# Skew-Tolerant Domino Circuits

David Harris, *Student Member, IEEE*, and Mark A. Horowitz, *Senior Member, IEEE*

***Abstract*—Domino circuits are widely used in high-performance CMOS microprocessors. However, textbook domino pipelines suffer significant timing overhead from clock skew, latch delay, and the inability to borrow time. To eliminate this overhead, some designers provide multiple overlapping clock phases such that domino gates are always ready for evaluation by the time critical inputs arrive and do not precharge until the next gate consumes the result. This paper describes a systematic framework, called skew-tolerant domino circuits, for understanding and analyzing domino circuits with overlapping clocks. Simulations confirm that a speedup of 25% or more can be achieved over textbook domino circuits in high-speed systems.**

***Index Terms*—Adders, clock skew, clocks, CMOS digital integrated circuits, dynamic logic, VLSI circuit design.**

## I. Introduction

SINCE microarchitectural improvements have been yielding diminishing returns, microprocessor designers seeking high performance have been forced to aggressively reduce cycle times beyond that which simple process scaling would permit. We can normalize cycle time improvement due to faster processes by expressing cycle time in terms of the delay of a fanout-of-four (FO4) inverter, i.e., an inverter driving a load that is four times its input capacitance. Today's fastest microprocessors are operating at cycle times below 18 fanout-of-four inverter delays [1].[1] Domino circuits [2] are an important enabler for this cycle time improvement [3]–[5]. At such short cycle times, however, clocking overhead which was once negligible becomes a significant fraction of the clock period.

As we will see in Section II, when domino circuits are pipelined in the same way that two-phase static circuits have traditionally been pipelined, they are highly sensitive to clock skew, include latch delays on the critical path, and are incapable of borrowing time across clock phases to balance the pipeline. Some designers have discovered that by overlapping the clocks controlling domino gates, these sources of overhead can be hidden, as we illustrate in Section III. We proceed to analyze domino gates using overlapping clocks in a systematic framework which we call skew-tolerant domino. Section IV presents the analysis under a single clock skew budget. Even more global clock skew can be hidden if we take advantage of tighter bounds on local clock skew, as described in Section V. For many reasonable designs, this global skew tolerance greatly exceeds the actual system skews, so Section VI explains how to take advantage of the extra overlap to allow time borrowing across phases. Section VII then addresses the critical issue of clock generation and shows how a single global clock and relatively simple local clock generators can produce the needed clock phases, while Section VIII looks at the interfaces of skew-tolerant domino with static and self-timed logic. Section IX presents simulation results of skew-tolerant domino applied to an adder self-bypass path. Finally, Section X summarizes the skew-tolerant domino techniques and the performance benefits which they offer.

Manuscript received April 10, 1997; revised August 5, 1997. This work was supported in part by a National Science Foundation fellowship, by Stanford's Center for Integrated Systems, and by DARPA Contract DABT63-94-C-0054.

The authors are with Stanford University, Stanford, CA 94305 USA.

Publisher Item Identifier S 0018-9200(97)08035-9.

[1] DEC reports an Alpha 21164 cycle time of 14 "gate delays" where a "gate delay" is roughly an average fanout-of-three two-input gate. Simulation found that the average of a two-input fanout-of-three NAND and NOR delay is about 1.24 fanout-of-four inverter delays.

## II. Textbook Domino Circuits

We begin with a review of a simple form of domino circuits, including a motivation of why domino is beneficial, how pipelines can be constructed, and why such textbook pipelines have serious overhead.

Static CMOS gates are slow because an input must drive both NMOS and PMOS transistors. In any transition, either the pull-up or pull-down network is activated, meaning the input capacitance of the inactive network loads down the path. Moreover, PMOS transistors have poor mobility and must be sized larger to achieve comparable rising and falling delays, further increasing input capacitance. Dynamic gates overcome this weakness by eliminating the PMOS transistors and replacing them with a single precharge transistor. The dynamic gate is precharged high, then may evaluate low through an NMOS stack. Unfortunately, if one dynamic inverter directly drives another, a race can corrupt the result. When clk rises, both outputs have been precharged high. The high input to the first gate causes its output to fall, but the second gate's output also falls in response to its initial high input. The circuit therefore produces an incorrect result because the second output will never rise during evaluation. Domino circuits solve this problem by using inverting static gates between dynamic gates so that the input to each dynamic gate is initially low. The falling dynamic output and rising static output ripple through a chain of gates like a stream of toppling dominos. In summary, domino logic runs 1.5–2$\times$ faster than static CMOS logic [6] because dynamic gates present a much lower input capacitance for the same output current and have a lower switching threshold, and because the inverting static gate can be skewed to favor the critical monotonically rising evaluation edges.

After domino gates evaluate, they must be precharged before they can be used in the next cycle. If all domino gates were to precharge simultaneously, the circuit would waste time during which no useful computation occurs. Therefore, domino logic is conventionally divided into two phases, ping-ponged such

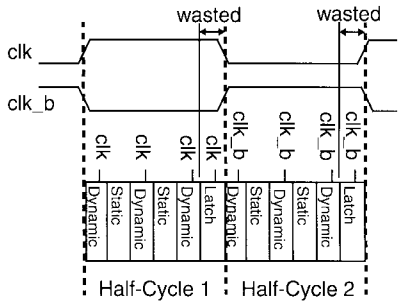



Fig. 1. Domino pipeline with ideal clocks.

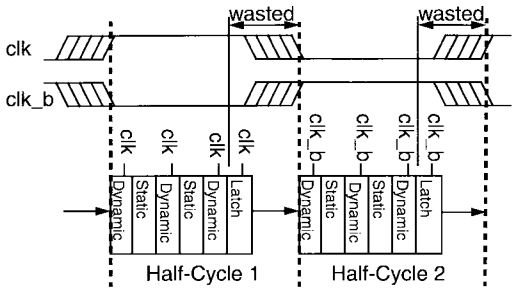



Fig. 2. Domino pipeline with clock skew.

that one precharges while the other evaluates, thus hiding the precharge time from the critical path. In textbook domino clocking schemes [7], latches are used between phases to sample and hold the result before it is lost to precharge, as illustrated in Fig. 1. Such schemes are analogous to two-phase static designs which consist of two blocks of logic separated by transparent latches. In all figures in this paper, logic is positioned along the horizontal axis underneath the clocks corresponding to when the gate nominally evaluates.

