PAPER

An All-Digital Clock Recovery and Data Retiming Circuitry for High Speed NRZ Data Communications

Muhammad E.S. ELRABAA[†a)], *Nonmember*

SUMMARY This paper describes a new circuit technique for performing clock recovery and data re-timing functions for high-speed source synchronous data communications, such as in burst-mode data transmission. The new clock recovery circuit is fully digital, non-**PLL**-based, and is capable of retiming the output clock with the received data within one data transition. The absence of analog filters or other analog blocks makes its area much smaller than conventional circuitry. It can also be described by any hardware description language, simulated, and synthesized into any digital process. This enables it to be ported from one technology to another and support system on a chip (**SOC**) designs. The design concept is demonstrated with **T-Spice**® simulations using a 0.25 μm digital **CMOS** technology. Static performance was evaluated in terms of supply and temperature dependent skews. The shifts in output clock due to these static conditions were within ±40 pS. Also dynamic behaviours such as jitter generation and jitter transfer were evaluated. The circuit generates a jitter of 68 pS in response to a supply noise of ±250 mV amplitude and 100 MHz frequency. Input data jitter transfer is within ±0.1 dB up to a jitter frequency of 150 MHz.

key words: *clock recovery, low-power digital CMOS circuits*

1. Introduction

The fast expansion of multimedia (Audio/Video) over the Internet in recent years has caused an explosion in the data transfer volume over wide-area networks (**WANs**) and local-area networks (**LANs**). This led to a rapid migration from copper wires to optical fibers as transmission media for high-speed digital-transmission schemes such as synchronous optical networks/synchronous digital hierarchy (SONET/SDH). This necessitated the development of low-cost high-speed clock recovery circuits (**CRC**s) in the repeaters and receivers that can accurately extract clock signals from nonreturn-to-zero (**NRZ**) source-synchronous serial bit streams. The **CRC** must maintain synchronism between the generated clock and the data in the presence of data phase noise (jitter), supply and temperature fluctuations. Also, in a point-multipoint operating scenario (as in optical networks), the **CRC** must be agile in extracting synchronized clocks for different data packets arriving from different. In the next section a comprehensive review of existing clock recovery techniques for **NRZ** data, their advantages, and shortcomings is presented. Also, the targeted characteristics of the new **CRC** are stated. The circuit description of the proposed **CRC** is given in Sect. 3. The performance evaluations using **T-Spice**® simulations are presented in Sect. 4. These include skew due to supply and temperature variations as well as jitter characteristics of the output clock. Conclusions are in Sect. 5 followed by the list of references.

2. Review of Clock Recovery Techniques

Random **NRZ** data has two properties that make clock recovery difficult [1], [2]; 1) No spectral contents at the bit rate or its even-order harmonics and 2) Long streams of consecutive 1 s and 0 s. The first property can be dealt with by using edge detection of the **NRZ** data to generate strong spectral contents at the bit rate and its harmonics [1]. The second property plagued traditional **CRC**s such as Phase-Locked-Loops (**PLL**s). A **PLL** that employs phase locking view long sequences of 1 s or 0 s as new low frequency data and attempts to lock to this new frequency. This produces data-dependent jitter in the **PLL**'s output. To reduce this jitter the **PLL**'s bandwidth has to be limited which in turn limits the **PLL**'s capture range (i.e. limit the **PLL**'s ability to lock-in a different data with an actual different frequency). Also, larger loop gain reduces jitter generation but reduces stability. A switched filter can tune the loop gain and bandwidth to reduce the jitter generation [3]. Alternatively two loops can be used, one for frequency acquisition (coarse tuning) and another for phase acquisition (fine tuning) [2], [4]. In summary, analog **PLL**-based **CRC**s suffer from several shortcomings; 1) The analog blocks such as the **VCO**, charge pump and loop filter are large in area and difficult to port to other processes or supplies, 2) The **VCO** has to operate at double the data frequency to ensure 50% duty limiting the data rate, 3) Phase error accumulation in the **VCO**, 4) Long lock times due to the loop damping behavior. Many recent designs proposed solutions for some of the above shortcomings. In [5] a **PLL** with a **VCO** that runs at half the input data frequency is proposed. However, an additional delay-locked-loop is required to generate 4 phases of this clock to retime the data and the other analog features are still retained. In [6] the **VCO** is operated

Manuscript received July 9, 2001.
Manuscript revised October 24, 2001.

