Symbol Timing Recovery Using Oversampling Techniques

Hong-Kui Yang[1] and Martin Snelgrove
Dept. of Electronics, Carleton University
Ottawa, ON K1S 5B6, Canada
'Also with Nortel Wireless Networks, Ottawa, Canada
Email: hkyang@nortel.ca snelgar@doe.carleton.ca

ABSTRACT: It is advantageous to use oversampling techniques with either ΔΣ or broadband data converters in both wireline and wireless digital receivers. This paper discusses the oversampling techniques for all-digital implementation of symbol timing recovery in digital receivers. The idea of oversampling techniques for timing recovery is to adjust the timing phases while decimating the oversampled signals. The spurious signal introduced by adjusting the CIC's (cascaded integrator-comb) timing phase has been analyzed and was found to be a serious problem. In this paper, a dual-differentiator timing phase adjustable decimation filter has been proposed and was used for symbol timing recovery. Simulations were provided to verify the validity of the proposed method.

I. INTRODUCTION

Symbol timing recovery is critical for reliable data detection in modern digital communications [l]. There are a number of ways to recover the symbol timing. In general, they can be categorized as [l]-[3]: pure analog recovery, mixed (analog-digital) recovery, and all-digital recovery. The first two methods require VCOs to create synchronized timing clocks. To take advantage of digital techniques, it is desirable to implement timing recovery circuit all-digitally.

In a digital system, there is often a fixed system clock, and asynchronous digital inputs create difficulties. Interpolation for symbol timing recovery shown in Fig. l(a) was proposed in [2,3] and is suitable for all-digital implementation where Nyquist rate sampled input signals are available. Due to the widespread use of oversampling in digital communications (see Section II), the interpolation method for timing recovery shown Fig. l(a) is not optimal for this case. This is made clearer in Fig. l(b), where the analog input is oversampled. The basic principle of interpolation for timing recovery reveals that the interpolation seems to be redundant in Fig. l(b). An alternative way is shown in Fig. l(c), where the timing recovery is done in the decimation. In this paper, we will discuss the novel symbol timing recovery where we incorporate timing phase adjustment in the decimation and therefore save lots of computation.

II. OVERSAMPLING IN DIGITAL RECEIVERS

Block diagrams for digital receivers using oversampling techniques are shown in Fig. 2. Note that the structures are suitable for many applications.

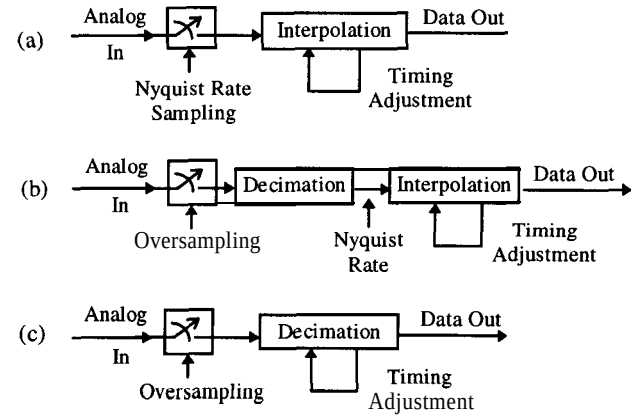

Fig.1 All-digital symbol timing recovery: (a) interpolation for Nyquist sampling; (b) interpolation for oversampling, and (c) decimation for oversampling.

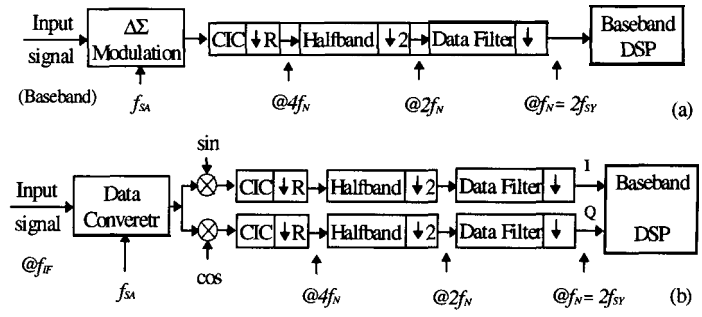

Fig.2 Block diagrams of digital receivers: (a) digital baseband receiver; (b) digital quadrature receiver.