In a system with zero overhead, the cycle time is the sum of the delays through all logically useful gates in the longest path. With ideal clocks, textbook domino clocking comes close to achieving this goal, adding only the propagation delay of the latch in each half-cycle to the overall cycle time. With an average delay of 1.5 FO4 inverters for a typical latch and two latches per cycle, this amounts to three FO4 delays wasted for latching rather than used for logical work. Some designs partially alleviate this penalty by incorporating logic into the latch, but this requires a large cell library and even these designs pay some time penalty over a fully latchless design.

Unfortunately, a real pipeline shown in Fig. 2 has uncertainty in the arrival time of clocks which introduces more overhead. The uncertainty comes from phase-lock-loop jitter, mismatches in the clock distribution network, cross-die process variation, data-dependent clock loading, etc., and will be referred to as clock skew.[2] The light hashed lines indicate the range of possible skewed clocks. A half-cycle begins evaluation when the clock controlling the first dynamic gate rises. Evaluation must complete at least some setup time before the clock controlling the latch falls at the end of the half-cycle. These constraints on the start and end of evaluation create "hard edges" or "synchronization points" because the arrival of the clock determines the exact timing of the data. The first gate does not evaluate until the clock rises, and the last gate output must settle before the clock falls on the latch.

This evaluation time is nominally half of the total period $T$, but is reduced by any clock skew between the clock on the domino gate and the clock controlling the latch. Since worst-case clock skew is subtracted from both half-cycles, the actual time available for logic is $T - 2 * t_{\text{skew}}$. Clock skew is one to two FO4 delays for a well-designed clock distribution scheme such as a grid or H-tree in current $\sim$0.35-$\mu$m processes [1], [8], but may be four or more FO4 delays in a poorly balanced distribution scheme used in older systems [9]. As processes continue to scale, relative process tolerances will degrade while gate speeds increase, resulting in larger skews relative to intrinsic gate delays.

A third disadvantage of textbook domino circuits is that their hard edges prevent time borrowing. In other words, the logic must fit entirely within the evaluation period of the half-cycle and may not utilize extra time available in adjacent half-cycles. Since a discrete number of gates must be placed in each phase, there is generally wasted time left at the end of a half-cycle where a complete gate cannot fit. Moreover, the longest half-cycle sets the operating frequency of the machine. Therefore, a half-cycle which is longer than expected due to process variation or modeling inaccuracy cannot borrow time from adjacent, less critical half-cycles, but rather will degrade the operating frequency.

In summary, textbook domino circuits add latch delay, clock skew, and imbalanced logic delays to the cycle-limiting paths. Merely counting latch delays and skew budget indicates that at least three to five FO4 delays are wasted on latching and clock skew, depending on the effort devoted to controlling clock skew and incorporating logic into latches. The penalty for lack of time borrowing is more difficult to quantify, but is certainly significant.

## III. Relaxing Domino Clock Constraints

Static CMOS circuits suffer from similar penalties if one employs edge-triggered flip-flops which also impose hard edges initiating and terminating computation. Therefore, high-speed static designs generally soften edges by using transparent latches [10], [11] or pulsed latches [12]. The obvious advantage of latches is to allow time borrowing between stages, both to balance the pipeline and to compensate for delay uncertainties. A more subtle yet very important advantage is that reasonable amounts of clock skew can be eliminated from the cycle time by placing the latches such that data arrives at the inputs while the latch is transparent, even under worst case skew [13].

Some domino designers have recognized that this fundamental idea of softening the hard clock edges can be applied to domino circuits as well. To soften the rising edge, clocks should arrive early so evaluation occurs as soon as data arrives, even under worst-case clock skew. This requires that the inputs to the gate be reset low when the evaluation clock rises and only monotonically rise during evaluation so glitches will not corrupt the domino operation. Unfortunately, some dynamic

[2] The jitter component of skew is special because it does not affect hold times and phases generated off the same clock edges, but for simplicity we will conservatively lump it with other sources of clock uncertainty.

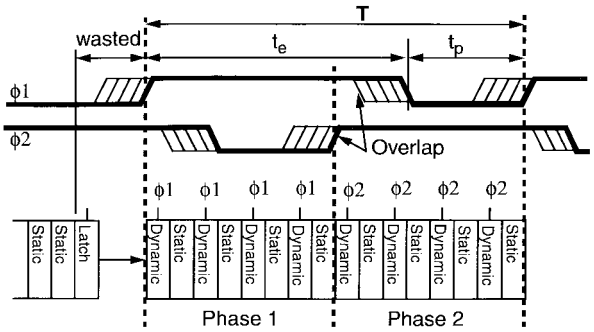



Fig. 3. Two-phase overlapping domino clocks.

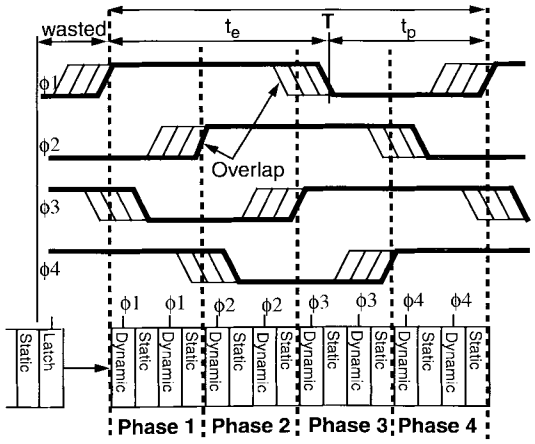



Fig. 4. Four-phase overlapping domino clocks.

gates like XOR require that both the true and complementary versions of the inputs be monotonically rising. Therefore, the complement cannot be obtained by passing the true signal through an inverter; instead, it too must come from a domino gate. Thus, designers commonly use domino gates which accept true and complementary monotonic inputs and produce true and complementary monotonic outputs. Such gates can be constructed with dynamic differential cascode voltage switch (DCVS) logic, also known as dual-rail domino.

To soften the falling edge, the result of the domino gate should be consumed well before precharge begins. This can be done by overlapping clocks such that the next phase begins evaluation before the previous phase precharges. Once the next phase has evaluated and consumed the result of the previous phase, the result is no longer needed and may fall low without impacting the next phase. Therefore, latches are not truly necessary between domino phases as long as the clocks overlap sufficiently.

Paths using two and four overlapping clock phases to soften clock constraints are illustrated in Figs. 3 and 4. The heavy lines indicate the latest clock timing used for conservative timing analysis, while the light hashed lines indicate the possible skew $t_{\rm skew}$ which might exist between any two clocked nodes. At the interface from static to domino logic, the static result must be stable before the earliest time that the domino gate might begin evaluation. Since the domino phase may actually begin evaluation late, time may be wasted between when static results are stable and the domino gate consumes them. Therefore, skew must be budgeted in the path at the interface. Once in the domino pipeline, all the sources of overhead from textbook domino are eliminated. Clock skew never impacts the critical path because the domino gates are guaranteed to be in evaluation by the time critical data arrives. Further, latches are removed from the critical path. Finally, logic can be balanced by borrowing modest amounts of time into the next phase.