[†] The author is with the Computer Engineering Department, King Fahad University of Petroleum and Minerals, P.O.Box 584, Dhahran 31261, Saudi Arabia.
a) E-mail: elrabaa@kfupm.edu.sa

at the clock frequency but both data and clock are divided by 2 such that the phase detector (**PD**) operates at half the clock frequency. This increased the operating frequency of the **CRC** but also increased the jitter since the jitter correction is done every two input transitions. Fully digital or semi-digital solutions have been proposed to solve some of the problems associated with analog **PLL**s. A semi-digital **CRC** that utilizes an analog **PLL** to generate a clock with 10 phases and a digital **PLL** to select the appropriate phase for data retiming is described in [7]. The relatively poor resolution, however, means increased jitter. In [8], feedback phase-selection and averaging phase-interpolation are added to increase resolution and reduce the jitter. In [9] an analog **PLL** and a gated **VCO** are used to achieve faster lock times. Still, these solutions retain many of the analog **PLL** shortcomings and stability is an issue with the two loops interaction. A fully digital **PLL** was proposed in [10]. The period of a high-speed clock is used as a reference for the digitally controlled oscillator (**DCO**) and the phase-comparator. Both resolution and frequency of operation are limited, but fast locking is achieved. A fully digital delay-locked-loop clock multiplier that utilizes delay interpolation was also developed [11]. The above-mentioned digital or semi-digital **PLL** implementations retain most of the architectural features of their analog counterparts. The use of feedback techniques requires some form of filtering to adjust the loop gain and bandwidth. This complicates the design because it has to be done while insuring that the loop remains stable. Non-PLL/DLL digital techniques for clock generation, multiplication, or de-skewing were proposed [12], [13]. They use phase interpolation [12] or synchronous delay mirrors [13] to shorten lock times and correct duty cycles. These techniques utilize complex control schemes (for cycle detection and duty cycle corrections) and require more than one clock cycle for locking. The circuit described in this paper does not require any control and is able to lock to the data's phase within one transition. The main objective of this work was to devise a fully digital **CRC** thus allowing cell-based designs that can be ported from one process to another. The following characteristics were targeted; 1) Compact size, 2) Low power consumption, 3) Unconditionally stable, 4) Low jitter generation, 5) Fast locking, and 6) High jitter tolerance.

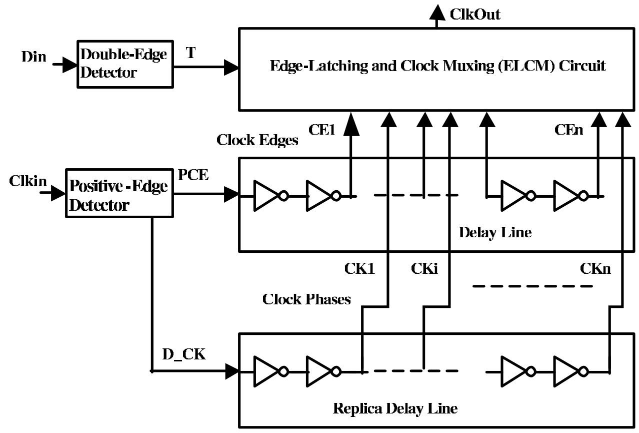


Fig. 1 The architecture of the proposed all-digital **CRC**.

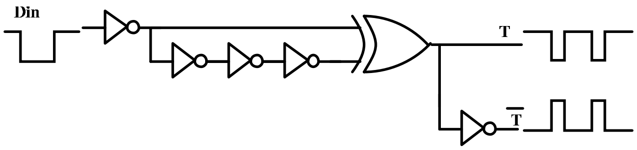


Fig. 2 The double-edge detector circuitry.

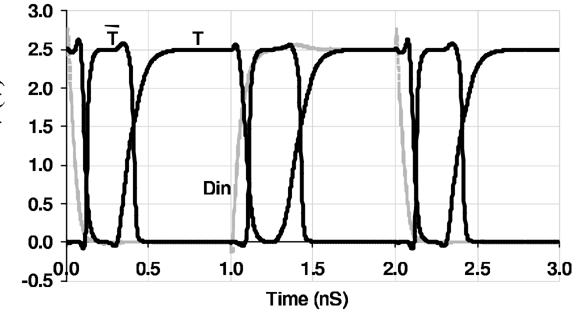


Fig. 3 Simulation of the input/outputs of the **DED** circuit.

