\right)\Big\}$
(A-12) $\hat{\tau}_{k+1}(q) = \hat{\tau}_k\left(\pi_{k+1}(q)\right) + \alpha\hat{\varepsilon}$
(A-13) $\Delta_{k+1}(q) = \max_p \left\{\mathbf{\Phi}_k(p) + \rho_k(p,q)\right\} - \mathbf{\Phi}_{k+1}(p)$
(A-14) Initialize $\hat{L}_k(q) = +\infty$ [Soft decision update]
(A-15) For $j = k - \nu,...,k-\delta$
Compare the two paths merging
in state q (i.e., $\mathbf{\Psi}_{k+1} = q$)
If $\hat{a}_j^{(1)}\left(\mathbf{\Psi}_{j+1}\right) \neq \hat{a}_j^{(2)}\left(\mathbf{\Psi}_{j+1}\right)$ then
Update $\hat{L}_j\left(\mathbf{\Psi}_{j+1}\right) = \min\left(\hat{L}_j\left(\mathbf{\Psi}_{j+1}\right), \Delta_{k+1}(q)\right)$
End
(A-15) if $k \geq \delta$ then
Output the soft decision according to $\lambda_k^p = \hat{a}_{k-\delta}\hat{L}_{k-\delta}$, which
can be extracted from the survivor path that minimizes $\mathbf{\Phi}_{k+1}$
End
End

Fig. 3 The modified PSP-SOVA algorithm, where (A-7) is an additional step beyond the PSP-SOVA.

4. Complexity Comparison

To measure the complexity of iterative timing recovery schemes, we consider the total number of additions and mul-

Table 1 The total number of operations (per bit) of each function used in the MPS-ITR scheme.

Module	Number of operations (**per bit**)	
	Addition	Multiplication
Ideal sinc interpolation filter	$(4N_{sinc}\text{-}1)Q$	$(N_{sinc})Q$
1st-order PLL	Q	Q
SOVA	$7Q+\frac{\delta^2+9\delta+9}{2}+1$	$6Q+1$
modified PSP-SOVA	$(7+4N_{sinc})Q+\frac{\delta^2+9\delta+9}{2}+1$	$(7+N_{sinc})Q+1$
LDPC decoder	$(1+(k-1)(1-R))N_{in}+1$	$(1-R)N_{in}$

Table 2 The total number of operations (per bit) of each function used in the full PS-ITR scheme.

Module	Number of operations (**per bit**)	
	Addition	Multiplication
Ideal sinc interpolation filter	$2(4N_{sinc}\text{-}1)Q$	$2(N_{sinc})Q$
1st-order PLL	$2Q$	$2Q$
BCJR	$12Q\text{-}2$	$20Q+1$
PSP-BCJR	$(10+8N_{sinc})Q\text{-}2$	$(22+2N_{sinc})Q+1$
LDPC decoder	$(1+(k-1)(1-R))N_{in}+1$	$(1-R)N_{in}$

Table 3 Complexity (per bit) of different iterative timing recovery schemes, where N is the number of turbo iterations.

System	Number of operations (**per bit**)	
	Addition	Multiplication
Conventional receiver	$27+223.94N$	$9+25.56N$
Full PS-ITR scheme	$730.44N$	$257.56N$
PS-ITR scheme	$569.94N$	$113.56N$
MPS-ITR scheme	$569.94N$	$113.56N$

tiplications as a criterion used in each scheme. For other mathematical functions, such as $\log(x)$, $\exp(x)$, and etc., we assume they can be implemented as lookup tables, and that we ignore their complexity. Note that we attempt to fairly count the number of operations (both addition and multiplication) for each scheme such that the memory requirement is minimized.

It can be shown that the complexity of each component is given in Table 1 for modified PSP-SOVA and Table 2 for PSP-BCJR, where N_{sinc} is the total number of ideal sinc interpolation filter taps used to sample the analog signal and to refine the samples at each iteration [20] based on a set of the previous samples and their corresponding sampling phase offsets; $Q = 2^{\upsilon}$ is the number of trellis states [21], δ is the decoding depth used to output the soft decision in SOVA [12]; k is a parameter of a low-density parity-check (LDPC) code [22]; N_{in} is the internal iterations used in the LDPC decoder; and R is a code rate.