While these ideas are well-known among some expert designers, they have often been viewed as proprietary design techniques and were not published. Self-timed systems [14]–[16] pioneered the concepts of "zero overhead" latchless domino clocking, but suffer from difficulties of control overhead, insufficient CAD infrastructure, and verification and testability challenges. Clock and data precharged dynamic (CDPD) circuits [17] use NORA gates and a slow precharge ripple to eliminate latches. Hewlett-Packard was one of the earliest companies to publish, describing a time-borrowing "pipeline latch" used in the PA7200 floating point multiplier to reduce skew sensitivity, especially that caused by duty cycle uncertainty [18]. Digital Equipment Corporation employs overlapping clock phases on the 21164 Alpha to eliminate a latch altogether from the critical adder self-bypass path, instead storing state on dynamic nodes [1], but has not generalized the technique for widespread use. Intel developed a systematic domino clocking scheme called Opportunistic Time Borrowing Domino that removes all three forms of overhead [19]. Engineers from most other microprocessor companies also have mentioned in private conversation the existence of proprietary domino clocking methodologies addressing the overhead.

In the next sections, we develop a systematic framework for designing and analyzing domino systems that use overlapping clocks to eliminate overhead. The framework is generically called skew-tolerant domino. We show how the framework can be used to understand and optimize the amount of skew and time borrowing tolerable under various clocking alternatives.

## IV. Skew-Tolerant Domino

To eliminate overhead from domino pipelines, we must arrange clocks such that domino gates are always ready for evaluation by the time critical inputs arrive and do not precharge until the next gate consumes the result. This section derives the fundamental constraints on precharge and evaluation times for a synchronous system and then solves them to maximize the allowable clock skew.

In general, we can consider $N$ overlapping clocks. The clock cycle of period $T$ is divided into $N$ phases. Each phase rises $T/N$ after the previous phase, and by symmetry all phases have the same duty cycle. Each phase is high for an evaluation period $t_e$ and low for a precharge period $t_p$ as illustrated in Figs. 3 and 4 for two- and four-phase systems. Evaluation time must be more than 50% of the cycle time in two-phase systems to create overlapping clocks, but may be $T/2$ for systems with three or more phases. Given these $N$ clocks, we can derive waveforms which maximize the tolerable skew in an $N$-phase system, independent of the actual logic contained in the phases.

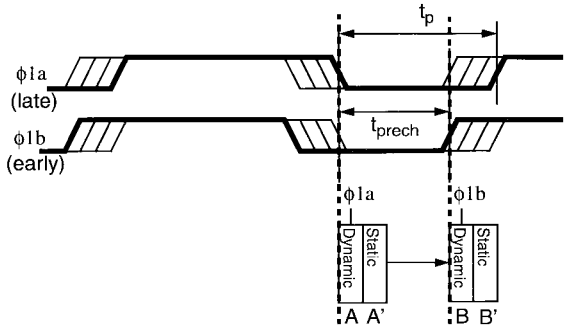



Fig. 5. Precharge time constraint.

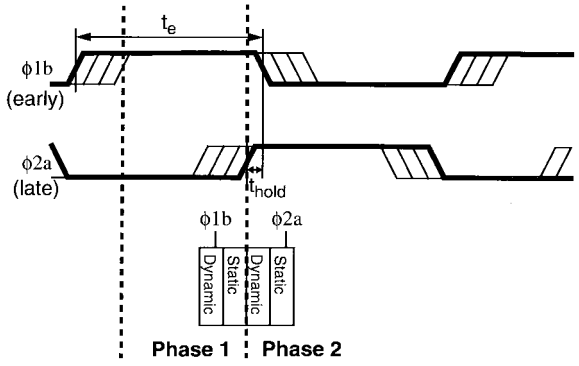



Fig. 6. Evaluation time constraint.

$t_p$ is limited by the rate at which a gate precharges, while $t_e$ is set by the required overlap between clock phases.

Fig. 5 illustrates the constraint on precharge time set by two consecutive domino gates in the same phase. $t_p$ is set by the requirement that dynamic gate $A$ must fully precharge, flip the subsequent static gate $A'$, and bring the static gate's output below $V_t$ by some noise margin before domino gate $B$ reenters evaluation so that the old result from $A'$ does not cause $B$ to incorrectly evaluate. We call the time required $t_{\rm prech}$ and design the cell library and maximum unbuffered wire length to guarantee a bound on this time. The worst case occurs when $\phi 1a$ is skewed late and $\phi 1b$ is skewed early, reducing the effective precharge window width by $t_{\rm skew}$. Therefore, we have a lower bound on $t_p$ to guarantee proper precharge

$$t_p \geq t_{\rm prech} + t_{\rm skew}. \tag{1}$$

Similarly, Fig. 6 illustrates the constraint on evaluation time set by the requirement that a phase remain in evaluation until the following phase consumes the data. The necessary overlap is called $t_{\rm hold}$ and will be later shown to generally be a small negative time. The minimum nominal overlap in the phases is then $(t_{\rm hold} + t_{\rm skew})$. Adding this overlap to the nominal time shift between phases of $T/N$ gives the evaluation time

$$t_e \geq \frac{T}{N} + t_{\rm skew} + t_{\rm hold}. \tag{2}$$

Combining the constraints on precharge and evaluation time (1) and (2), we find the relationship between cycle time and maximum allowable skew

$$t_{\text{skew-max}} = \frac{\frac{N-1}{N}T - t_{\rm hold} - t_{\rm prech}}{2}. \tag{3}$$

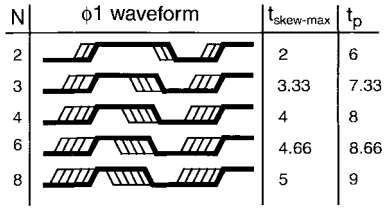



Fig. 7. Basic skew tolerance (in FO4 inverter delays).

For large $N$ and $T$, the maximum tolerable skew approaches $T/2$. Small $N$ reduces the tolerable skew because phases overlap less. The budget for precharge and hold time further reduces tolerable skew. Notice that if the actual skew between any two clocked gates $t_{\text{skew-actual}}$ exceeds $t_{\text{skew-max}}$, the pipeline may fail to operate at the target frequency, no matter how fast the gates within the pipeline evaluate. This is because the precharge window and hold time must be guaranteed, independent of logic evaluation delays.