1) ISDN: This is a baseband transmission and no carrier is needed. Hence, a lowpass ΔΣ data converter is used, as shown in Fig. 2(a). Decimation filters follow the data converter to downconvert the signal rate, as discussed below.

2) *Voiceband data transmission:* QAM is popular for high speed data transmission. The center frequency of the incoming IF signal is comparable with the signal band. Therefore, a lowpass ΔΣ modulation data converter can be used, shown in Fig. 2(b). The signals, *Sin* and Cos, are used to mix the modulated signals to I and Q signals.

3) *Digital quadrature radio receiver with bandpass ΔΣ modulation:* Fig. 2(b) is suitable for both basestation and handset digital cellular, where a bandpass ΔΣ data converter is used. Here, the sampling frequency, f_{SA}, is typically chosen to be four times the center frequency of the incoming

IF signal, f_{IF}; namely, $f_{SA} = 4f_{IF}$. Therefore, the *Sin* and *Cos* signals for the mixing become simple sequences {1, 0, -1, 0, ...} and {0, -1, 0, 1, ...}. After mixing, oversampled baseband I and Q signals are obtained.

4) Digital quadrature radio receiver with broadband data converter: The front-end can be a broadband data converter to digitize an IF signal in a digital quadrature radio receiver in Fig. 2(b). The design difference between this receiver and that in 3) lies in different considerations in the following decimation filters.

The digital receiver structures shown in Fig. 2(a) and (b) are two general frames which fit all the oversampling digital receivers considered here. The common point among them is that we have oversampled baseband signals after mixing (no mixing for ISDN transmission). Therefore decimation filters are necessary to remove the out-of-band noise and simultaneously downconvert the oversampling rate to appropriate rate. A very efficient CIC decimation filter followed by two half-band decimation filters can complete this job [7],[9-11]. Since ΔΣ data converters are very promising in achieving the stringent requirements in digital receivers, we concentrate on ΔΣ oversampling technique (although it is not necessary) in the following discussion. The principle is easy to extend to broadband data conversion.

The decimation is split into three stages [9]. The CIC filter first downconverts the oversampling rate, f_{SA}, to four times the final rate, f_N, which is twice the data symbol rate, f_{SY} [7-9]. The reason for "four times" is to keep the droop in the edge of the signal band low enough to ease frequency compensation in the baseband DSP. The CIC decimation filter is optimal if its order is one more than the order of the preceding lowpass ΔΣ modulation or one more than half the order of the preceding bandpass ΔΣ modulation. Following the CIC filter is a halfband decimation filter which further downconverts the sampling rate to $2f_N$. Then two data filters are used to shape the received signal pulses to meet the Nyquist criterion and also downconvert the sampling rate to f_N. Included in the baseband DSP are symbol timing recovery (if we use the interpolation method), carrier recovery (not for ISDN transmission) and channel equalization.

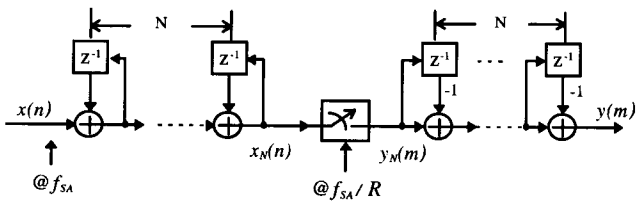

Fig. 3 An *N*th-order CIC decimation filter with decimation factor R

An *N*th-order CIC filter [10] is shown in Fig. 3. The filter consists of N digital integrators operating at a high sampling rate, f_{SA}, and N differentiators at a low sampling rate, f_{SA}/R, where R is an integer sampling downconversion factor. Its transfer function is $H(z) = \left((1-z^{-R})/(1-z^{-1})\right)^N$.

There are many advantages to a CIC decimation filter, such as: no multipliers, no need for storage elements, wide range of rate change, etc.