3. Circuit Description of the Proposed CRC

The proposed fully digital, non-PLL/DLL, non-feedback **CRC** is shown in Fig. 1. It uses an external clock (no frequency acquisition is performed) to generate an output clock synchronized with the data. Clock folding and phase interpolation are used to increase the output clock's resolution. There are four main blocks; 1) Data double-edge detector, 2) Clock positive-edge detector, 3) Replicated delay lines (two), and 4) A data-edge latching and clock muxing (**ELCM**) circuit. All blocks are implemented using standard **CMOS** circuits. Circuit description of each block is given below.

3.1 The Double-Edge Detector (**DED**)

The schematic of the **DED** circuit is shown in Fig. 2. This circuit generates narrow output pulses for every transition in the **NRZ** input data. The **XOR** gate in Fig. 2 is a conventional static **CMOS** gate. The output, **T**, and its complement are used as trigger signals for the **ELMC** circuit. The output pulse has a width of thee inverters-delay. Adding more inverters in the second input path of the **XOR** gate can increase the width of the output pulse. Figure 3 shows the **T-Spice** output for the **DED** circuit at an input data rate of 1 Gb/s. The output pulses of the **DED** circuit are used to re-synchronize the output clock of the **CRC** with the input data. Hence if there are no input transitions (for example an all **1** s stream), the output clock will remain at the same frequency and phase. In other words, phase corrections to the output clock will only occur in response to an input transition. Hence At least one input transition is required to separate data from different sources.

3.2 The Positive-Edge Detector (**PED**)

This circuit, depicted in Fig. 4, is similar to the **DED** except that it generates a pulse only when the input exhibits a positive transition. It has two outputs that are generated from the input clock (**Clk**$_{in}$); a pulsed-clock edge (**PCE**) output that is used as an input to the first delay line, and a delayed clock (**D_CK**) output that drives the second delay line. Two inverters are used to delay **Clk**$_{in}$ approximately by an equal delay to that of the input inverter and **XOR** in the **DED** circuit.

Simulation result for the **PED** circuit at 1 GHz is shown in Fig. 5. As this figure shows, the **PCE** output pulse is significantly narrower than the trigger pulse generated by the **DED** circuit. As will be explained later, this is very important for the proper operation of the **CRC**. This, however, does not require any special design since the **NAND** gate in the **PED** circuit is significantly faster than the **XOR** gate in the **DED** circuit. A concern might arise about the possibility of transmitting such a wideband signal on a **CMOS** IC. These signals will have short paths since the delay line and **ELMC** can be packed very closely. This eliminates the effects of self-inductance or substrate losses and **RC** effects would dominate. Assuming the output impedance of the **PED** is 1 kΩ (a much higher value than the real effective output impedance) and its load capacitance to be 50 fF, the cutoff frequency would be about 20 GHz. Hence all the major harmonics of the **CE**$_i$**s** signals will pass.

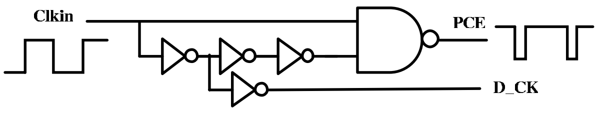


Fig. 4 Schematic of the positive-edge detector circuit.

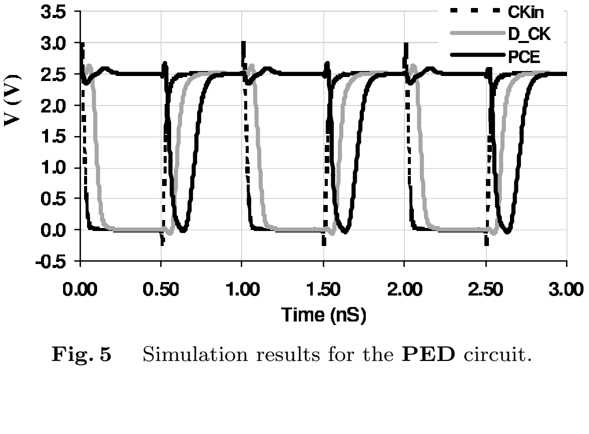


Fig. 5 Simulation results for the **PED** circuit.

3.3 The Delay Lines

The two identical delay lines, each made of $2n$ identical inverters, are used to generate n phases of the **PCE** and the **D_CK** signals (Fig. 1). The successive outputs of these delay lines (**CE**$_1$ to **CE**$_n$ and **CK**$_1$ to **CK**$_n$) are separated by two inverters' delay ($2 \times \mathbf{T}_{Dinv}$) and are fed to the **ELCM** circuit.

