A Low-Power Multiplying DLL for Low-Jitter Multigigahertz Clock Generation in Highly Integrated Digital Chips

Ramin Farjad-Rad, *Member, IEEE*, William Dally, *Fellow, IEEE*, Hiok-Tiaq Ng, *Member, IEEE*, Ramesh Senthinathan, M.-J. Edward Lee, Rohit Rathi, and John Poulton, *Senior Member, IEEE*

***Abstract*—A multiplying delay-locked loop (MDLL) for high-speed on-chip clock generation that overcomes the drawbacks of phase-locked loops (PLLs) such as jitter accumulation, high sensitivity to supply, and substrate noise is described. The MDLL design removes such drawbacks while maintaining the advantages of a PLL for multirate frequency multiplication. This design also uses a supply regulator and filter to further reduce on-chip jitter generation. The MDLL, implemented in 0.18-μm CMOS technology, occupies a total of 0.05 mm^2 of active area and has a speed range of 200 MHz to 2 GHz with selectable multiplication ratios of $M =$ 4, 5, 8, 10. The complete synthesizer, including the output clock buffers, dissipates 12 mW from a 1.8-V supply at 2.0 GHz. This MDLL architecture is used as a clock multiplier integrated on a single chip for a 72 $\times$ 72 STS-1 grooming switch and has a jitter of 1.73 ps (rms) and 13.1 ps (pk–pk).**

***Index Terms*—Clock multiplication, clock synthesis, delay-locked loop (DLL), low jitter, phase-locked loop (PLL), serial links.**

I. Introduction

As THE bandwidth demand of computer and digital communications components continues to grow, high-speed serial I/O links are replacing traditional parallel buses. Operating at speeds of up to 5 Gb/s, such high-speed I/O circuits are already found in packet switches, circuit switches, and processor-memory interconnects. Hundreds of these high-speed I/Os have been successfully integrated on a single chip enabling monolithic switches with aggregate I/O bandwidths as high as 1 Tb/s [1]. By increasing the bandwidth per package pin and connector pin, high-speed I/Os reduce the system size and cost per unit bandwidth and meet the demand for higher bandwidth computer and communication systems.

Fig. 1 shows a high-level diagram of a high-speed serial I/O consisting of a serializing transmitter, a channel, and a deserializing receiver. To operate with a bit period that is small compared to the time-of-flight over the channel, high-speed I/O circuits are typically terminated with a matched impedance at either or both ends to achieve *incident-wave signaling* and recover the clock phase from the data arriving at the receiver. With such an approach, many bits can be wave-pipelined on the line—with several bits in flight at once. The bit period of such a system is limited by three factors as illustrated by the abstract eye diagram of Fig. 2. The bit period must be at least the sum of the rise time (t_r), the receiver aperture time (t_a) and the peak-to-peak timing uncertainty or noise (t_u) [2]. Notice that t_u represents all timing uncertainties in the system. In most systems, t_u is a dominant factor limiting bit rate.

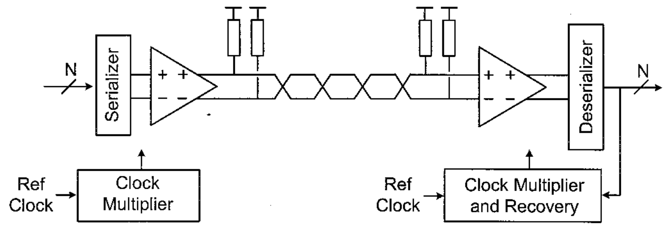
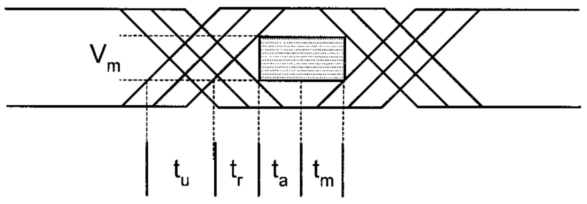


Fig. 1. Basic components of a high-speed serial I/O.



t_u: p-p timing uncertainty or noise
t_r: time required to switch state
t_a: the duration the signal must be above the voltage margin V_m
t_m: timing margin of the system

Fig. 2. Timing components of an eye diagram.

A major timing noise contributor in high-speed I/O systems is the clock multiplier, which takes a low frequency (e.g., 125 MHz) and, in most cases, accurate (low jitter), reference clock and synthesizes a high-frequency (e.g., 2 GHz) timing reference for the bit stream. Any jitter in the output of the clock multiplier adds directly to t_u. In conventional systems, the clock multiplier is typically implemented as a phase-locked loop (PLL) or using delay-locked loop (DLL) frequency synthesis.

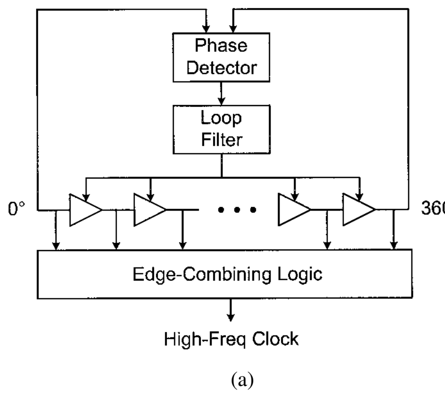
PLLs are attractive for use as clock multipliers because they provide a simple means of frequency multiplication and can support a programmable multiplication rate. Unfortunately, given an identical noise environment and circuit components, a PLL has higher jitter than a DLL due to phase noise accumulation [3]. A step variation in the ring oscillator delay causes a constant

Manuscript received March 31, 2002; revised June 15, 2002.

R. Farjad-Rad, H.-T. Ng, R. Senthinathan, M.-J. E. Lee, R. Rathi, and J. Poulton are with Velio Communications, Inc., Milpitas, CA 95035 USA.

W. Dally is with Computer Systems Laboratory, Stanford University, Stanford, CA USA. He is also with Velio Communications, Inc., Milpitas, CA 95035 USA.

Digital Object Identifier 10.1109/JSSC.2002.804340



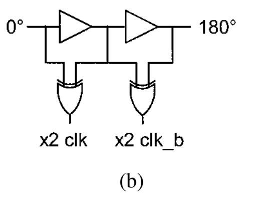


Fig. 3. (a) A DLL frequency synthesizer. (b) A simple frequency doubler.

offset in a DLL. In a PLL, however, the offset is integrated over many (ten or more) cycles until the loop filter can respond, resulting in a peak phase error considerably larger (typically three to four times) than the original phase variation. For an expanded comparison analysis of PLL and DLL jitter performance refer to [3] and [9]. This factor is especially important when one intends to integrate a large number of I/Os with other digital processing components, where a significant amount of power supply and substrate noise is present.