III. ADJUSTABLE TIMING PHASE CIC FILTERS FOR TIMING RECOVERY

Now that oversampled baseband signals are available with the digital receiver in Fig. 2, the question becomes how to adjust the timing through decimation.

A. Adjustable Timing Phase CIC Decimation Filters

One straightforward way to adjust the timing phase in Fig. 2 is to vary the CIC filter's downconversion factor, R, in one symbol interval to minimize the timing error. It can be seen from Fig. 2 that the relation between the symbol rate, f_{SY}, and the oversampling rate, f_{SA}, is $f_{SY} = f_{SA}/(8R)$ or $T_{SA} = T_{SY}/(8R)$, where T_{SA} and T_{SY} stand for the oversampling and symbol intervals, respectively. As can be seen from Fig. 3 that we can advance or retard the timing phase by reducing or increasing R. The minimum timing increment is T_{SA}. R is now time-varying, with an average value that tracks drift between local and transmitted clocks. The definition of T_{SA} is different for a lowpass and bandpass ΔΣ, although the definition of the *OSR* is similar (defined as the ratio of the oversampling rate to twice the band of interest).

$$T_{SA} = \begin{cases} T_{SY}/(2 \cdot OSR), & \text{for a lowpass } \Delta\Sigma \text{ data converter} \\ T_{SY}/(4 \cdot OSR), & \text{for a bandpass } \Delta\Sigma \text{ data converter} \end{cases} \quad (1)$$

Note that the minimum phase adjustment is better than 1 % of the symbol interval with the *OSR* over 64 for these two cases.

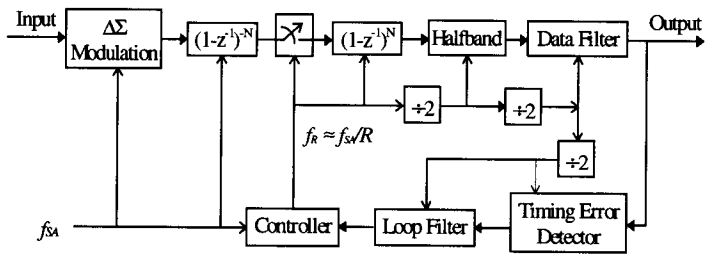

Fig. 4 Block diagram of oversampling timing recovery with a timing phase adjustable CIC decimation filter

B. Timing Recovery Loop

Fig. 4 shows a block diagram for timing recovery which incorporates the above idea. Note that only one path is shown for the sake of simplicity and the idea can easily be extended to the quadrature case in Fig. 2. Also shown in the figure are the sampling rate relationships among all the blocks, which are in agreement with those of Fig. 2. The timing error detector in Fig. 4 uses a two-sample-per-symbol algorithm to detect the timing error, such as in [4]. Other symbol-rate

algorithms [1] can also be used, depending on the applications. The loop filter works at the symbol rate and outputs a smoothed timing error for each symbol interval. The controller, which is clocked at the oversampling rate, f adjusts the timing phase by changing the downconversion factor, R, in a counter. There are basically two ways to adjust the timing phase. In one-oversampling-interval adjustment, we need to advance or retard one oversampling interval, T_{SA}, one time. This can be realized by reducing or increasing R by a factor of 1. In multi-oversampling-interval adjustment, the controller calculates exact how many oversampling intervals are needed to advance/retard in order to minimize the timing error. we can implement this by changing R by more than 1.