3.4 The **ELCM** Circuit

This circuit selects and maintains the appropriate clock phase that is synchronized with the input data. As shown in Fig. 6, the **ELCM** circuit consists of an array of n pulsed flip-flops (**PFF**) and transmission gates (**TG**s). The transmission gates form an n-to-1 clock mux. The outputs of the **PFF**s circuit control the **TG**s. The circuit diagram of the **PFF** with the **TG** is shown in Fig. 7. The **CE** pulsed signals are latched by the trigger signal **T** (and its complement) signals. The **PFF**s have an input, **I**, that, if asserted, would inhibit the latching of the **CE** input pulse. The output of the first latching stage of a **PFF** provides the **I** input to the next **PFF** and last **PFF** provides the **I** input for the first **PFF** as shown in Fig. 6. Hence no two successive **PFF**s could latch simultaneously. n is chosen according to the wanted frequency of operation.

3.5 Operation of the **CRC**

The basic operation of the **CRC** circuit is illustrated in Fig. 8. The **DED** circuit generates the trigger signal T and its complement from the **NRZ** input data. The **PED** circuit generates a train of negative pulses (**PCE**)

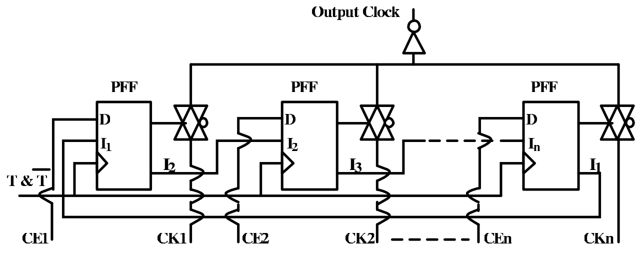


Fig. 6 The **ELCM** circuit's schematic.

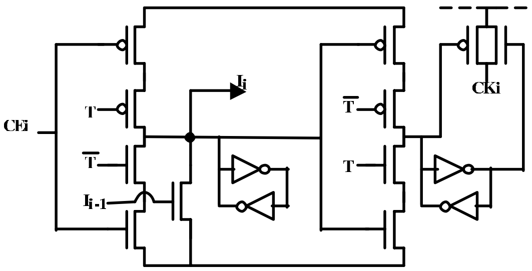


Fig. 7 The **PFF** and **TG** circuitry.

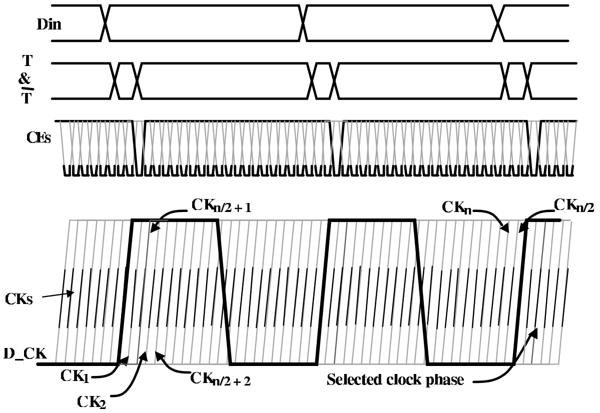


Fig. 8 Concept of operation of the proposed **CRC**.

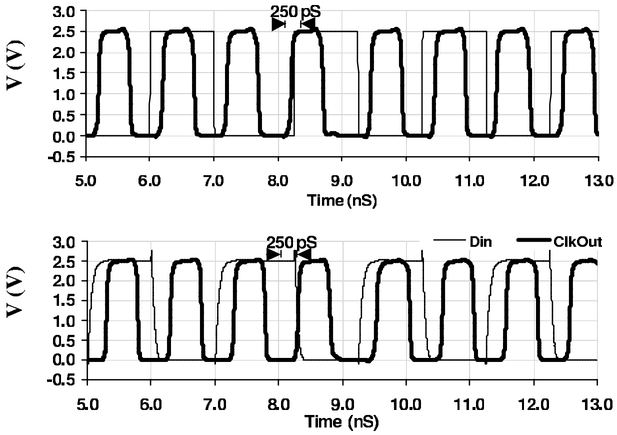


Fig. 9 Simulation results of the **NRZ** input and output clock of the **CRC** @ 1 Gb/s. An input phase error of 250 pS is injected and the **CRC** immediately retimes the output clock.