The above consideration has motivated the use of DLLs for frequency synthesis [6], [7]. Fig. 3 shows one possible scheme [7]. The equally spaced phases of the reference clock are processed through an edge-combining logic to produce a higher frequency clock. An example of a simple frequency doubler is also shown in Fig. 3. Besides no jitter accumulation, a DLL is mostly a single-pole system, does not rely on a high loop bandwidth to correct for jitter and exhibits negligible jitter peaking. However, the architecture as shown in Fig. 3 suffers from two major drawbacks. First, any mismatch in the delay element or the edge-combining logic translates directly into duty cycle error and fixed-pattern jitter. Second, a programmable clock multiplication ratio is difficult with this architecture.

To overcome the difficulties of previous PLL and DLL clock multipliers, we have developed a multiplying delay-locked loop (MDLL) clock multiplier. This circuit accepts an input clock and generates a phase-locked output clock at a multiple of the input clock frequency. As with a DLL, each rising edge of the input clock zeros the phase error of the loop. Hence this circuit combines the low phase noise of a DLL with the clock multiplication ability of a PLL. Because the same delay elements generate each edge of the output clock in an MDLL, the fixed-pattern jitter due to device mismatch in a conventional DLL frequency synthesizer is eliminated.

The basic idea of the frequency multiplying DLL is shown in Fig. 4. Each rising edge of the input clock, *rclk*, enters an inverting delay element via a multiplexer. After each edge passes, the multiplexer switches to select the output of the delay element, connecting it as a ring oscillator. After passing $N-1$ rising edges in this configuration (and the preceding falling edges), the multiplexer switches back to route the next rising edge on *rclk* into the delay element. When this edge arrives, it is compared to the rising edge on *bclk* and the delay element control voltage, V_{cntl}, is adjusted to align the two edges. Once the loop is locked, M pulses are generated on *bclk* for each input pulse on *rclk* and the rising edge of each Mth output pulse is aligned with the rising edge of each input pulse.

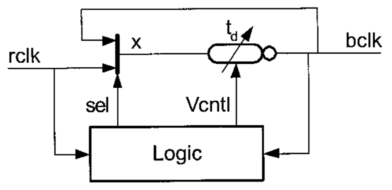


Fig. 4. Basic idea of the frequency multiplying DLL.

The advantages of the MDLL stem directly from this basic design. Each rising edge of *rclk* zeros the phase error of the output *bclk*. Thus, there is no phase error accumulation. Because a single delay element is used to generate the edges of *bclk*, there is no fixed pattern jitter due to device mismatch. Also, the multiplication rate is easily programmable by changing the number of cycles of *bclk* that are recirculated before switching the multiplexer.

The basic design of the MDLL also leads to two issues that require careful attention. First, the input *rclk* must be kept very clean since any jitter on this signal will be passed directly to the output and will all appear during a single cycle. This is not a serious issue as inexpensive crystal oscillators have sufficiently low jitter for this application. Second, any mismatch in the phase of *bclk* and *rclk* will result in fixed-pattern jitter that occurs on every Mth *bclk* edge. Reducing this fixed pattern jitter to acceptable levels requires a novel phase comparator design and careful attention to the design of the selection and multiplexing circuits.

The remainder of this paper describes the design and evaluation of our MDLL in more detail. Section II describes the overall architecture of the MDLL. Circuit details, including the design of the phase comparator, are described in Section III. Experimental results are presented in Section IV.

II. Multiplying DLL Architecture

The multiplying DLL (MDLL) architecture, shown in Fig. 5, provides the programmable clock multiplication without the associated jitter accumulation of a PLL and without the fixed pattern jitter due to device mismatch of a DLL frequency synthesizer [4]. The MDLL consists of a three-stage differential inverter DLL whose delay is controlled by adjusting its power supply voltage using a voltage regulator. The MDLL output is fed back to its input through a multiplexer. The differential 2 : 1 multiplexer selects between the clean reference clock *rclk*, when the *sel* is asserted by select logic, and the DLL output *bclk*, when *sel* signal is deasserted. The front-end clock buffer level converts the input PECL clock to match the DLL internal signal levels and transition times. A phase detector is used to align the edge of *rclk* to the corresponding edge of *bclk* when multiplexing hap-

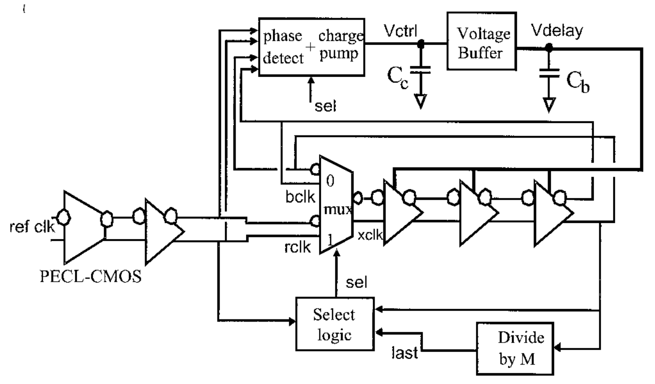


Fig. 5. Proposed MDLL.

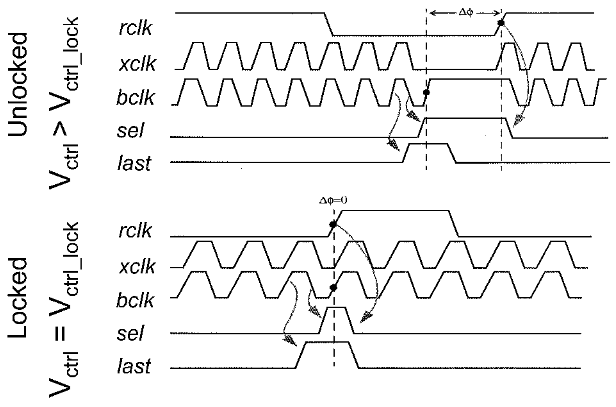


Fig. 6. Multiplying DLL timing.

pens. A divide-by-M counter, with adjustable divide ratio, provides a programmable multiplication ratio M for the MDLL.

When the DLL loop is closed through the multiplexer, a ring oscillator is formed that oscillates with a period twice that of the delay around the DLL loop composed of the inverters and multiplexer generating the high-speed bit clock, *bclk*. The divide-by-M counter divides down the *bclk* signal to generate a pulse, *last*, every M cycles of the *bclk*. The *last* signal triggers the select logic that switches the multiplexer input to pass a clean reference clock edge, *rclk*. This function resets the ring oscillator phase to the phase of the clean *rclk* edge removing any accumulated jitter over the past $M - 1$ cycles. Therefore, the maximum number of cycles that the ring oscillator accumulates jitter is limited to M. In contrast, a PLL typically accumulates jitter over at least $10M$ cycles—until its loop filter can act to correct the error.