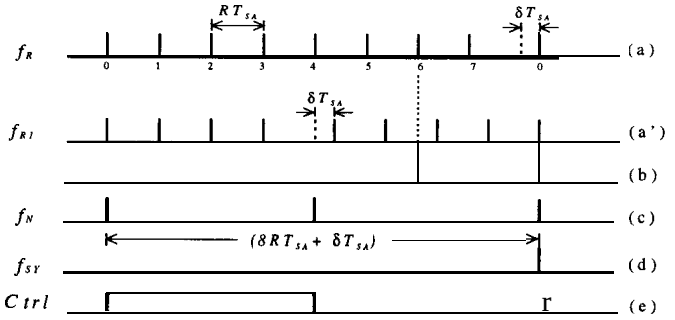

Fig. 5 Timing diagram for the timing phase adjustment

Since f_R = 8 f_{SY} (that is, one symbol interval corresponds to eight controller's output clock intervals), one simple way to adjust the timing phase is shown in Fig.5, where we adjust the timing phase of the sample, for example, after sample #7 during one symbol interval (see (a); (a') is not used here and will be described in the following section). The reason is that new symbol and timing error data are available after the first sample #0. The timing phase is adjusted by &.T_{SA} in one symbol interval, where δ is an integer (> 0 for advancing, < 0 for retarding, = 0 for retaining).

C. *Delays in the Timing Recovery Loop*

The timing recovery loop in Fig. 4 uses feedback and extra delays besides those introduced by the timing error detector, loop filter and controller should be discussed. Loop delay has an adverse effect on the stability of the feedback loop. The extra delays are introduced by N differentiators of the CIC decimation filter, the halfband decimation filter and data filter. The transfer function of the N cascaded differentiators is $(1-z^{-1})^N$. Hence, they introduce a delay of NT_R I 2, where T_R = 1/ f_R. We assume FIR halfband and data filters have N_h and N_d taps, respectively. Thus, the delays introduced by these two filters are $(N_h - 1)T_R/2$ and $(N_h - 1)T_R$, respectively. Using $T_R = \frac{1}{8}T_{SY}$, *we* get the total extra loop delay as follows:

$$D_{extra} \cong (N_h + 2N_d)T_{SY}/16 \qquad (2)$$

In practice, the first term in (2) introduces about 1 symbol interval delay, and the second has a delay of 2~ 3 symbols. The total extra delay would be around 3 ~ 4 symbol intervals.

VI. A PRACTICAL OVERSAMPLING TECHNIQUE FOR TIMING RECOVERY

A. Problems with the Timing Adjustable CIC Filter

After the timing phase adjustment, the CIC filter's output needs a while to settle down due to the delays in its N differentiators. Before settling down, there are a couple of "spikes" in the output of the CIC filter, possibly causing misadjustment. To see the problem, let us go back to Fig. 3. The output of the CIC filter at time instant t_{Rm}, $y(t_{Rm})$, can be written as,

$$y(t_{Rm}) = \sum_{k=0}^{N} c_k\, y_N(t_{R(m-k)}) \qquad (3)$$

where index m means that the sample is the mth at time instant t_{Rm}, c_k's *are* the coefficients for the combined transfer function of the N-cascaded differentiators and $y_N(t_{R(m-k)})$ is the input of the differentiators at time instant $t_{R(m-k)}$. For a uniformly sampled signal with the period of T_R, *we* have $t_{Rm} = mT_R$, where m is an integer.

Due to decimation by a factor of R, the input of the differentiators, $y_N(t_{Rm})$, is a decimated version of the output of the N cascaded integrators, $x_N(t_{Sn})$, where index n stands for the nth sample. Therefore, we have, at time instant, t_{Rm},

$$y_N(t_{Rm}) = x_N(R \cdot t_{Sm}) \qquad (4)$$

which will be made clearer by using $t_{Sn} = nT_{SA}$ and $T_R = R \cdot T_{SA}$ for uniform sampling, but R is timing-varying.

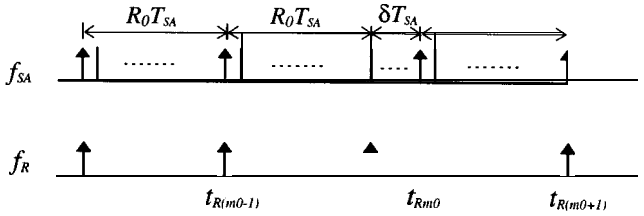

Fig.6 Timing phase adjustment starting after $t_{R(m0-1)}$

Let us consider a case where we start to adjust timing phase at the sample time after $t_{R(m0-1)}$ by changing R from R_0 to $(R_0 + \delta)$ and back to R_0 afterwards, as shown in Fig. 6, where R_0 is the nominal value of R. We have