To get a feeling for the tolerable skew, consider a fast system with a cycle time $T = 16$ and $t_{\rm prech} = 4$ FO4 delays.[3] $t_{\rm hold}$ is generally a small negative number because the first domino gate of the subsequent phase evaluates immediately after its rising clock edge while the precharge must ripple through both the last dynamic gate and the following static gate (which is generally sized to favor the rising edge, not the falling precharge edge). Hold time can therefore conservatively be approximated as zero provided minimum and maximum fanout guidelines are followed. This hold time is required for capturing data before precharge and is distinct from race-through issues which will be discussed in Section VII. The clock waveforms, along with the corresponding $t_{\text{skew-max}}$ and $t_p$, are illustrated in Fig. 7. The sweet spots appear to be $N = 2$ and $N = 4$ because they are good tradeoffs between tolerable clock skew and ease of clock generation. A two-phase system tolerates a modest amount of skew. In this example, a four-phase system doubles the skew tolerance and uses a 50% duty cycle at the expense of additional complexity in generating the phases.

Notice that the designer must still budget $t_{\rm skew}$ at the interface between static and domino logic because the static output must be stable before evaluation could begin, yet the domino clock may arrive late. This motivates building critical loops entirely from domino to avoid the skew penalty. Since most such loops contain recombinant logic with differing numbers of inversions, dual-rail circuits are usually needed to avoid the recombinations of inversions of single-ended signals.

As we will see later, skew-tolerant domino paths should include at least one gate per phase to avoid race-through problems. For example, a ph1 domino gate may not directly

[3] Selecting the precharge time involves a careful balancing act between long precharge times which reduce $t_{\text{skew-max}}$ and short precharge times which require large precharge devices and lower P/N ratios on static gates, slowing the evaluation path. For a cycle time of 16 FO4, a precharge time of four FO4 delays is a reasonable lower bound.

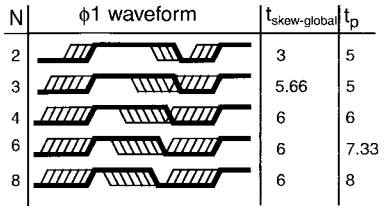



Fig. 8. Global skew tolerance (in FO4 inverter delays).

drive a ph3 domino gate without at least passing through a ph2 buffer. In the special case of exactly one domino gate per phase, interesting for very high frequency designs, the precharge constraint of (1) can be relaxed. In this case, the output of the static gate must only fall low by the time the next phase, rather than the current phase, reenters evaluation because there are no other gates in the current phase to corrupt. Therefore, the available precharge time effectively increases by $T/N$ and (3) becomes

$$t_{\text{skew-max}} = \frac{T - t_{\text{hold}} - t_{\text{prech}}}{2}. \tag{4}$$

## V. Global and Local Skew

The skew-tolerant domino pipeline presented in the previous section allows a moderate amount of skew between any two points of a chip. In real systems, however, we know that the skew between nearby elements, $t_{\text{skew-local}}$, may be much more tightly bounded than the skew between arbitrary elements, $t_{\text{skew-global}}$. We take advantage of this fact to increase the tolerable global skew. We therefore partition the chip into multiple regions, called local clock domains, which have at most $t_{\text{skew-local}}$ between clocks within the domain.

Assume that all connected blocks of logic in a phase are constructed within local clock domains (this is a reasonable restriction, especially for large $N$ and short phases, but will be relaxed later for exceptional cases). Therefore, the precharge constraint (1) becomes dependent on local skew because precharge only must complete before the next gate in the same phase resumes evaluation, while the evaluation constraint (2) remains dependent on global skew because clocks from different domains must still overlap. These constraints may be solved for maximum tolerable global skew

$$t_{\text{skew-max-global}} = \frac{N-1}{N}\,T - t_{\text{hold}} - t_{\text{prech}} - t_{\text{skew-local}}. \tag{5}$$

In the event that local skew is tightly bounded, a second constraint is placed on the precharge period to guarantee that the last gate in a phase precharges before the first gate in the next phase begins evaluation extremely early under huge global skew. The analysis of this case is straightforward but is omitted because typical chips should not experience such large global skews.

Fig. 8 summarizes the improved results of global skew assuming local skew can be bounded at one FO4 delay and that no clock domain crossings exist. Observe that the tolerable global skew saturates at six FO4 delays for large $N$ because of the constraint mentioned in the last paragraph.

In exceptional cases, critical paths and floorplanning constraints may prevent a phase of logic from being entirely within a local domain. If we do not handle such cases specially, we could not take advantage of local skew to increase tolerable global skew. A possible workaround is to locally introduce an additional phase at the domain crossing delayed by an amount less than $T/N$.

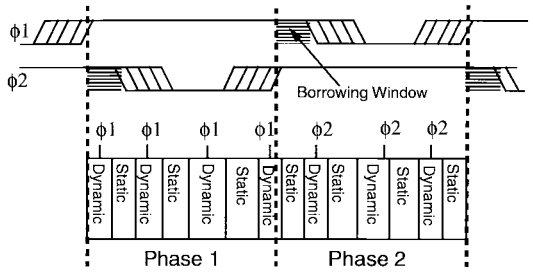



Fig. 9. Time borrowing.

## VI. Time Borrowing

In systems with a good clock distribution network which take advantage of local skew domains, the overlap between phases may greatly exceed the actual global skew. The excess overlap can be used for time borrowing, as illustrated for $N = 2$ in Fig. 9. The nominal logic delay through each stage is $T/N$. If logic completes earlier than $T/N$ after the start of the phase, time may be wasted waiting for the next phase to begin evaluation. If logic takes much more than $T/N$, it will not complete before precharge begins and the clock frequency will have to be reduced. However, there is a period of time from $T/N$ to $T/N + t_{\text{borrow}}$ during which the logic may complete without adverse effect. This period of time is useful because it helps balance the pipeline. For example, consider the last $\phi 1$ domino gate. The borrowing window allows the gate's evaluation to extend into the next phase. In textbook domino, there is no borrowing window and only the first three gates would fit in the first phase; the remaining time would be wasted. Time borrowing may continue across many phases. For example, if the first phase evaluates for $T/N + t_{\text{borrow}}$, the second phase may evaluate for a time between $T/N - t_{\text{borrow}}$ and $T/N$.