Based on Table 1 and Table 2, we can summarize the complexity of each iterative timing recovery schemes as given in Table 3, where we employ $N_{sinc} = 21$, $\upsilon = 2$, $\delta = 5(\upsilon + 1)$ [20], and $N_{in} = 5$, and N is the number of turbo iterations. It should be pointed out that multiplication has much more complexity than addition in terms of circuit implementation. Thus, we consider only the number of multiplications when comparing the performance of different iterative timing recovery schemes. Because the modified PSP-SOVA has few extra computation steps beyond the PSP-SOVA, we can then assume that both schemes have same complexity.

5. Numerical Result

The modified per-survivor iterative timing recovery (MPS-ITR) scheme exchanges information between the modified PSP-SOVA and the SISO decoder as illustrated in Fig. 1.

Consider a rate-8/9 system in which a block of 3640 message bits is encoded by a regular (3, 27) low-density parity-check (LDPC) code [22], resulting in a coded block length of 4095 bits. The parity-check matrix has 3 ones in each column and 27 ones in each row. The SISO equalizer is implemented based on SOVA, whereas the SISO decoder is implemented based on the message-passing algorithm with 5 internal iterations. During acquisition mode, the PLL gain parameter, α, for the conventional receiver, PS-ITR, Full PS-ITR, MPS-ITR, and the trained PLL receiver are designed to recover the phase change within 256, 256, 256, 128, and 256 symbols, respectively, based on a linearized model of PLL [5], assuming that the S-curve slope is one at the origin, and there is no noise in the system. However, we consider the case where the α designed to recover the phase change within 256 symbols is used for all schemes during tracking mode.

Figure 4(a) compares the BER performance of different iterative timing recovery schemes at the 5-th iteration for the system with a moderate random walk parameter $\sigma_w/T = 0.6\%$, which implies a low probability of occurrence of cycle slips, as a function of per-bit SNRs, E_b/N_0's. Note that the number inside the parenthesis in Fig. 4 indicates the total number of iterations used to generate each curve. The curve labeled as "Perfect timing" represents the conventional receiver that uses $\tau_k = \hat{\tau}_k$ to sample $y(t)$. Also, the curve labeled as "Trained PLL" is the conventional

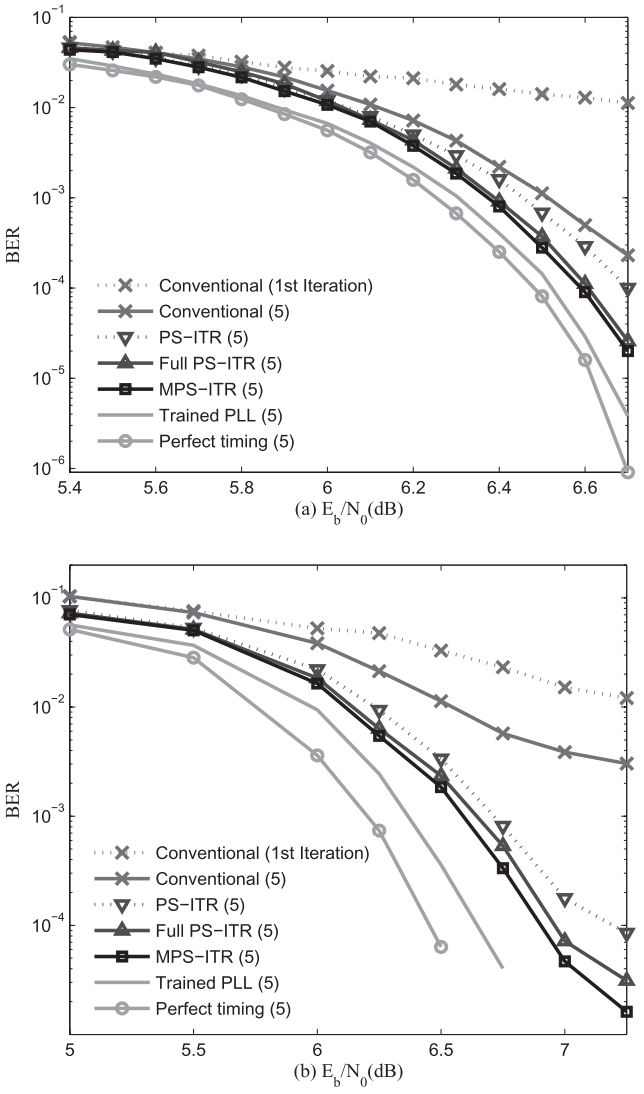

Fig. 4 Performance comparison of different iterative timing recovery scheme for (a) $\sigma_w/T = 0.6\%$ and (b) $\sigma_w/T = 1.2\%$.