$$t_{R(m0+l)} = \begin{cases} R_0(m_0 + l)T_{SA}, & l < 0 \\ R_0(m_0 + l)T_{SA} + \delta T_{SA}, & l \geq 0 \end{cases} \qquad (5)$$

where *1* is an integer. Substituting (5) and (4) into (3), we get

$$y(t_{R(m0+l)}) = \begin{cases} \sum_{k=0}^{N} c_k x_N\left(R_0(m_0+l-k)T_{SA}\right), & l<0 \\ \sum_{k=0}^{l} c_k x_N\left((R_0(m_0+l-k)+\delta)T_{SA}\right) & \\ \quad + \sum_{k=l+1}^{N} c_k x_N\left(R_0(m_0+l-k)T_{SA}\right), & 0\le l<N \\ \sum_{k=0}^{N} c_k x_N\left((R_0(m_0+l-k)+\delta)T_{SA}\right), & l\ge N \end{cases} \quad (6)$$

It is noted that the CIC filter's outputs, $y(t_{R(m0+l)})$ for $0 \le 1 < N$, calculated from (6) are not the actual values we expected since there exist nonuniform samples in (6). The nonuniform samples are due to having a time shift of δT_{SA} in the first term compared to second term in (6) for $0 \le l < N$. *The* correct values for $0 \le 1 < N$, $y'(t_{R(m0+l)})$, should be

$$y'(t_{R(m0+l)}) = \sum_{k=0}^{N} c_k x_N\left((R_0(m_0+l-k)+\delta)T_{SA}\right), 0 \le l < N \quad (7)$$

The nonuniform sampling (or jitter) due to the timing phase adjustment results in errors, $\Delta y(m_0+l) = y(t_{R(m0+l)}) - y'(t_{R(m0+l)})$, **that is**

$$\Delta y(m_0+l) = -\sum_{k=l+1}^{N} c_k \Delta x_N\left(R_0(m_0+l-k)\right), \quad 0 \le l < N \quad (8)$$

where

$$\Delta x_N(R_0 m) = x_N(R_0 m T_{SA}) - x_N(R_0 m T_{SA} + \delta T_{SA}) \quad (9)$$

Since $x_N(n)$ is the output of the N cascaded integrators (accumulators), it is very large and varies a lot from sample to sample. In the design of the timing adjustable CIC decimation filter, a natural ovefflow method is used and the word length is limited by $B_{\max} = \lceil N \log_2 R \rceil + B_{in}$, where $(B_{in}+1)$ is the input data's word length in 2's complement representation, and $\lceil x \rceil$ is the smallest integer not less than x. *The* errors in (8) introduced by nonuniform sampling are very large and span N samples starting from the beginning of timing phase adjustment. The direct results from simulations shown that there are N "spikes" (namely, spurious samples). These errors will affect the following both symbol data detection and timing error detection.

B. Dual-Differentiator Adjustable Timing Phase CIC Filters

Considering the fact that the CIC decimation filter needs N samples to settle down and the facts that $N \le 4$ is the most often used and $T_R = 8T_{SY}$, we propose a dual-differentiator timing phase adjustable CIC decimation filter, which is shown in Fig. 7 with the whole timing recovery loop. The timing diagram for clocks and a control signal needed for the timing recovery is depicted in Fig. 5.

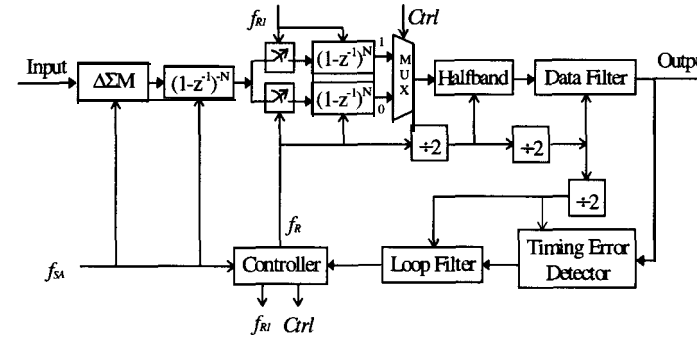

Fig. 7 A practical timing recovery loop with a dual-differentiator **CIC** decimation filter