From Fig. 9, it is clear that the width of the borrowing window is the minimum guaranteed overlap between phases, i.e., the tolerable global skew from (5) minus the actual global skew

$$t_{\text{borrow}} = \frac{N-1}{N}\,T - t_{\text{hold}} - t_{\text{prech}} - t_{\text{skew-local}} - t_{\text{skew-global}}. \tag{6}$$

Assuming an actual maximum global skew of two, local skew of one, precharge time of four, hold time of zero, and cycle time of 16 FO4 delays, we illustrate the time borrowing window for various $N$ in Fig. 10. The clock waveforms are identical for all cases because their duty cycle is set by the local skew, not the number of phases. However, the portion of $t_e$ nominally used for evaluation decreases as $N$ increases, leaving more time available for borrowing into the next phase. A two-phase system offers a small amount of time borrowing, which makes balancing the pipeline somewhat easier. $N = 4$

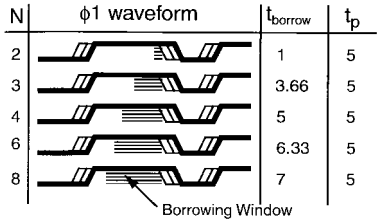



Fig. 10. Time borrowing availability (in FO4 inverter delays).

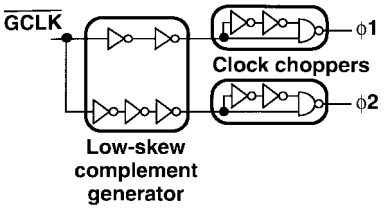



Fig. 11. Two-phase clock generation.

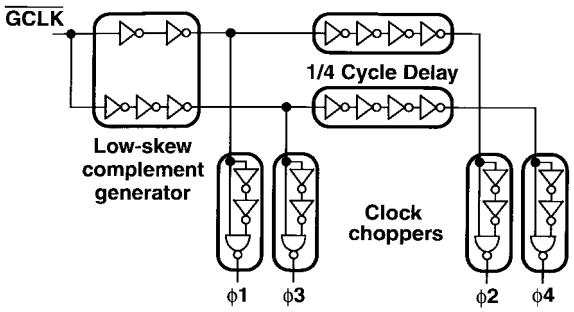



Fig. 12. Four-phase clock generation.

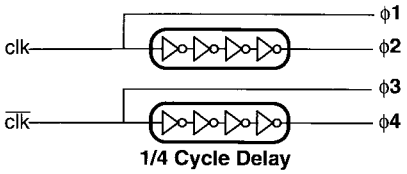



Fig. 13. Simplified four-phase clock generation.

offers more than a full phase of time borrowing, granting the designer tremendous flexibility.

Time borrowing is also useful because it automatically helps compensate for process and operating condition variations across the die and for inaccuracies in the modeling and simulation of the critical paths. These uncertainties are expected to increase as feature size continues to drop, so this time borrowing characteristic is especially useful. In a system such as conventional domino that does not allow time borrowing, the longest half-cycle directly sets cycle time. In skew-tolerant domino, a phase which is longer than anticipated may borrow time from an adjacent phase which is of nominal, or better yet, shorter, length, resulting in an averaging effect over many phases [19]. Since the actual skew between gates on any particular path is typically smaller than the worst-case skew, this additional margin is available for such "opportunistic" time borrowing.

## VII. Clock Generation

We have analyzed the benefits of overlapped domino clocks, but the advantages would be moot if generating the required clocks were impractical. In this section, we describe simple ways to generate two, four, and $N$-phase clocks, analyze the clock skew introduced by the clock generators, and consider race-through problems from overlapping clocks. Remember that the key principle is to produce overlapping clocks. Thus, buffer delay methods presented in this section are only a few of many workable schemes that include self-timing, matched delays, and closed-loop oscillators.

### *A. Clock Phase Generation*

In most high frequency systems, a single clock is distributed globally using a modified H-tree or grid to minimize skew [10], [20]. Skew tolerant domino can use this same clock distribution scheme with a single global clock. Within each unit or functional block, local clock generators produce the multiple phases required for latches and skew-tolerant domino.

Two-phase clock generation is shown in Fig. 11. From the global clock gclk, true and complementary clocks must be locally generated in a low-skew manner. Shoji [21] showed that the paths containing two and three inverters may be sized to have almost no process-dependent skew. However, the output slopes are not equal. Another popular technique tries to match the delay of two inverters against the delay of one inverter plus a transmission gate. Once the complementary clocks are produced, the falling edges can be delayed. Such delay circuits, called clock choppers, are widely used in microprocessors to solve setup or hold time violations.

Four-phase clock generation shown in Fig. 12 is nearly identical to two-phase generation, but delays $\phi 2$ and $\phi 4$ by a quarter cycle using inverter chains. Clock choppers can be used to produce duty cycles greater than 50%, or may be omitted to allow more precharge time while maintaining moderate amounts of skew tolerance. A simplified version of the four-phase generator is shown in Fig. 13. The simplified design omits the clock chopper and derives the four domino phases from clk and its complement, 50% duty cycle clocks used by static latches. Although four-phase clocking has not received much attention recently, the idea has been considered by MOS designers for decades [22].

In general, $N$ phases can be produced by delaying the clock or the complement of the clock with buffer chains. Many designers already produce such delayed phases to stagger the precharge, allowing removal of the series evaluation transistor from gates with delayed clocks [11]. As long as designers are careful to always include domino gates clocked by the delayed phases, they can remove latches and automatically enjoy the benefits of skew tolerance and time borrowing.

How local should the local phase generators be? To keep distribution easy, the generators should serve a small enough region that skew from wiring is low. This typically implies that a local phase generator serves a radius of less than 2 mm. Just as clocks can be enabled on a very fine granularity for power savings, the local phase generators may serve an even

smaller load such as $N$ gates in an $N$-bit datapath. This scheme might seem to resemble self-timing with a delay matched to the logic. The important difference is that the clock phases are designed independently of the logic they serve and logic is clocked by the most suitable phase. Therefore, the designer's task is simplified because each local phase generator does not have to be individually tuned to logic. Moreover, as we will see in the next part of this section, the designer can simply budget a worst-case skew introduced by the phase generators, rather than simulating how well each matched delay tracks a particular piece of logic over process and environmental variation. The drawback is that this worst-case skew budget will be larger than delay mismatches in custom-designed self-timed systems, but this does not impact performance if sufficient skew tolerance is available.

### B. Phase Generator Skew

Using delay chains immediately raises concerns about variations in their delay caused by temperature, voltage, transistor orientation, and processing. While the absolute variation in delay can be very large, a chip is only sensitive to relative, not absolute, variations in delays. What is critical is how well the delays in the clock generator track the critical paths of the chip. We can model this relative variation by introducing additional skew caused by the local clock generator. This skew is some fraction of the overall delay through the generator; this fact implies that it is important to minimize the absolute delay of the generator because a fraction of that delay will appear as extra skew.