that correspond to the positive clock edges of the input clock as well as the delayed clock signal D_CK. The delay lines generate ***n*** phases of these two signals (for clarity, only the positive edges of the signals **CK1** - **CKn** are shown in Fig. 8). The ***n*** phases of the **PCE** signal ($\mathbf{CE}_1$ to $\mathbf{CE}_n$) are fed to the **ELCM** circuit. The **CE** signal(s) with the appropriate synchronization with the input data is latched-in by the corresponding **PFF** in the **ELCM** circuit. This in turn enables the appropriate clock phase **TG**(s). It is very important to keep the $\mathbf{CE}_i$s pulses as narrow as possible to obtain the highest resolution. When the number of generated clock phases, ***n***, is large enough, the delay line delay will span more than one clock cycle. This in effect folds back the clock phases and hence provide a resolution less than $2 \times \mathbf{T}_{Dinv}$ between successive phases. Also the **TG** mux and output clock inverter, will perform phase interpolation between selected phases, yielding even higher resolution. In Fig. 8, ***n*** is high enough to span the clock cycle twice. Higher resolution also means less generated jitter, since the delay separation between successive phases becomes less than an inverter's delay. ***n*** has to be odd (integer) to ensure that the folded clock phases do not fall into the positions of original phases. It is chosen to satisfy the required resolution, $\mathbf{t}_{res}$, at a clock period, $\mathbf{T}_{clk}$, simply as:

$$\mathbf{n} \geq \left\lceil \frac{\mathbf{T_{clk}}}{\mathbf{t_{res}}} \right\rceil_{\mathbf{Odd}}$$

Also, to ensure that the folded clock phases span the whole clock period (as in Fig. 8), the product $2 \times n \times \mathbf{T}_{Dinv}$ should be more than the required integer multiple of $\mathbf{T}_{clk}$. For example, if $\mathbf{T}_{clk}$ =1 nS, $\mathbf{T}_{Dinv}$ = 50 pS and the required resolution is 50 pS, then n should be chosen to be 21. Then the clock phases will span the clock period twice. But if $\mathbf{T}_{Dinv}$ = 40 pS, then ***n*** would have be 25 in order to span the clock period also twice. Figure 9 shows the simulation results for the **CRC** at 1 Gb/S input data. The **CRC** relocks to the input data within one data transition after the injection of 250 pS phase error into the input data for both, rising edge (upper figure) and falling edge (lower figure). In the case of a frequency mismatch between the input data and $\mathbf{Clk}_{in}$ two possible outcomes might result; 1) If there are enough data transitions in the input data the frequency mismatch will manifest itself as a data-dependent jitter in the output clock since the **CRC** would be selecting different clock phases every few data bits. 2) If there are no data transitions for long periods, the **CRC** will loose lock. This period depends on the frequency mismatch. For example if the frequency mismatch is 2,000 ppm, the **CRC** will loose lock after 500 consecutive **1** s or **0** s. Such a long bit stream of **1** s or **0** s represent an even bigger problem to conventional **PLL**-based **CRC**s. One possible solution is to supply $\mathbf{Clk}_{in}$ from an external **PLL** that tracks the data frequency. This **PLL** would only perform frequency acquisition and can be tailored to have minimal data-dependent jitter (very low bandwidth).

4. Performance Evaluation

The performance of the proposed **CRC** was evaluated as; 1) Performance under different static conditions (temperatures, phase shift between input data and input clock, and supply voltages), 2) Jitter generation due to supply noise and input clock jitter, and 3) Tracking of input data jitter. The **CRC** was implemented using a 0.25 μm, 2.5 V **CMOS** technology with a $\mathbf{T}_{Dinv}$ of 45 pS. The targeted resolution was to be better than 45 pS hence ***n*** was set to 23.

4.1 Static Performance

The **CRC** was simulated under different starting conditions in terms of the skew between input **NRZ** data (Din) and input clock ($\mathbf{Clk}_{in}$) for several temperatures in the range −25 to 100°C. At 25°C and **0** skew between **Din** and $\mathbf{Clk}_{in}$, the rising edge of the output

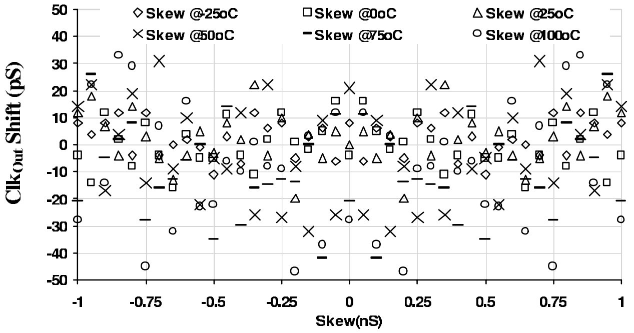


Fig. 10 Shifts in output clock edge relative to **Din** versus skew between **Din** and $\mathbf{CK}_{in}$ for several temperatures.