Here we adjust the timing phase of channel 1 (Fig. 5(a')) during the first half symbol interval (that is, starting after sample #3 of the differentiators's input data) and output channel 0. After the adjusted data settles down, we start to adjust the timing phase of channel 0 in the second half interval (Fig. 5(a)) (namely, starting after sample #7 of the differentiators' input data) and output the correct data from channel 1. MUX is used to alternatively select between channels 0 and 1, controlled by a control signal, Ctrl . This scheme can accommodate up to 3rd-order $\Delta\Sigma$ or 6th-order bandpass $\Delta\Sigma$ modulation, which is usually used in practice [7]. For the CIC filter's order $N \ge 5$, *we* have to increase T_R to $T_R = 16T_{SY}$.

The controller provides the sampling rates f_R and f_{R1} required by channels 0 and 1, respectively, with different phases, as shown in Fig. 5. The averages of these two sampling rates are the same during one symbol interval. All the other sampling rates, $2f_N$, f_N, and f_{SY}, *are* derived from f_R, shown in the figure. The phase jitter introduced by adjusting the timing phase in the second half symbol interval has a negligible effect on the following halfband filter and data filter due to the small difference between two samples separated by a phase shift of δT_{SA} for baseband signals.

C. *Simulation Results*

Simulations have been done to verify the validity of the proposed method. Using SPW simulator [12], we set up a QPSK system for simulations, where an bandpass $\Delta\Sigma$ modulation is used. The timing error detection algorithm is taken from [4], which can be written as,

$$e(k) = y_I(k-\tfrac{1}{2})[y_I(k) - y_I(k-1)] + y_Q(k-\tfrac{1}{2})[y_Q(k) - y_Q(k-1)] \quad (10)$$

where e(k) is the timing error at instant k, and y,, y_Q are the data outputs of I, Q channels, respectively. It is noted that the proposed method is independent on the timing error detection algorithm. This algorithm is good for detecting data signals with alternating O's and 1's. If a proper loop filter is

put after this algorithm, it also works well with random data signals. Two cases are considered here: one is for training sequences with alternating O's and l's, and another is for random data signals. The sampling phase shift error and frequency drift error are simulated for each case. The phase shift is around a quarter of a symbol period and the local clock frequency is 0.1 % fast relative to the transmitter sampling rate. The timing errors shown in Fig. 8 are taken at the output of a first-order lead-lag loop filter, where the errors are normalized to the symbol amplitude and T is symbol interval. Fig. 9 shows the constellation scatter plots for the output signals. In the simulations, we did not compensate for around 3 dB droop introduced by the CIC filter. Therefore, the points in the scatter plots seem to be slightly large.

V. CONCLUSIONS

Interpolation is a way to implement symbol timing recovery all-digitally. However, many applications in digital communications require oversampling techniques for the data conversion. By incorporating the timing phase adjustment in the CIC decimation for the oversampling signals, we have shown a very efficient way to recover the symbol timing while performing decimation. In addition to having the advantages inherent in all-digital implementation, the oversampling techniques for symbol timing recovery have other advantages over the interpolation method:

1) When the symbol data is recovered, we also recover the timing clock. This is important in some application such as the ISDN U-interface **[1** 1], where the recovered timing clock will be used in the network termination to synchronize the transmitting data to the master clock in the line termination. Extra circuits will be needed for this job in the interpolation method [3].

2) It takes advantages of oversampling techniques ($\Delta\Sigma$ modulation and broadband sampling) which are becoming increasingly important in meeting the stringent requirements for digital receivers. By incorporating symbol timing recovery in the required decimation filter, we can have simple implementation.