There are three fundamental timing constraints which are affected by local clock generator skew; two affect cycle time and one affects functionality. The first constraint is that sufficient time must be available for precharge, even under worst-case skew. This condition sets the maximum clock duty cycle; if the condition is violated, the clock must be slowed to reduce the duty cycle. The second constraint is that sufficient overlap must exist between phases. If this condition is violated, the chip may not work at any frequency. The third constraint is that sufficient extra overlap must exist for the required time borrowing. If this condition is violated, the clock must be slowed to reduce the need for borrowing. We can analyze these constraints to determine the impact of phase generators on tolerable clock skew and time borrowing.

The precharge constraint is satisfied by setting the clock duty cycle for the worst-case operating conditions in the target process corner. We will refer to this environment and processing as the "design corner." If precharge works in the design corner, it will also work in all other environmental corners because both the clock chopper and precharge will get faster, allowing more precharge time for a less-critical precharge. In the event of slower processing on PMOS devices, the clock frequency must be reduced to accommodate precharge.

Given such a design, it is vital that all clocks overlap by at least $t_{\mathrm{hold}}$ so the chip will function. Clock skews are caused by global clock jitter and duty cycle variation, by mismatches in wire length and coupling, and by buffer delay variation from voltage, temperature, and processing differences across the chip. The overlap constraint is usually hardest to satisfy in the corner with fast gates and slow wires where the clock choppers produce minimum overlap, yet wire-induced skew is maximum. Additional overlap must be provided in the design corner to guarantee sufficient overlap in this worst-case relative skew corner. This overlap appears as time available for borrowing in the design corner. Since a moderate amount of time borrowing is important for balancing logic anyway and is therefore provided in the design corner, guaranteeing overlap is usually not an issue.

Finally, we must examine how the budget for time borrowing is affected. Time borrowing margins are important when the logic runs slowly and needs to borrow time, yet the local clock generators run faster and provide less time for borrowing. Since the local generators are not replicas of the circuits they are tracking, and indeed are static gates tracking the speed of dynamic paths, their relative delays may vary over process corners as well as due to local variation in voltage and temperature and local process variations. Simulation finds that when most of the chip is operating in the design corner but a local clock generator sees a temperature 30° lower and supply voltage 300 mV higher, the local generator will run 13% faster than nominal (6% from temperature, 7% from voltage). The relative delay of simple domino gates with respect to fanout-of-four inverters varied up to about 6% across process corners. Finally, process tilt, i.e., fluctuation in $L_e$, $t_{\mathrm{ox}}$, etc., across the die, may speed the local clock generator more than nearby logic. Little data is available on process tilt, but if we guess it causes similar 13% variation, we conclude that nearly a third of the total local clock generator delay should be subtracted from the time borrowing budget.

These estimates are conservative because the local clock generators are physically close to the logic which may need to borrow time and therefore track better than might be feared. Moreover, since only a small fraction of paths depend on significant time borrowing, it is less likely that these paths will see the theoretical worst-case skews. Since four-phase clocking adds a quarter cycle of margin less the relative variation in the quarter cycle delay chain, four-phase skew-tolerant domino always has at least one sixth of a cycle more margin for skew and time borrowing than two-phase systems.

An alternative to delay chains would be local delay-locked loops which precisely generate local clocks from a globally distributed reference. In addition to the area penalty, this scheme adds loop-to-loop jitter to the global skew budget. Since such jitter is difficult to control, the overall performance may be no better.

### C. Race-Through

Like latch or flop-based designs, skew-tolerant domino is subject to race-through (also known as min-delay) failure. Race-through is exacerbated by the overlapping clocks and by the large amounts of skew which a skew-tolerant system might encounter. Since there is at least one gate in each phase, race-through only occurs if all $N$ phases are simultaneously high for at least the contamination delay $t_{\mathrm{cd}}$ through one of the phases. Since the nominal nonoverlap between the first phase

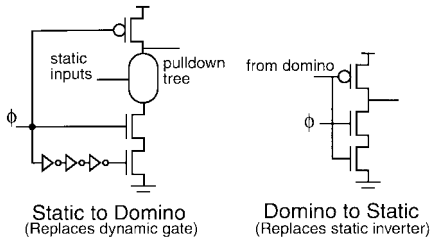



Fig. 14. Domino/static interfaces.

falling and the last phase rising is $T(N-1)/N - t_e$ and an actual overlap of $t_{\rm cd}$ can be tolerated, the maximum global skew still guaranteeing that not all phases are simultaneously high too long is

$$t_{\text{skew-global}} < \frac{N-1}{N} T - t_e + t_{\rm cd}. \tag{7}$$

Reasonable control over skew in four-phase systems avoids race-through. For example, four-phase systems with 50% duty cycle clocks can tolerate a quarter cycle of skew before race-through becomes a possibility. In two-phase skew-tolerant domino, extra nonoverlapping clocks may be used on the first domino gate in each phase to solve race-through problems [19]. If a three-phase system is used, race-through limits the duty cycle.

When paths are very short or a large number of phases are used, the rule of at least one gate per phase may be violated. If the path is short, the logic is noncritical and static logic with latches is probably a better design choice anyway. If a very large number of phases are used, certain phases may safely contain no logic.

### D. Clock Edge Rates

For simplicity, the analysis so far has neglected the finite slope of clock edges. The primary effect of slower clock edges is to increase $t_{\rm prech}$ and $t_{\rm hold}$, just as they increase setup time in latch-based systems. The appropriate values can be determined by simulating precharge and hold times with slowest clock edges. Another effect of slow clock edges is that the delay through a domino gate depends on both the clock and data inputs. A thorough timing analyzer should take both inputs into account. Circuits will operate fastest when sufficient time is borrowed into each phase that the clock has fully risen before the data arrives.

## VIII. Pipelining Issues

Domino circuits do not operate in isolation. In this section, we examine the interface from static to domino circuits, from domino to static circuits, and between skew-tolerant and self-timed dynamic gates.

As mentioned earlier, clock skew must be budgeted in paths crossing from static to domino to guarantee that the static result is stable before the first domino gate evaluates. Furthermore, the static result must remain stable as long as the domino gate is in evaluation. This can be guaranteed by latching the result with a transparent latch at the end of the static logic. Alternatively, the dynamic gate pulldown stack can be pulsed briefly to effectively integrate a pulsed latch into the domino [12] as shown in Fig. 14. Such a scheme saves the latency and area of the latch at the expense of introducing a min-delay constraint that the static result must remain stable until the end of the pulse.