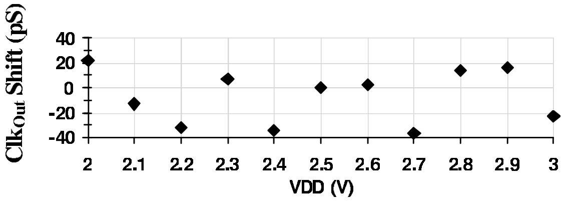


Fig. 11 Shifts in $\mathbf{Clk}_{Out}$ edge relative to **Din** vs. VDD.

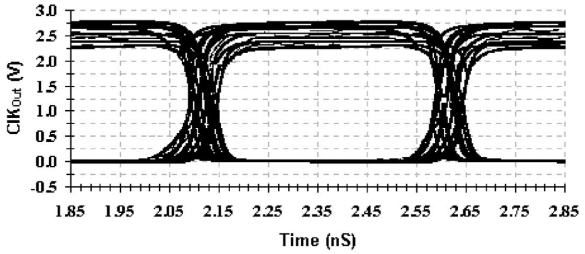


Fig. 12 The output clock jitter with a ±0.25 V and 100 MHz noise superimposed on the supply.

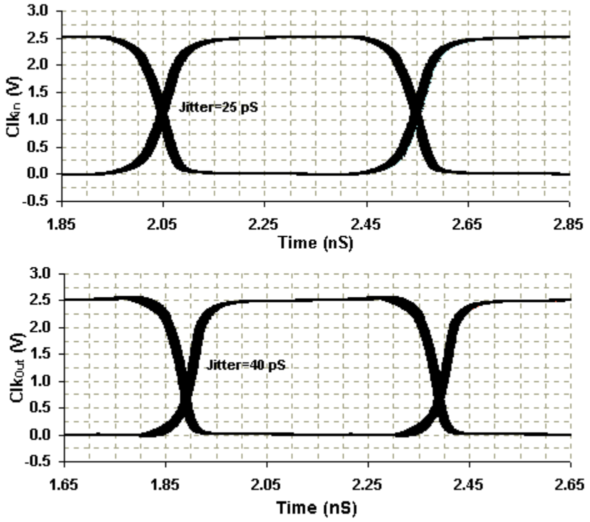


Fig. 13 The **CRC**'s input & output clock jitters.

clock ($\mathbf{Clk}_{Out}$) was delayed with respect to Din by about 200 pS. Changes in this delay (shifts) of $\mathbf{Clk}_{Out}$ are reported in the scatter graph of Fig. 10 for different temperatures and skews between Din and $\mathbf{Clk}_{in}$. As this figure shows, the shifts in $\mathbf{Clk}_{Out}$ range mostly between 25 and −25 pS with the spread increasing at higher temperatures. This is due to the increased delay of the inverters in the delay lines, which decreases the resolution. Also due to interpolation between adjacent clock phases the actual resolution is better than the designed value (for which clock interpolation was ignored). The effect of the supply voltage VDD on the output clock shifts is shown in Fig. 11. The supply was swept from −25% to +25% and again the shifts do not exceed the designed resolution. Actually significant portion of the shifts is due to the changes in the delay of the output clock buffer. This indicates that this buffer should be kept to a minimum number of stages.

4.2 Jitter Generation

Many sources contribute to jitter generation in conventional **PLL**-based **CRC**s (low loop gain, supply noise, data-pattern). In the proposed **CRC**, supply noise and input clock jitter will generate output jitter but the data pattern wont affect the jitter since there is no feedback. Many workers only report jitter generated due to low loop gain and input data pattern (quite supply jitter). Figure 12 shows the jitter in the output clock due to supply noise. A 100 MHz sinusoid with ±0.25 V amplitude (±10% of VDD) was superimposed on VDD and the output clock periods were overlaid on top of one another to emulate a jitter scope measurement. Each period was laid out twice, once as is, and another with half a period phase shift to account for effects of duty-cycle modulation on output jitter. A relatively high noise frequency was used to reduce the simulation time, which had to span hundreds of cycles. At such high noise frequency a feedback loop (e.g. a **PLL**) will not be able to track and correct the resulting jitter. The measured peak-to-peak jitter of the proposed **CRC** was 64 pS, a very low jitter at this supply noise level. This is consistent with the selected output clock resolution. To simulate the effects of input clock jitter, the input clock was frequency modulated with various values of the modulation index (X) and a modulation frequency of 50 MHz. Again the jitter frequency was made high enough to reduce the simulation times and the output files sizes, but in reality, the jitter frequency would not exceed 1 or 2 MHz. As X increases, the jitter increases, with the jitter being equal to $0.5 \times \mathrm{X} \times \mathbf{T}_{Clk}$. Figure 13 shows both $\mathbf{Clk}_{in}$ and $\mathbf{Clk}_{Out}$ signals with an input modulation index of 0.05. The $\mathbf{Clk}_{in}$ jitter is 25 pS while the $\mathbf{Clk}_{Out}$ jitter is 40 pS. Thus the jitter gain is about 1.6x (or 4.1 dB) at this modulation index.