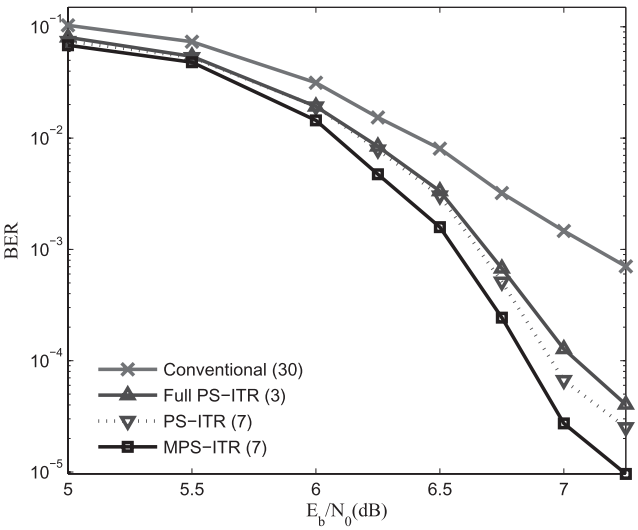

Fig. 5 Performance comparison with same complexity at $\sigma_w/T = 1.2\%$.

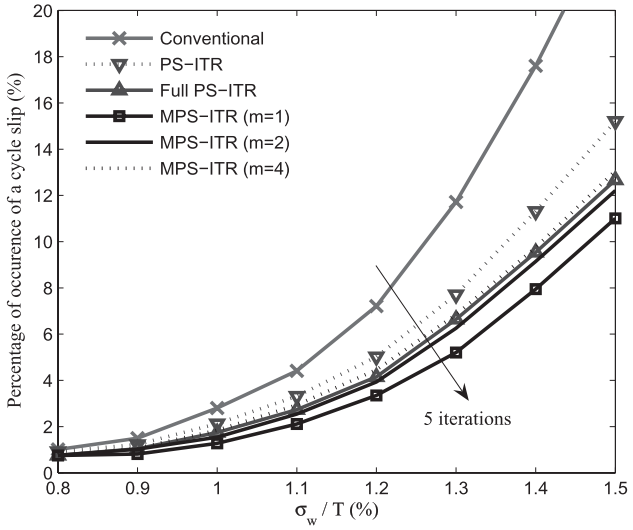

Fig. 6 Percentage of occurrence of a cycle slip at $E_b/N_0 = 5$ dB.

timing recovery whose PLL has access to all correct decisions, thus serving as a lower bound for all timing recovery schemes based on a PLL.

Apparently, the MPS-ITR performs better than the conventional receiver, and yields about 0.15 dB gain at BER = 10^{-4} over the PS-ITR. However, the MPS-ITR is slightly better than the Full PS-ITR. In addition, the MPS-ITR performs close to the trained PLL receiver and is only a 0.1 dB away from the system with perfect timing at BER = 10^{-4}. We also compare the performance of different schemes in Fig. 4(b) at a severe random walk parameter $\sigma_w/T = 1.2\%$, which implies a high probability of occurrence of a cycle slip. It is obvious that the MPS-ITR provides a large performance gain over both the PS-ITR, the Full PS-ITR, and outperforms the conventional receiver. Again, the MPS-ITR is only 0.25 dB and 0.45 dB away from the trained PLL receiver and the system with perfect timing at BER = 10^{-4}, respectively.

It is worth comparing their performance when they all have same complexity. Therefore, we assume that the current technology can support the total number of multiplications equal to 3 iterations of the Full PS-ITR scheme, which is approximately equal to 7, and 30 iterations of the MPS-ITR, and the conventional receiver, respectively (see Table 3). Figure 5 compares the BER performance of different iterative timing recovery schemes when they have same complexity at $\sigma_w/T = 1.2\%$. It is apparent that the MPS-ITR performs better than other schemes and PS-ITR is better than the Full PS-ITR, especially at high SNR. Consequently, it is worth employing the MPS-ITR in the system when the complexity is limited to a low-to-moderate amount.