Any mismatch between the swing levels, transition times, and timing alignment of the *rclk* edge to the corresponding *bclk* edge of the DLL stages results in signal distortion during clock multiplexing which leads to deterministic jitter in final multiplied clock. Therefore, the design of the clock buffer and phase detector circuits is very critical for the MDLL performance. A similar idea is used in [2]; however, lack of attention in its circuit implementation to the factors above makes its performance suitable only for much lower-frequency applications even in faster processes.

Fig. 6 shows the detailed timing diagram of the MDLL for two cases when the closed oscillation frequency is more than (during startup) and equal to (after settling) M times that of the reference clock. At circuit startup, the DLL control voltage is pulled high, setting the line delay to its minimum and the ring frequency to its maximum. As a result, the *bclk* completes its M cycles considerably before the following rising edge of *rclk*. At the Mth rising transition of *bclk*, the divide-by-M counter asserts the *last* signal, which in turn activates the select logic. The select logic asserts the *sel* signal at the following falling transition of *bclk*, switching the multiplexer to its *rclk* input. During this period the DLL stops oscillation and the multiplexer continues to select *rclk*. The phase detector, which is also enabled by the *sel* pulse high, compares the corresponding rising edge of the *bclk* with that of *rclk* and applies a correction pulse, proportional to the phase difference between the two clock edges $\Delta\phi$, to a charge pump circuit that moves the loop control voltage and thus the DLL delay toward smaller phase error. Finally, the select logic deasserts *sel* at the rising edge of *rclk* which restarts the ring oscillator—with its phase reset to the phase of *rclk*. This correction loop is repeated till the phase error $\Delta\phi$ is zero and the MDLL is locked. After the loop is locked, any phase offset in the phase detector and charge pump circuits directly shows up as cycle-to-cycle jitter on MDLL output when *rclk* is muxed in. Therefore, the phase offset in the phase detector and charge pump must be minimized or cancelled.

The MDLL open loop has one pole at the output of the charge pump (V_{ctrl}), and one pole at the regulated voltage node (V_{delay}) due to the large bypass capacitor C_b. Considering that the voltage buffer is a unity gain buffer, The open loop transfer function is

$$H_o(s) = \frac{K_p \cdot K_d \cdot M}{sC_c \cdot (sC_b \cdot R_o + 1)} \tag{1}$$

where K_p is the chargepump gain, K_d is the DLL gain, M is the multiplication factor, and R_o is the effective ac resistance at the regulated supply node. As can be seen from (1), unlike a PLL loop, the MDLL has only one pole at the origin and therefore does not require an extra zero to stabilize the feedback loop. To ensure the closed loop has acceptable phase margin, the unity gain crossover frequency of the open loop transfer function $(H_o(s))$ must be well below the second pole caused by the regulated supply capacitor. The worst case condition for stability is when the loop gain is highest, i.e., highest multiplication factor, highest DLL and chargepump gain. Maximum M is equal to ten in this design, and maximum charge pump and DLL gain happens at maximum speed of 2 GHz and fast corners of the process. However, note that R_o reduces at this corner, but it is a slower function of process corner and speed as the feedback in unity gain voltage buffer mainly controls the effective resistance at the node.

At maximum MDLL speed of operation and fastest corner of the process with $M = 10$, the loop parameters measured from simulations are $K_d = 0.75$ ns/V, $K_p = 11$ μA/ns, $R_o = 500\ \Omega$, $C_c = 10$ pF, and $C_b = 40$ pF, resulting in a worst case phase margin of $>60°$ at a unity gain bandwidth of $\sim$1.5 MHz.

One advantage of our MDLL is that it captures lock to any frequency below its reset frequency. Therefore, as long as the MDLL is started at its maximum startup voltage and thus its

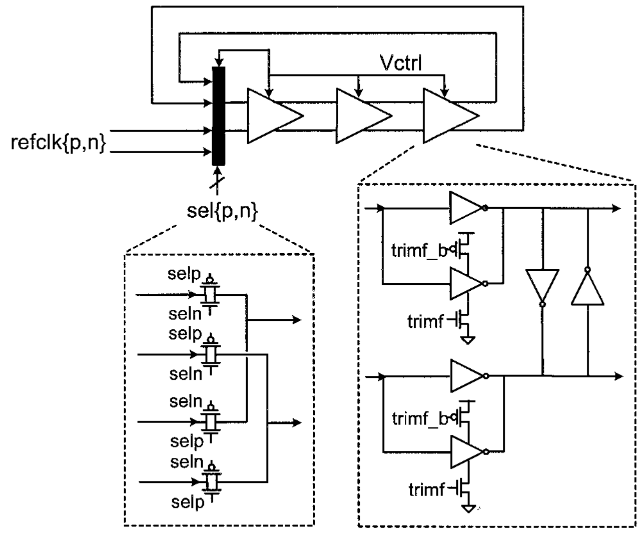


Fig. 7. Delay line schematic.

maximum frequency, there is no need for a frequency acquisition loop. Another advantage of this design is that it exactly aligns the phase of the output *bclk* signal to the phase of the input *rclk* signal—because the phase detector operates directly on these two signals. In contrast, conventional PLL implementations compare the phase of the reference clock with the output of the divider and hence do not precisely align the input phase with the output phase. A drawback of the MDLL architecture is that in a rare case of a missing *rclk* edge, there will be no output *bclk* for a *rclk* cycle, Fig. 6, and the charge pump charges the loop control voltage down during this period. The control voltage drop may result in a low enough oscillation frequency that can make recovery impossible in the next cycle. This situation is avoided by adjusting the charge pump current low enough so that any control voltage reduction during a *rclk* period in this event is small enough, and the oscillation frequency is still within MDLL capture range. A lock detector is also implemented in the MDLL to reset the circuit in the rare event of multiple missing *rclk* edges.

III. Circuit Implementation

A. Delay Line

Fig. 7 shows the delay line schematic. The clock selection multiplexer is implemented with complementary transmission gates. The delay elements are simple CMOS inverters whose delay is varied through supply regulation. As the control voltage approaches the transistor threshold voltage, the gain and the supply sensitivity of the delay line increases significantly. To alleviate this problem, a secondary shunting inverter controlled by the binary programmable signal *trimf* is placed around the main inverter stage to prevent the control voltage from becoming excessively low at fast corner and low frequencies. The delay element is slower with *trimf* low than high. Cross-coupled inverters are inserted at each delay stage to suppress skew between complementary clock signals.