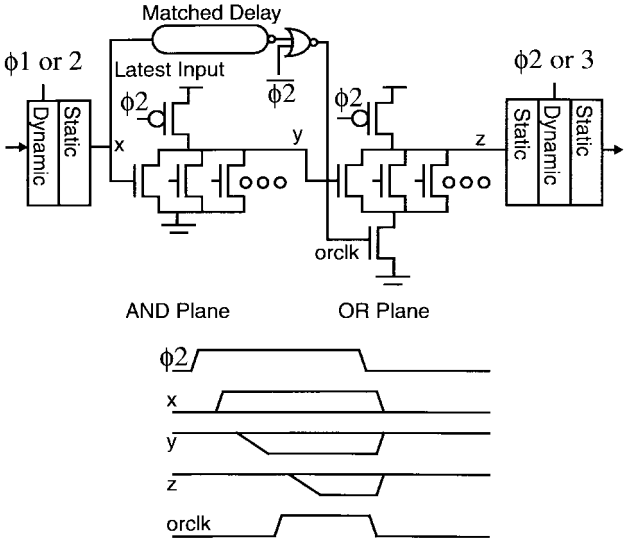



Fig. 15. Self-timed PLA in skew-tolerant pipeline.

When domino gates drive a block of static gates, the dynamic outputs should first be latched so that precharge does not send glitches through the static logic. The latch shown in Fig. 14 is a good choice because it is very fast and permits time-borrowing even as the clock falls. It may be controlled with the same clock as the dynamic gate it follows to eliminate skew. Weak cross-coupled inverters may be placed on the output node to support stop-clock operation and logic may be incorporated into the latch if desired. When dual-rail inputs are available, cross-coupling can improve noise immunity [5].

Certain very useful dynamic structures such as wide comparators and dynamic programmable logic arrays (PLA's) are inherently nonmonotonic and are conventionally built for high performance using self-timed clocks to signal completion. The self-timed clocks can be combined with skew-tolerant domino by locally producing a self-timed completion signal to clock the gate after nonmonotonic inputs have settled sufficiently. For example, Fig. 15 shows a dynamic NOR–NOR PLA integrated into a skew-tolerant pipeline. The AND plane is illustrated evaluating during $\phi 2$ and adjacent logic can evaluate in the same or nearby phases. The latest input $x$ to the AND plane is used by a dummy row to produce a self-timed clock *orclk* for the OR plane that rises after AND plane output $y$ has settled. Notice how the falling edge of *orclk* is not delayed so that when $y$ precharges high the OR plane will not be corrupted. The output $z$ of the OR plane is then indistinguishable from any other dynamic output and can be used in subsequent skew-tolerant domino logic.

## IX. Simulation Results

To evaluate the performance benefits of skew-tolerant domino in the context of high-speed microprocessors, we

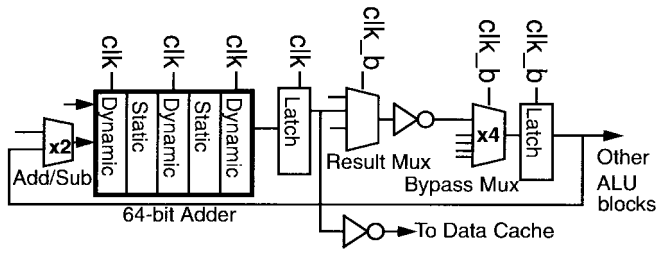



Fig. 16. ALU self-bypass path (textbook domino).

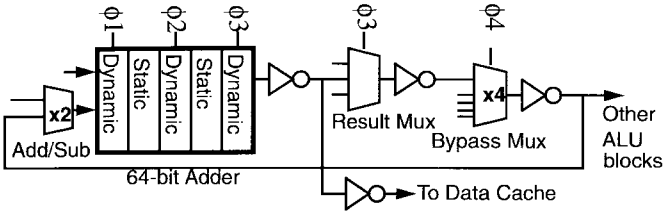



Fig. 17. ALU self-bypass path (skew-tolerant domino).

compared two 64-b adder self-bypass paths, one constructed using textbook domino with latches and the other using four-phase skew-tolerant domino, as shown in Figs. 16 and 17. The paths were simulated in the HP14 0.6-$\mu$m three-metal process with an FO4 delay of 138 ps under nominal process parameters at 3.3 V, 25°C. We assume a microarchitecture and floorplan similar to the dual integer arithmetic logic units (ALU's) of the DEC Alpha 21 164 [10]. The adder involves two levels of carry selection implemented with dual-rail domino logic. We assumed contacted diffusion parasitics on each transistor source/drain and worst-case capacitance on long signal lines, but did not model smaller wire parasitics.

As shown in Fig. 18, the textbook path has a latency of 13.0 FO4 delays (1.80 ns), but a cycle time of 16.6 FO4 delays because the first half-cycle has more logic than the second. This cycle time bloating is a common problem in ALU design and is often solved in practice either by moving the latch in to the middle of the adder path or stretching the first clock phase. The former choice is costly because the bisection width of the circuit is greater within a carry-select adder so more latches are required. The later choice is an *ad hoc* solution with an effect similar to that systematically achieved with skew-tolerant domino.

The skew-tolerant path improves the latency because latches are replaced with fast inverters. Cycle time equals latency because a modest amount of time borrowing is used to balance the pipeline. The skew-tolerant waveforms are designed with $t_p = 5.2$ FO4 delays and $t_e = 6.7$ in order to accommodate $t_{\text{prech}} = 4.2$ and $t_{\text{skew-local}} = 1$. According to (6), $t_{\text{skew-global}} + t_{\text{borrow}} = 3.7$. In the actual circuit, we observed a global skew tolerance of 2.5 FO4 delays because some of the overlap was used for intentional time borrowing.

When a skew of one FO4 delay is introduced, the textbook latency increases to 15.0 FO4 delays because skew must be budgeted in both half-cycles. The skew-tolerant latency and cycle time are unaffected. Overall, the skew-tolerant design is at least 25% faster than the textbook design, achieving 600-MHz simulated operation.

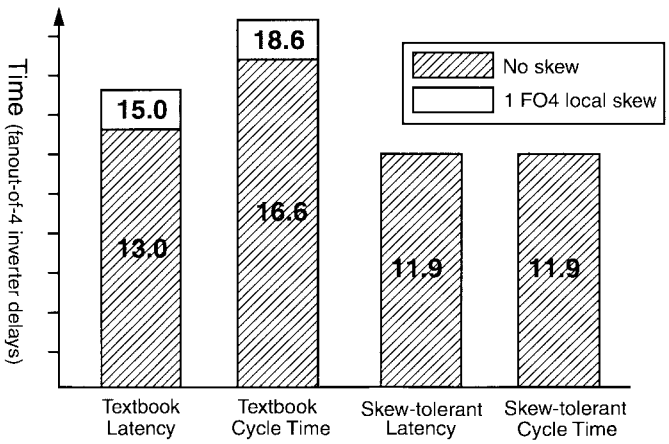



Fig. 18. ALU performance simulation results.