One reason that MPS-ITR performs better than the PS-ITR might be because the split preamble embedded uniformly within the data stream helps reduce the occurrence of cycle slips. This can be verified by plotting the percent-

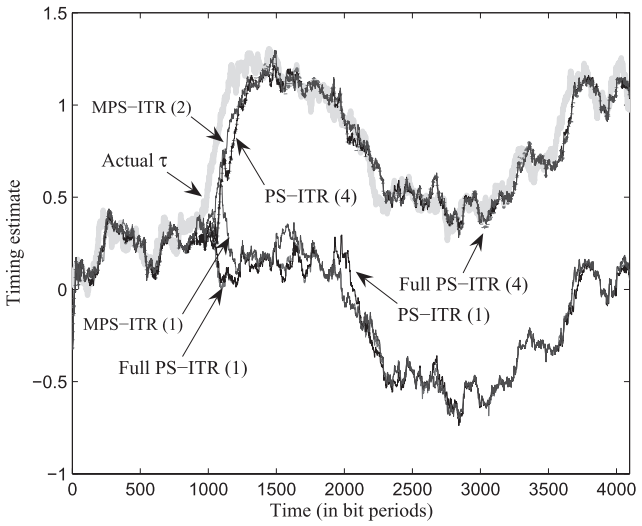

Fig. 7 Cycle slip correction at $E_b/N_0 = 6$ dB and $\sigma_w/T = 1.2\%$.

age of occurrence of a cycle slip at $E_b/N_0 = 5$ dB as shown in Fig. 6. Clearly, the MPS-ITR experiences a fewer number of cycle slips than the PS-ITR, the Full PS-ITR, and the conventional receiver, especially when the timing jitter is severe. Furthermore, the one-bit split-preamble arrangement (i.e., $m = 1$) provides the best performance.

Figure 7 shows the timing waveforms for a sample packet at $E_b/N_0 = 6$ dB and $\sigma_w/T = 1.2\%$, where the gray curve represents the actual sequence. It is clear that at the first iteration, the PS-ITR, the Full PS-ITR, and the MPS-ITR cannot keep track of the rapid change in the actual τ sequence after about 1000 symbols. Nonetheless, the MPS-ITR can correct a cycle slip within two iterations, while both the PS-ITR and the Full PS-ITR require four iterations to correct it. This implies that the MPS-ITR corrects a cycle slip faster than both the PS-ITR and the Full PS-ITR. In other words, the MPS-ITR requires fewer turbo iterations than other scheme to yield good performance.

6. Conclusion

In this paper, we propose the MPS-ITR scheme, which iteratively exchanges soft information between the modified PSP-SOVA and the decoder. This modified PSP-SOVA uses the new split-preamble strategy, which divides a preamble into two parts, each of which is used to adjust the branch metric calculation in PSP-SOVA so as to guarantee that the survivor path occurs in a correct direction.

Simulation results have shown the MPS-ITR performs better than the PS-ITR, the Full PS-ITR, and the conventional receiver, especially when the timing jitter is severe. Then, we also show the MPS-ITR is lower complexity than the Full PS-ITR. In addition, we observed that the MPS-ITR can reduce the occurrence of cycle slips and can also automatically correct a cycle slip much more efficiently than both the PS-ITR and the Full PS-ITR. In other words, the MPS-ITR requires fewer number of turbo iterations than other scheme to correct a cycle slip.

Acknowledgement

This work was supported by National Electronics and Computer Technology Centre (NECTEC) and I/U CRC in Data Storage Technology and Applications (D*STAR), KMITL under grant HDDA50-001D.

References

[1] P. Kovintavewat, J.R. Barry, M.F. Erden, and E.M. Kurtas, "Per-survivor iterative timing recovery for coded partial response channels," Proc. IEEE Global Telecommunications Conference (GLOBECOM'04), vol.4, pp.2604–2608, Texas, USA, Nov.-Dec. 2004.

[2] P. Kovintavewat, J.R. Barry, M.F. Eeden, and E. Kurtas, "Reduced-complexity per-survivor iterative timing recovery for coded partial response channels," IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2005), vol.3, pp.iii/841–iii/844, Philadelphia, USA, March 2005.

[3] K.H. Mueller and M. Müller, "Timing recovery in digital synchronous data receivers," IEEE Trans. Commun., vol.24, no.5, pp.516–531, May 1976.

[4] R. Venkataramani and M.F. Erden, "MAP-based timing recovery for magnetic recording," IEEE International Conference on Communications (ICC'08), pp.1982–1985, May 2008.

[5] J.W.M. Bergmans, Digital baseband transmission and recording, Kluwer Academic Publishers, Boston, Massachusetts, 1996.

[6] E. Hwang, R. Negi, and B.V.K. Kumar, "Extended Kalman filter based acquisition timing recovery for magnetic recording read channels," IEEE International Conference on Communications (ICC'08), pp.1986–1990, May 2008.