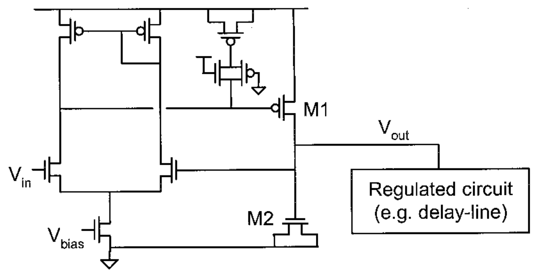


Fig. 8. Supply regulator schematic.

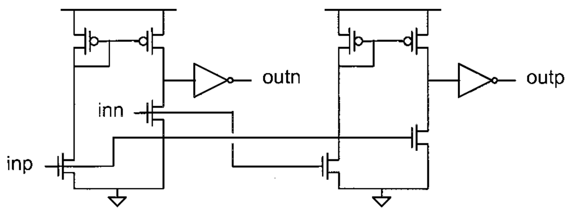


Fig. 9. Level converter schematic.

B. Supply Regulator

The goals of the supply regulator are to provide enough driving capability to regulate the clock multiplier and to shield the clock multiplier from the supply noise while having a high bandwidth (>300 MHz) to ensure the DLL stability. Shown in Fig. 8, this design uses the same topology as well as stability compensation scheme as [5]. For high frequency supply noise beyond the loop bandwidth of the supply regulator, the output noise is mostly determined by capacitive feed through as well as sudden current surge due to instantaneous V_{GS} change in $M1$. To minimize these effects, a large bypass capacitor $M2$ is placed at the output. Therefore, at lower frequencies a high power supply rejection ratio (PSRR) is provided by the supply regulator feedback loop, and at higher frequencies high PSRR is guaranteed by the large bypass capacitor. This design has a PSRR of at least 20 dB across the whole frequency spectrum, reducing the supply noise sensitivity of the delay line by at least $10\times$.

C. Level Converter

The supply regulated delay line produces clock signals that swing between ground and the control voltage. A level converter is inserted whenever a rail-to-rail clock signal is required. Fig. 9 shows the schematic of the level converter. The first current mirroring stage has positive supply sensitivity while the subsequent inverter stage has negative supply sensitivity. The fan out of the two stages can be adjusted so that the overall supply sensitivity is significantly less than either stage to minimize jitter [5].

In addition, a level converter is also required at the reference clock input to convert the PECL level signal to the CMOS level under the regulated supply. Fig. 10 shows the schematic. The front-end *RC* network provides ac coupling and biases the input signal at the transistor threshold voltage. The preceding common-source amplifier uses cross-coupled negative resistance to boost gain. A current-mirroring amplifier to provide

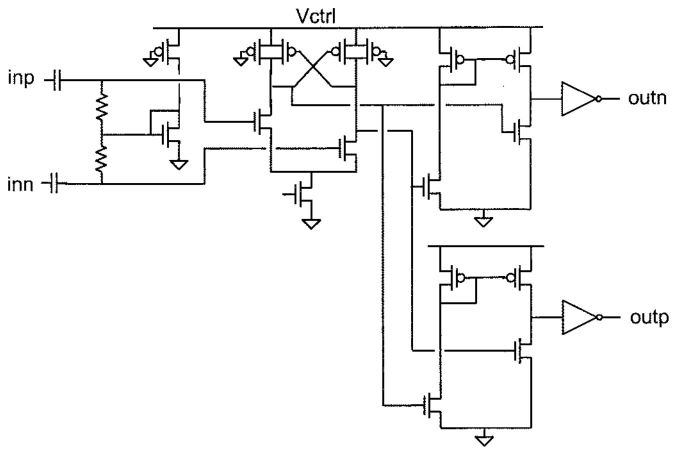


Fig. 10. Reference clock input level converter.

full CMOS levels follows this stage. All circuits are under the regulated supply to minimize supply noise sensitivity and provide proper signal levels for the delay line.

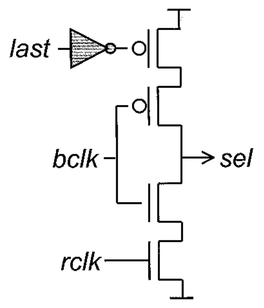
D. Select Logic

The timing of the select logic is critical at very high frequencies and can affect the MDLL jitter performance. If the select logic does not completely switch the mux input before the appropriate clock edge passes through the mux, the clock edge experiences a distortion that results in extra delay and therefore a phase error in that cycle. To avoid this condition, the *sel* signal should have fast transitions and be asserted immediately after the falling edge of the last *bclk* and deasserted immediately after the rising transition of *rclk*. Therefore, the dynamic CMOS gate shown in Fig. 11(a) is used to increase the speed and slew rate of the select logic. Note that *sel* is deasserted when both *rclk* and *bclk* go high. This is done to guarantee a minimum *sel* pulse width when the *rclk* rising edge advances the *bclk* rising edge following the *bclk* falling edge that asserted the *sel* pulse. Fig. 11(b) show acceptable and unacceptable cases of the select pulse, where the unacceptable select pulse is late with a large rising time, therefore distorting the following edge of *bclk*.

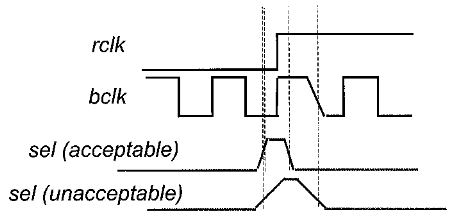
E. Combined Phase Detector and Charge Pump

Fig. 12 shows the block diagram of the combined phase detector and charge pump circuit. The phase detector is composed of two AND gates that are activated during the select pulse window. The AND gates take advantage of the available complementary clocks to generate a high output pulse whose width is equal to the phase lag (UP gate) or lead (DOWN gate) between *rclk* and the corresponding edge of *bclk*, Fig. 12. Device mismatches and asymmetries in the phase detector and charge pump result in phase error in the multiplied clock, as discussed earlier.

To minimize phase error, a combined phase detector and charge pump circuit performs the AND operations using series FETs to generate narrow current pulses as shown in Fig. 13, [8]. Each 3-input AND gate is constructed by a stack of three NMOS devices that gate the source of an NMOS current source device in a current mirror. When all devices in the stack of three turn on, the NMOS current source pumps a fixed current



(a)



(b)

Fig. 11. (a) Select logic circuit. (b) Example of bad and good select pulse.