REFERENCE

[1] E.A. Lee and D.G. Messerschmitt, *Digital Communication,* Boston, N.Y.: Kluwer Academic Publishers, 2nd printing, 1990

[2] A. Haoui, H.-H. Lu and D. Hedberg, "An all-digital timing recovery scheme for voiceband data modem," *Proc. IEEE Conf. Acoust., Speech, Signal Processing,* pp. 1911-1914, 1987

[3] F.M. Gardner, "Interpolation in digital modem — Part I: Fundamentals," *IEEE Trans. Commun.*, vol. 41, pp. 502-508, Mar. 1993

[4] F.M. Gardner, "A BPSK/QPSK timing-error detector for sampled receivers," *IEEE Trans. Commun.*, vol. 41, pp. 423-429, May 1986

[5] D.G. Messerschmitt, "Echo cancellation in speech and data transmission, " *IEEE J. Selected Areas in Commun., vol. 2,* no. 2, pp. 283-297, Mar. 1984

[6] J. Mitola, Guest Editor, Special Issue on Software Radios, *IEEE Commun.* Mag., May 1995

[7] J.C. Candy and G.C. Temes, *Oversampling* Delta-Sigma Data *Converters: Theory, Design and Simulation.* IEEE Press, New York, 1992

[8] R. Schreier and M. Snelgrove, "Bandpass sigma-delta modulation," *Electron. Lett.,* vol. 25, pp. 1560-1561, Nov. 1989

[9] J.C. Candy, "Decimation for sigma delta modulation," *IEEE Trans. Commun.,* vol. 34, pp. 72-76, Jan. 1986

[10] E.B. Hogenauer, "An economical class of digital filters for decimation and interpolation," *IEEE Trans. Acoust,, Speech, Signal Processing,* vol. ASSP-29, pp. 155-162, Apr. 1981

[11] D.G. Messerschmitt, "Design issues in the ISDN U-interface transceiver, " *IEEE J. Selected Areas in Commun.,* vol. 4, no. 8, pp. 1281-1293, Nov. 1986

[12] SPW Manuals, Alta group of Candence System, 1992

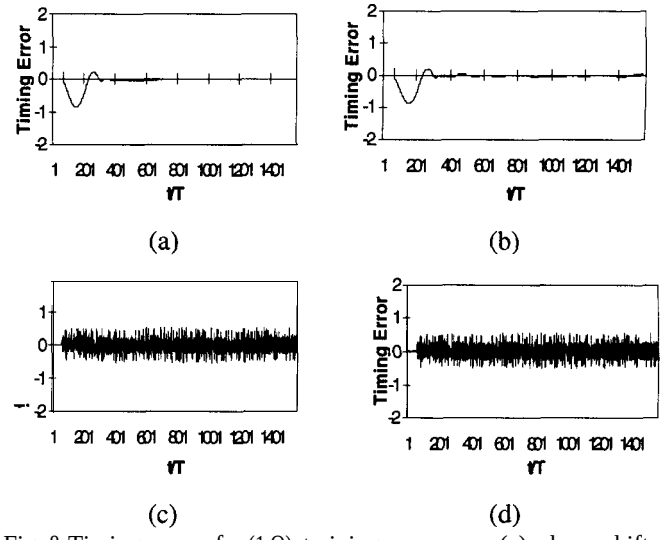

Fig. 8 Timing errors for (1,O) training sequence: (a) phase shift, (b) frequency drift. For random data: (c) phase shift, (d) frequency drift

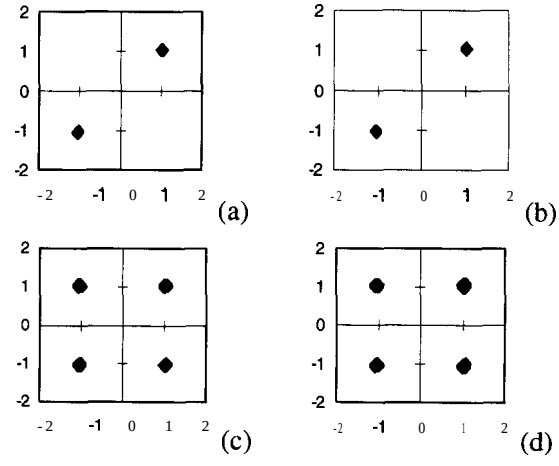

Fig. 9 Scatter plots for training sequences: (a) phase shift, (b) frequency drift. For random data: (c) phase shift, (d) frequency drift