[7] P. Kovintavewat, J.R. Barry, M.F. Erden, and E.M. Kurtas, "Per-survivor iterative timing recovery for uncoded partial response channels," Proc. ICC'04, vol.27, no.1, pp.2715–2719, Paris, June 2004.

[8] Z. Wei, M.F. Erden, A. Kavcic, E.M. Kurtas, and R.C. Venkataramani, "Trellis-based optimal baud-rate timing recovery loops for magnetic recording systems," IEEE Trans. Magn., vol.43, no.7, pp.3324–3332, 2007.

[9] T. Souvignier, A. Friedmann, M. Oberg, P. Siegel, R. Swanson, and J. Wolf, "Turbo decoding for PR4: Parallel vs. serial concatenation," Proc. ICC'99, vol.3, pp.1638–1642, 1999.

[10] R. Raheli, A. Polydoros, and C.K. Tzou, "The principle of per-survivor processing: A general approach to approximate and adaptive MLSE," Proc. Globecom'91, vol.2, pp.1170–1175, Dec. 1991.

[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, pp.284–287, March 1974.

[12] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," Proc. Globecom'89, pp.1680–1686, Nov. 1989.

[13] V. Annampedu and P.M. Aziz, "Adaptive algorithms for asynchronous detection of coded servo signals based on interpolation," IEEE Trans. Magn., vol.41, no.10, pp.2890–2892, Oct. 2005.

[14] S. Adireddy, L. Tong, and H. Viswanathan, "Optimal placement of known symbols for frequency-selective flat-fading channels," IEEE Trans. Inf. Theory, vol.48, no.8, pp.2338–2353, Aug. 2002.

[15] R. Negi and J. Cioffi, "Pilot tone selection for channel estimation in a mobile OFDM system," IEEE Trans. Consum. Electron., vol.44, no.3, pp.1122–1128, Aug. 1998.

[16] A.R. Nayak, J.R. Barry, and S.W. McLaughlin, "Optimal placement of training symbols for frequency acquisition: A Cramer-Rao bound approach," IEEE Trans. Magn., vol.42, no.6, pp.1730–1742, June 2006.

[17] A.N. Andrea, U. Mengali, and G.M. Vitetta, "Approximate ML decoding of coded PSK with no explicit carrier phase reference," IEEE Trans. Commun., vol.42, no.2/3/4, pp.1033–1039, Feb./March/April 1994.
[18] C. Warisarn, P. Kovintavewat, and P. Supnithi, "Bi-directional timing recovery for perpendicular magnetic recording channels," Proc. DST-CON 2009, Bangkok, May 2009.
[19] 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, 1992.
[20] P. Kovintavewat and J.R. Barry, "Iterative timing recovery: A per-survivor approach," VDM Verlag Publisher, Germany, Nov. 2009.
[21] G.D. Forney, "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," IEEE Trans. Inf. Theory, vol.IT-18, no.3, pp.363–378, May 1972.
[22] R. Gallager, "Low-density parity-check codes," IRE Trans. Inf. Theory, vol.IT-8, pp.21–28, Jan. 1962.

Chanon Warisarn received the B.Eng. (Hon.) in Electronics Engineering Technology from King Mongkut's Institute of Technology North Bangkok in 2006. His research interests are in the areas of communications and signal processing for data storage systems. He is currently a doctoral student at King Mongkut's Institute of Technology Lardkrabang, Bangkok, Thailand.

Piya Kovintavewat received the B.Eng. summa cum laude from Thammasat University, Thailand (1994), the M.S. degree from Chalmers University of Technology, Sweden (1998), and the Ph.D. degree from Georgia Institute of Technology (2004), all in Electrical Engineering. He currently works at Nakhon Pathom Rajabhat University. His research interests include coding and signal processing as applied to digital data storage systems. Prior to working at NPRU, he worked as a research assistant at National Electronics and Computer Technology Center (1999), both in Thailand. He also had work experiences with Seagate Technology, Pennsylvania, USA (summers 2001, 2002, and 2004).

Pornchai Supnithi received the B.S. degree in Electrical Engineering from University of Rochester, Rochester, New York, USA, in 1995. M.S. degree in Electrical Engineering from University of Southern California, Los Angeles, California, USA in 1997 and Ph.D. in Electrical Engineering from Georgia Institute of Technology, Atlanta, Georgia, USA in 2002. He is with the Telecommunications Engineering Department, KMITL, Bangkok, Thailand. His research interests are in the area of channel modeling, error correction control and equalization and iterative processing applied to recording channels.