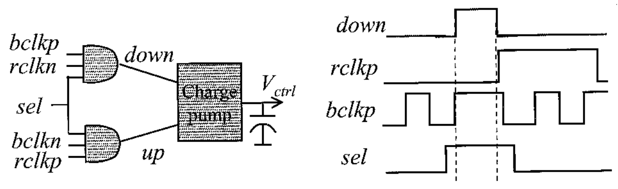


Fig. 12. Phase detector and charge pump.

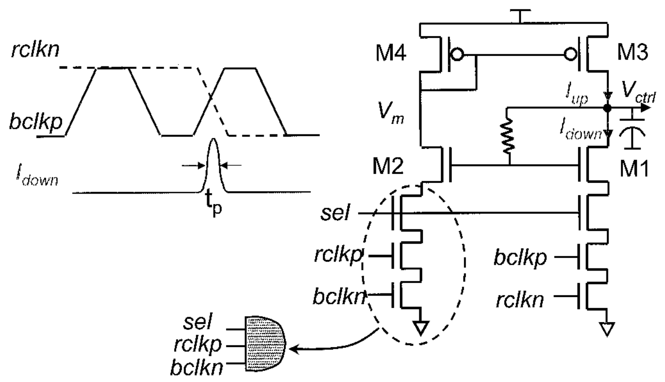


Fig. 13. Combined phase detector and charge pump.

(I_{up} or I_{down}) into the loop capacitor. This choice of AND gate implementation is to generate a narrow current pulse when the loop is locked, (as shown in Fig. 13) which minimizes the phase error caused by the charge pump current mirroring. This current pulse is significantly narrower than what is possible with a voltage signal that must charge and discharge parasitic capacitances.

Phase errors in the charge pump are also due to the channel length modulation of devices $M1$–$M4$, and size mismatch of all transistors in Fig. 13, resulting in I_{up} and I_{down} current mismatch for zero input phase difference. The phase or timing error Δt_{er} of the charge pump when locked is related to the current mismatch factor α (i.e., $I_{\text{up}} = \alpha I_{\text{down}}$ for zero phase error) and

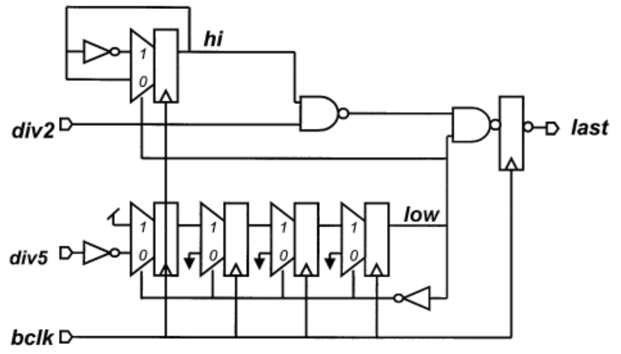


Fig. 14. Clock divider schematic.

the width of the current pulse t_p. When the loop is locked and has settled, the up and down charges pumped into the loop capacitor are equal:

$$Q_{\text{up}} = Q_{\text{down}} \Rightarrow (t_p - \Delta t_{\text{er}})I_{\text{up}} = (t_p + \Delta t_{\text{er}})I_{\text{down}} \quad (2)$$

$$\Rightarrow \Delta t_{er} = [(1-\alpha)/(1+\alpha)]t_p. \quad (3)$$

As shown by (2), there are two ways to reduce the phase or timing error Δt of the combined phase comparator and charge pump: 1) reduce α, the current mismatch in the charge pump and 2) using a very narrow current pulse (small t_p) that is achieved by the stacked NMOS AND gate. To reduce α, the NMOS and PMOS pairs in the current mirror ($M1$–$M4$) must have large area to reduce size mismatch and long channel length (>2 μm) to reduce the channel length modulation effect on output current when the voltage difference between the two sides of the charge pump (V_{ctrl} and V_m) is large. Implementing the observations above and adjusting the device sizes for best performance, the phase comparator and charge pump combination shows simulated phase errors of less than ± 5 ps across process, voltage, and temperature.

The gate of the two NMOS devices ($M1$ and $M2$) are biased at the output voltage of the charge pump through a large resistor to ensure the current source NMOS devices are always in saturation for different values of charge pump output (V_{ctrl}). This self-biasing scheme also causes the charge pump current to track the process, supply voltage, and temperature variations in the chip. For example, at high temperature and with weaker devices, the DLL requires a higher V_{ctrl} to maintain the correct frequency. Thus the gate bias voltages of the charge pump's weak NMOS devices are also higher to adjust their currents.

F. Clock Divider

The clock divider takes the multiplied clock and generates a pulse every M cycles. This pulse is used by the select logic to mux in the fresh reference clock edge every M cycles. The clock multiplication factor M is programmable and can be 4, 5, 8, or 10. Fig. 14 shows the schematic of the clock divider. The divider is implemented as a divide-by-4 or -5 stage (*low*) and a divide by 2 stage (*hi*). The low stage is a free running thermometer counter that produces a pulse on *low* every four or five cycles depending on *div5* being 0 or 1. The *hi* signal toggles on every *low* pulse. If *div2* is 0, the *low* pulse passes to the output directly giving a divide-by-4 or -5 function. On the other hand, if *div2*

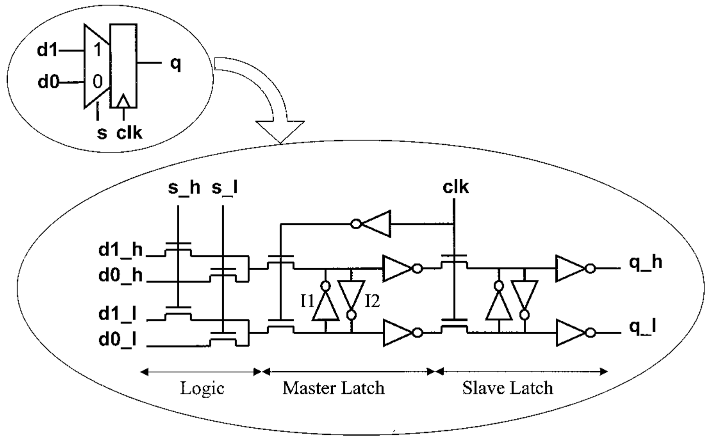
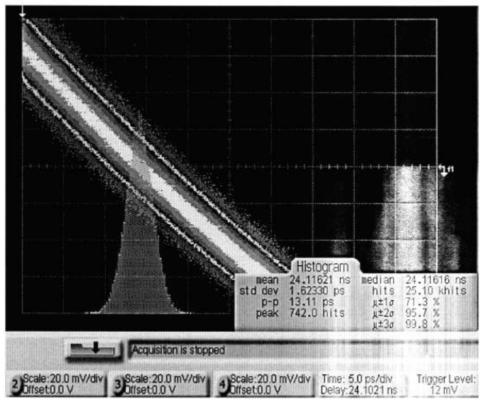