## X. Summary

Domino circuits are increasingly popular because they offer a significant performance boost over static gates. We have seen that textbook domino clocking methods on high-frequency chips lose much of their benefit to clocking overhead. Designers have realized that overlapping domino clocks can be used to guarantee that data ripples through domino gates as soon as it arrives, even if clocks are skewed. Eliminating latches reduces the latency of the critical path and allows time borrowing. As cycle time decreases relative to intrinsic gate speed and process variations become larger, time borrowing becomes crucial to balance the pipeline and average delay variations across longer paths. In summary, overlapping domino clocks provides many of the performance advantages of self-timed logic while maintaining the simplicity of synchronous design.

Most of the overlapping domino clock techniques used so far have been *ad hoc* or proprietary. We provide a framework for understanding essential domino timing constraints and a systematic methodology for taking advantage of the ideas. Two-phase skew-tolerant domino is similar to textbook domino design, yet offers modest amounts of skew immunity. Four-phase skew-tolerant domino requires additional locally generated clocks but allows large amounts of skew and time borrowing. Even more phases can be easily generated by delaying the precharge signals to domino gates. We therefore recommend four-phase or systematically delayed precharge skew-tolerant domino for high-speed chips. Simulations confirm that skew-tolerant domino improves the performance of an ALU self-bypass path in a fast superscalar microprocessor by at least 25% over a textbook domino design.

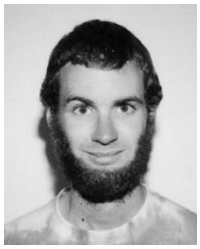
## Acknowledgment

The authors thank I. McClatchie, H. Partovi, and T. Williams for many suggestions and improvements.

## References

[1] P. Gronowski and B. Bowhill, "Dynamic logic and latches—Part II," in *Proc. VLSI Circuits Workshop (VLSI Symp.)*, June 1996.
[2] R. H. Krambeck, C. M. Lee, and H. Law, "High-speed compact circuits with CMOS," *IEEE J. Solid-State Circuits*, vol. SC-17, pp. 614–619, June 1982.

[3] P. Gronowski *et al.*, "A 433-MHz 64-b quad-issue RISC microprocessor," *IEEE J. Solid-State Circuits*, vol. 31, pp. 1687–1696, Nov. 1996.
[4] N. Vasseghi *et al.*, "200 MHz superscalar RISC processor," *IEEE J. Solid-State Circuits*, vol. 31, pp. 1675–1686, Nov. 1996.
[5] C. Heikes and G. Colon-Bonet, "A dual floating point coprocessor with an FMAC architecture," in *ISSCC Dig. Tech. Papers*, Feb. 1996, pp. 354–355.
[6] D. Harris, S. Oberman, and M. Horowitz, "SRT division architectures and implementations," in *Proc. 13th IEEE Symp. Computer Arithmetic*, July 1997, pp. 18–25.
[7] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design.* Reading, MA: Addison-Wesley, 1993, p. 351.
[8] J. Lotz *et al.*, "A quad-issue out-of-order RISC CPU," in *ISSCC Dig. Tech. Papers*, Feb. 1996, pp. 210–211.
[9] M. Shoji, "Electrical design of BELLMAC-32A microprocessor," in *Proc. IEEE Int. Conf. Circuits and Computers*, Sept. 1982, pp. 112–115.
[10] W. Bowhill *et al.*, "A 300 MHz 64b quad-issue CMOS RISC microprocessor," in *ISSCC Dig. Tech. Papers*, Feb. 1995, pp. 182–183.
[11] R. Colwell and R. Steck, "A 0.6 $\mu$m BiCMOS processor with dynamic execution," in *ISSCC Dig. Tech. Papers*, Feb. 1995, pp. 176–177.
[12] H. Partovi *et al.*, "Flow-through latch and edge-triggered flip-flop hybrid elements," *ISSCC Dig. Tech. Papers*, pp. 138–139, Feb. 1996.
[13] S. Unger and C. Tan, "Clocking schemes for high-speed digital systems," *IEEE Trans. Comput.*, vol. C-35, pp. 880–895, Oct. 1986.
[14] T. Williams, "Self-timed rings and their application to division," Ph.D. dissertation, Stanford University, Stanford, CA, May 1991.
[15] ——, "Performance of iterative computations in self-timed rings," *J. VLSI Signal Processing*, no. 7, pp. 17–31, Feb. 1994.
[16] T. Williams and M. Horowitz, "A zero-overhead self-timed 160-ns 54-b CMOS divider," *IEEE J. Solid-State Circuits*, vol. 26, pp. 1651–1661, Nov. 1991.
[17] J. R. Yuan, C. Svensson, and P. Larsson, "New domino logic precharged by clock and data," *Electron. Lett.*, vol. 29, no. 25, pp. 2188–2189, Dec. 1993.
[18] C. Heikes, "A 4.5 mm$^2$ multiplier array for a 200MFLOP pipelined coprocessor," in *ISSCC Dig. Tech. Papers*, Feb. 1994, pp. 290–291.
[19] Intel Corporation, "Opportunistic time-borrowing domino logic," U.S. patent 5 517 136, May 14, 1996.
[20] E. Cohen *et al.*, "A 533MHz BiCMOS superscalar microprocessor," in *ISSCC Dig. Tech. Papers*, Feb. 1997, pp. 164–165.
[21] M. Shoji, *High-Performance CMOS Circuits.* Englewood Cliffs, NJ: Prentice Hall, 1988.
[22] W. M. Penny and L. Lau, *MOS Integrated Circuits: Theory, Fabrication, Design and Systems Applications of MOS LSI.* New York: Van Nostrand, Reinhold, 1973, ch. 5.



**David Harris** (S'94) received the S.B. degree in mathematics and S.B. and M.Eng. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 1994. He is currently a graduate student in the Electrical Engineering Department of Stanford University, Stanford, CA.

He has done microprocessor design and circuit research at Intel and at Sun Microsystems. His research and teaching interests include high-speed CMOS circuit and execution unit design. He also enjoys teaching VLSI design to undergraduates and high school students.

**Mark A. Horowitz** (S'77–M'78–SM'95), for a photograph and biography, see p. 690 of the May 1997 issue of this JOURNAL.