Figure 14 shows the jitter gain in dBs as a function of $\mathbf{Clk}_{in}$ jitter. The gain decreases as the input jitter increases. The output jitter reaches a saturation level

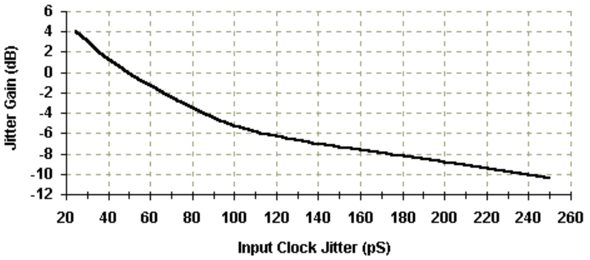


Fig. 14 The **CRC**'s output/input clock jitter gain in dBs.

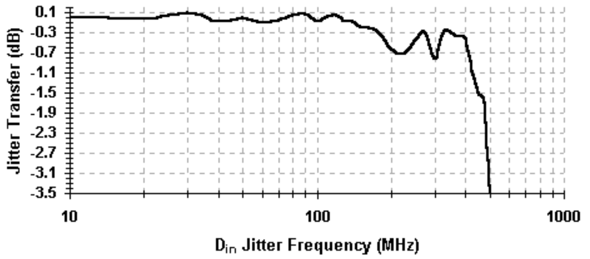


Fig. 15 The jitter transfer characteristics of the proposed **CRC** for a 1 **UI** peak-to-peak input data jitter and as a function of jitter frequency.

of 73 pS and will not increase further, a fact that is consistent with the designed resolution. These are very significant results since the proposed **CRC** utilizes an external input clock and hence jitter amplification is a serious concern.

4.3 Input Data Jitter Tracking

Ideally, the **CRC** is supposed to track any jitter in Din (i.e. Din to $\mathbf{Clk}_{Out}$ jitter gain should be 0 dB) in order to perform rapid re-timing of the output clock and reduce the bit-error-rate (**BER**). Feedback type **CRC**s can track this jitter up to certain frequencies that depend on the loop bandwidth. Beyond that, the **BER** would increases very rapidly and the **CRC**'s jitter tolerance would diminish. For the proposed **CRC**, the jitter tracking was evaluated using 1 unit interval (**UI**) peak-to-peak input data jitter at various frequencies. 1 **UI** jitter corresponds to changing the relative data position by one half of the data period (in this case 500 pS). The jitter gain (transfer) is reported in Fig. 15 as a function of the jitter frequency. The jitter gain does not exceed +0.1 dB and is within ±0.1 dB up to 150 MHz jitter frequency. Even beyond that frequency it does not fall quickly as in **PLL**-based **CRC**s. At 500 MHz, the jitter transfer drops only to −3.5 dB, a very remarkable performance compared to conventional **CRC**s. The average power consumption of the **CRC** at the maximum data rate of 1 Gb/s was 14.3 mW. The **CRC** was implemented using a digital standard cell library and no special design considerations were made to target a low power performance. Still, a power consumption of 14.3 mW at that data rate is a very good result. This circuit can easily be described using a synthesizable hardware description language (**HDL**) code, which makes it very portable from one process to another enabling **SOC** designs. Also, a **PLL** block can be used to replace the high frequency external input clock (only for frequency acquisition) as in [2], [4].