Fig. 15. CPL style (multiplexer merged with flop).



```
Histogram
    mean  24.11621 ns   median  24.11616 ns
 std dev  1.62330 ps      hits  25.10 khits
     p-p  13.11 ps        µ±1σ  71.3 %
    peak  742.0 hits      µ±2σ  95.7 %
                          µ±3σ  99.8 %
```

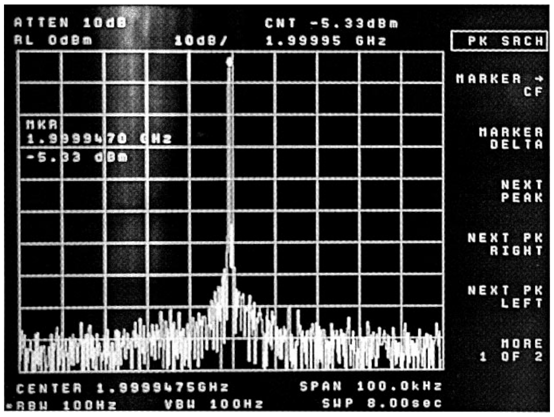
```
Acquisition is stopped
2 Scale:20.0 mV/div   3 Scale:20.0 mV/div   4 Scale:20.0 mV/div   Time: 5.0 ps/div       Trigger Level:
  Offset:0.0 V          Offset:0.0 V          Offset:0.0 V        Delay:24.1021 ns        12 mV
```

Fig. 16. Differential TX output jitter as an indicator of MDLL jitter performance at 2.0 GHz.

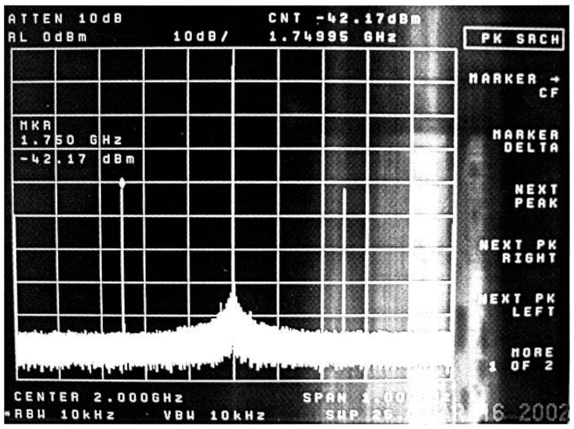
is 1, only every other *low* pulse is passed through, resulting in a divide-by-8 or -10 function.

To ensure correct functionality, the divider should be able to keep up with the VCO at all the times. The startup frequency of the VCO could be up to 4.5 GHz, thus requiring the divider to be able to operate at least that fast. This is achieved through careful logic and circuit design. As can be seen, there is a maximum of one gate between any two critical flops. It is important to note that the path from the *hi* flop to the *last* flop is a multicycle path. Also, both the low and hi counters are self-starting, enabling a faster design by avoiding the reset logic. This has the additional advantage of recovering from an illegal state without requiring a reset. Yet another advantage of the thermometer counter is that the *last* signal is automatically a pulse without requiring extra logic.

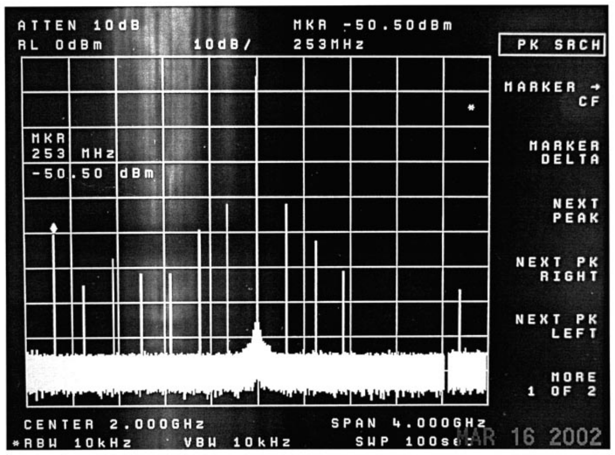
Fig. 15 shows the complementary pass-gate logic (CPL) style used to implement the divider. CPL is a simple and yet very fast logic style because inversions required to implement logic functions can be avoided. This avoids additional inverter delays, which can be prohibitive at high frequencies. The flip-flop is implemented in a similar differential style with the logic merged directly into the flip-flop input. This further enhances speed by eliminating the final level restoring stage typical of most CPL gates. Instead, the cross-coupled inverters $I1$ and $I2$ in the master latch provide the required level conversion.



(a)

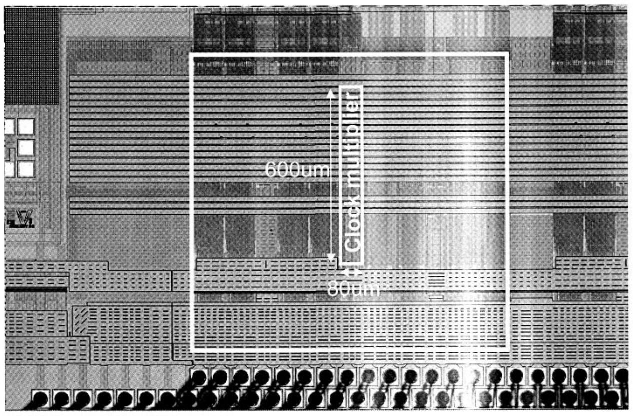


(b)

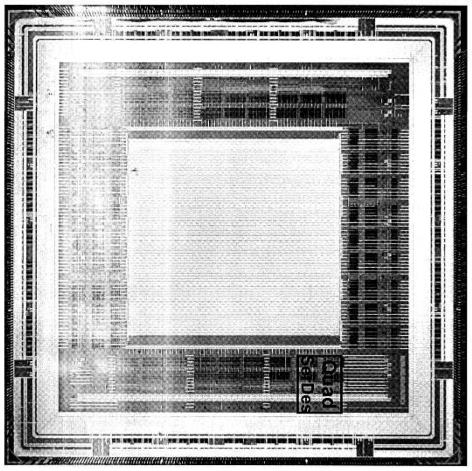


(c)

Fig. 17. (a) Spectrum of TX output showing MDLL performance. Span of 100 kHz. (b) Spectrum of TX output showing MDLL performance. Span of 1 GHz showing spurs at carrier ±250 MHz. (c) Spectrum of TX output showing MDLL performance. Span of 4 GHz showing reference clock feedthrough.



(a)



(b)

Fig. 18. (a) Micrograph of a quad SerDes block. (b) Micrograph of a 72 × 72 grooming switch.

IV. Experimental Results

The jitter performance of the MDLL can only be observed via a serializer's transmitter output when the output data has an nonreturn to zero (NRZ) alternating 1 and 0 data pattern. An alternating 1 and 0 data pattern ensures that the resulting serializer transmitter output is free from intersymbol interference (ISI) that will manifest itself as deterministic jitter, obscuring the true random jitter performance of the MDLL. Care has been taken that the serializer transmitter output is properly terminated to avoid reflections that will corrupt the jitter readings.

The output clock of the MDLL passes through several more stages of buffering before it is used by the srializer's transmitter. Thus, the jitter performance data shown here should be slightly pessimistic. The quality of the external reference clock is also important in determining the quality of the MDLL. Low jitter crystal oscillators with 1 ps of RMS jitter were used.

Fig. 16 shows the jitter performance of the MDLL when $M = 8$ is used in conjunction with a 250-MHz reference clock to produce a 2.0 GHz internal clock. The 1σ random jitter of 1.62 ps is obtained after 25k waveform samples have been collected on an Agilent 86 100A oscilloscope. When the same measurement is performed on a 72 × 72 STS-1 grooming switch, the 1σ random jitter increased to 1.73 ps due to higher switching noise on the power supply and through the substrate. Fig. 17 shows the spectrum of the serializer's transmitter output. Fig. 17(a) shows the low phase noise of the transmitted signal while Fig. 17(b) shows the 250 MHz reference clock spurs that are 37 dBm below the carrier frequency. The reference clock feedthrough is shown in

TABLE I
SUMMARY OF MDLL PERFORMANCE DATA

Process Technology	0.18μm standard CMOS
Supply Voltage	1.8V
Active Area	0.05mm^2
Power @ 2.0GHz	12mW
Frequency Range	200MHz – 2.0GHz
Random jitter @ 2.0GHz (1σ)	1.64ps (250MHz X 8) 1.73ps (250MHz X 8, 72 X 72 grooming switch)
Random jitter @2.0GHz (peak-to-peak)	13.11ps (250MHz X 8)
Deterministic jitter @2.0GHz	12.00ps (250MHz X 8)
Random jitter @2.0GHz, 1.65V (1σ)	1.89ps (250MHz X 8)
Random jitter @2.5GHz, 2.00V (1σ)	1.94ps (250MHz X 10)
Random jitter @1.25GHz (1σ)	1.74ps (250MHz X 5)
Random jitter @1.25GHz (1σ)	1.99ps (125MHz X 10)

Fig. 17(c), where it is approximately 45 dBm below the carrier frequency. When the power supply is reduced from 1.80 to 1.65 V, the MDLL jitter performance worsened to a 1σ random jitter of 1.89 ps. With a power supply voltage of 2.0 V, MDLL operation at 2.5 GHz was possible with a 1σ random jitter of 1.94 ps.

Lower multiplication ratios mean that the accumulated jitter in the MDLL while in VCO mode is reset more frequently as compared to a high multiplication ratio. This effect can be seen in the case of a 250-MHz reference clock with a multiplication ratio of $5\times$, where the 1σ random jitter is 1.74 ps compared to when a 125-MHz reference clock is used with $M = 10$, where the 1σ random jitter is 1.99 ps.

Fig. 18(a) is a die micrograph detailing the location of the MDLL clock multiplier in a quad SerDes block. Fig. 18(b) is another die micrograph detailing several clock multipliers used in the above mentioned 72×72 grooming switch. The performance of the MDLL is summarized in Table I.

V. CONCLUSION

We have developed an MDLL clock multiplier that combines the flexible frequency multiplication properties and low fixed-pattern jitter of a phase-locked loop with the low random jitter of a DLL. By operating the delay elements of the circuit as a ring oscillator, an arbitrary frequency multiplication ratio can be achieved and fixed-pattern jitter due to delay element mismatch is eliminated. By zeroing the phase error on every input clock edge, phase error accumulation is eliminated resulting in much lower random jitter than a PLL built from identical components.

The performance of our MDLL depends on the implementation of several critical circuits. A combined phase comparator and charge pump circuit computes phase difference using narrow current pulses to reduce the fixed-pattern jitter due to phase comparator offset. A power supply regulator with greater than 20 dB of PSRR reduces jitter due to power supply noise. Also, a fast, self-resetting divide-by-M counter is critical to proper initialization of the circuit.

ACKNOWLEDGMENT

The authors would like to acknowledge the contributions of T. Ansari, T. Stone, D. Hartman, V. Radulescu, K. Kinder, C. Okruhlica, M. Pimentel, S. Tell, S. Gowni, and J. Edmondson.

REFERENCES

[1] F. Heaton *et al.*, "A single chip terabit switch," in *Proc. Hot Chips Symp.*, Stanford, CA, Aug. 2001.
[2] W. J. Dally and J. Poulton, *Digital Systems Engineering*. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[3] B. Kim, T. Weigandt, and P. Gray, "PLL/DLL system noise analysis for low jitter clock synthesizer design," in *Proc. Int. Symp. Circuits and Systems*, vol. 4, 1994, pp. 31–38.
[4] W. J. Dally *et al.*, "Clock multiplying delay-locked loop for data communication," U.S. patent pending.
[5] M.-J. E. Lee, W. J. Dally, and P. Chiang, "Low-power, area efficient, high speed I/O circuit techniques," *IEEE J. Solid-State Circuits*, vol. 35, pp. 1591–1599, Nov. 2000.
[6] A. Waizman, "A delay line loop for frequency synthesis of de-skewed clock," in *Dig. Tech. Papers, IEEE ISSCC*, Feb. 1994, pp. 298–299.
[7] G. Chien and P. Gray, "A 900MHz local oscillator using a DLL-based frequency multiplier technique for PCS applications," in *Dig. Tech. Papers, IEEE ISSCC*, Feb. 2000, pp. 202–203.
[8] W. J. Dally *et al.*, "A combined phase comparator and charge pump circuit," U.S. Patent 6 275 072 B1, Aug. 2001.
[9] T. H. Lee *et al.*, "A 2.5V CMOS delay-locked loop for an 18 Mbit, 500Mbyte/s DRAM," *IEEE J. Solid-State Circuits*, vol. 29, pp. 1491–1496, Dec. 1994.