5. Conclusions

A comprehensive review of existing clock recovery techniques and their shortcoming was presented. A new non-**PLL**-based, fully digital clock recovery circuitry for **NRZ** data retiming was proposed. The new circuit was demonstrated using a 2.5 V, 0.25 μm **CMOS** technology at a 1 Gb/s data rate. Circuit simulations were utilized for performance evaluation. This **CRC**'s capture time was one data transition and it tracked both static and dynamic operating conditions. The jitter generation due to **VDD** noise or input clock jitter was within the designed resolution, and unlike conventional **CRC**s is data-pattern independent. The jitter transfer with a 1 **UI** input data jitter was within ±0.1 dB up to 150 MHz jitter frequency. Adding more stages to the delay lines and the **ELCM** circuit can enhance the resolution of the output clock and improve the jitter characteristics. This **CRC** is easily described by a synthesizable **HDL** code, making it very portable from one process to another and hence can support **SOC** designs.

Acknowledgement

The author is grateful for the facilities support provided by King Fahad University of Petroleum and Minerals.

References

[1] B. Razavi, "A 2.5-Gb/s 15-mW clock recovery circuit," IEEE J. Solid-State Circuits, vol.31, no.4, pp.472–480, April 1996.

[2] S.B. Anand and B. Razavi, "A CMOS clock recovery circuit for 2.5-Gb/s NRZ data," IEEE J. Solid-State Circuits, vol.36, no.3, pp.432–439, March 2001.

[3] K. Kishine, et al., "A 2.5-Gb/s clock and data recovery IC with tunable jitter characteristics for use in LAN's and WAN's," IEEE J. Solid-State Circuits, vol.34, no.6, pp.805–812, June 1999.

[4] H. Wang and R. Nottenburg, "A 1 Gb/s CMOS clock and data recovery circuit," Proc. 1999 IEEE International Solid-State Circuits Conference, pp.210–211, 1999.

[5] M. Rau, et al., "Clock/data recovery PLL using half-frequency clock," IEEE J. Solid-State Circuits, vol.32, no.7, pp.1156–1159, July 1997.

[6] S. Nakamura, et al., "A 4.25 GHz BiCMOS clock recovery circuit with an AV-DSPD architecture for NRZ data stream," Proc. 1997 IEEE International Solid-State Circuits Conference, pp.168–169.

[7] D.-L. Chen, "A power and area efficient CMOS clock/data recovery circuit for high-speed serial interfaces," IEEE J. Solid-State Circuits, vol.31, no.8, pp.1170–1176, Aug. 1996.

[8] P. Larson, "A 2-1600 MHz 1.2–2.5 V CMOS clock-recovery PLL with feedback phase-selection and averaging phase-interpolation for jitter reduction," Proc. 1999 IEEE International Solid-State Circuits Conference, pp.210–211.

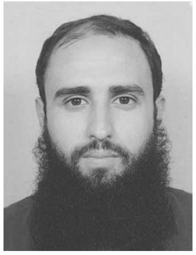
[9] M. Nakamura, et al., "A 156 Mbps CMOS clock recovery circuit for burst-mode transmission," Proc. 1996 IEEE Symposium on VLSI Circuits, pp.122–123.

[10] T.-Y. Hsu, et al., "An all-digital phase-locked loop (ADPLL)-based clock recovery circuit," IEEE J. Solid-State Circuits, vol.34, no.8, pp.1063–1073, Aug. 1999.

[11] M. Combes, et al., "A portable clock multiplier generator using digital CMOS standard cells," IEEE J. Solid-State Circuits, vol.31, no.7, pp.958–965, July 1996.

[12] T. Saeki, et al., "A 1.3-cycle lock time, non-PLL/DLL clock multiplier based on direct clock cycle interpolation for clock on demand," IEEE J. Solid-State Circuits, vol.35, no.11, pp.1581–1590, Nov. 2000.

[13] T. Saeki, et al., "A 2.5-ns clock access, 250-MHz, 256-Mb SDRAM with synchronous mirror delay," IEEE J. Solid-State Circuits, vol.31, no.11, pp.1656–1668, Nov. 1996.



Muhammad E.S. Elrabaa received his B.Sc. degree in computer Engineering from Kuwait University, Kuwait in 1989, and his M.A.Sc. and Ph.D. degrees in Electrical Engineering from the university of Waterloo, Waterloo, Canada, in 1991 and 1995, respectively. His graduate research dealt with Digital BiCMOS ICs and Low-Power circuit techniques. From 1995 till 1998, he worked as a senior circuit designer with Intel Corp., in Portland, Oregon, USA. He designed and developed low power digital circuits for Microprocessors. From 1998 till 2001 he was with the Electrical Engineering department, UAE University as an assistant professor. In 2001 he joined the computer engineering department, KFUPM as an assistant professor. His current research interest includes, integrated smart sensors, low-power circuits, and data communication circuits. He authored and co-authored several papers, a book and holds two US patents.