Ramin Farjad-Rad (S'95–M'00) was born in Tehran, Iran, in 1971. He received the B.Sc. degree in electrical engineering from Sharif University of Technology, Tehran, in 1993 and the M.Sc. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1995 and 1999, respectively.

He worked at SUN Microsystems Laboratories on a 1.25-Gb/s serial transceiver for fiber channeled and gigabit ethernet standards in the summer of 1995. During the summer of 1996, he was with LSI Logic, where he examined different multi-Gb/s serial transceiver architectures. He was part of the founding team of Velio Communications, Inc., Milpitas, CA, a communication chip manufacturer, where he has been in charge of multigigahertz CMOS circuit development as the company's research lead since 1999. He holds nine patents and patents pending in the area of high-speed integrated circuits, and has more than ten conference and invited journal papers.

Dr. Farjad-Rad ranked third in the 20th International Physics Olympiad in 1989.

William Dally (S'78–M'80–SM'01–F'02) received the B.S. degree in electrical engineering from Virginia Polytechnic Institute, Blackburg, the M.S. degree in electrical engineering from Stanford University, Stanford, CA, and the Ph.D. degree in computer science from the California Institute of Technology (Caltech), Pasadena.

He and his group have developed system architecture, network architecture, signaling, routing, and synchronization technology that can be found in most large parallel computers today. While at Bell Telephone Laboratories, he contributed to the design of the BELLMAC32 microprocessor and designed the MARS hardware accelerator. He was a Research Assistant and then a Research Fellow at Caltech where he designed the MOSSIM Simulation Engine and the Torus Routing Chip which pioneered wormhole routing and virtual-channel flow control. While a Professor of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology, Cambridge, he and his group built the J-Machine and the M-Machine, experimental parallel computer systems that pioneered the separation of mechanisms from programming models and demonstrated very low overhead mechanisms for synchronization and communication. He is currently a Professor of Electrical Engineering and Computer Science at Stanford University where his group has developed the Imagine processor, which introduced the concepts of stream processing and partitioned register organizations, and low-power high-speed signaling technology. He has worked with Cray Research and Intel to incorporate many of these innovations in commercial parallel computers, with Avici Systems to incorporate this technology into Internet routers, and cofounded Velio Communications, Inc., Milpitas, CA, to commercialize high-speed signaling technology. He currently leads projects on high-speed signaling, computer architecture, and network architecture. He has published over 150 papers in these areas and is an author of the textbook *Digital Systems Engineering* (Cambridge, U.K.: Cambridge Univ. Press, 1998).

Prof. Dally has received numerous honors including the ACM Maurice Wilkes Award.

Hiok-Tiaq Ng (S'93–M'01) was born in Penang, Malaysia. He received the B.A. degree in mathematics and physics from Whitman College, Walla Walla, WA and the B.S.E.E. degree from the California Institute of Technology, Pasadena, both in 1992, the M.S.E.E. degree from Carnegie Mellon University, Pittsburgh, PA, in 1994, and the Ph.D. degree from Arizona State University, Tempe, in 1999. His master's work focused on low-voltage low-power current steering logic families. His doctoral work focused on low-voltage, highly-linear, high-frequency continuous-time filters using multi-stage operational amplifiers.

His other research work involved designing read channel circuits for hard disk drives. From 1994 to 1995, he was a mixed-signal circuit design engineer at Texas Instruments, Dallas, TX. Since 1999, he has been at Velio Communications, Inc., Milpitas, CA where he is now designing high integration clock multiplier circuits for high-speed serial links.

Dr. Ng is a member of Sigma Xi, Tau Beta Pi and has received an SRC Inventor's Recognition Award.

Ramesh Senthinathan received the B.S. degree in computer engineering from the State University of New York at Buffalo in 1984 and the M.S. and Ph.D. degrees in electrical engineering from the University of Arizona, Tucson, in 1986 and 1992, respectively.

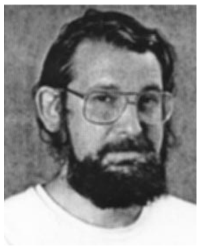
He is Director of engineering at Velio Communications, Inc., Milpitas, CA, responsible for technology and product development. From 1995 to 2000, he was with Intel Corporation as a design manager for the microprocessor group in Folsom, CA. He was responsible for all aspects of design and integration of Pentium Pro (shrink), Pentium Pro (1 Meg), and the flagship first Pentium III microprocessor design. With Level One communication acquisition, he moved to the Intel DSL group as a Distinguished Engineer responsible for analog front-end and UDSL design. During 1993 to 1995, he worked at Motorola, Inc., as a Principal Engineer responsible for 16-bit DSP (56K Motorola DSP) design for cellular communications. He was a Staff Design Engineer in the IBM mainframe computer group in Fishkill, NY, from 1992 to 1993. During 1986 to 1989, he was a design engineer with ASIC and microcontroller groups, Intel Corp., Chandler, AZ.

M.-J. Edward Lee received the B.S. degree in electrical engineering and computer sciences from the University of California, Berkeley, in 1997 and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 2000 and 2001, respectively.

During the summer of 1996, he was with HP Labs, Palo Alto, CA, working on a RF switched-mode power amplifier. Since 2000, he has been with Velio Communications, Inc., Milpitas, CA, where he leads the next generation gigabit serial link development.

Rohit Rathi received the B.Tech. degree in electrical engineering from Indian Institute of Technology, New Delhi, in 1996.

He was with Texas Instruments, Dallas, TX, in the DSP group from 1996 to 1999, working on several processors designs. He was responsible for logic and circuit design as well as for developing several in-house CAD tools for clock and power distribution and analysis. He was with National Semiconductor for a year working on an embedded microprocessor and later with ATI where he designed the MMX unit of an x86 processor. He then joined Velio Communications, Inc., Milpitas, CA, in 2001 where he has been working on logic and circuit design for high-speed IO cores. His technical interests are high-speed/low-power logic and circuit design and IC CAD tool development.



John Poulton (M'85–SM'90) received the B.Sc. degree in physics from Virginia Polytechnic Institute, Blackburg, in 1967, the M.Sc. degree in physics from the State University of New York at Stony Brook in 1969, and the Ph.D. degree in physics from the University of North Carolina at Chapel Hill in 1980.

From 1981 to 1999, he was a Researcher in the Department of Computer Science at the University of North Carolina at Chapel Hill, where from 1995 he held the rank of Research Professor. He did research on VLSI-based architectures for graphics and imaging and was a principal contributor to the design and construction of several experimental high-performance graphics systems. Since 1999, he has served as Chief Engineer for Velio Communications, Inc., Milpitas, CA, where he is engaged in the development of gigabit signaling systems.