The Future of Wires

RON HO, MEMBER, IEEE, KENNETH W. MAI, STUDENT MEMBER, IEEE, AND
MARK A. HOROWITZ, FELLOW, IEEE

Invited Paper

Concern about the performance of wires in scaled technologies has led to research exploring other communication methods. This paper examines wire and gate delays as technologies migrate from 0.18-μm to 0.035-μm feature sizes to better understand the magnitude of the wiring problem. Wires that shorten in length as technologies scale have delays that either track gate delays or grow slowly relative to gate delays. This result is good news since these "local" wires dominate chip wiring. Despite this scaling of local wire performance, computer-aided design (CAD) tools must still become more sophisticated in dealing with these wires. Under scaling, the total number of wires grows exponentially, so CAD tools will need to handle an ever-growing percentage of all the wires in order to keep designer workloads constant. Global wires present a more serious problem to designers. These are wires that do not scale in length since they communicate signals across the chip. The delay of these wires will remain constant if repeaters are used, meaning that relative to gate delays, their delays scale upwards. These increased delays for global communication will drive architectures toward modular designs with explicit global latency mechanisms.

Keywords**—***Capacitance, delay estimation, electromagnetic coupling, inductance, interconnections, resistance, technology forecasting, wire.*

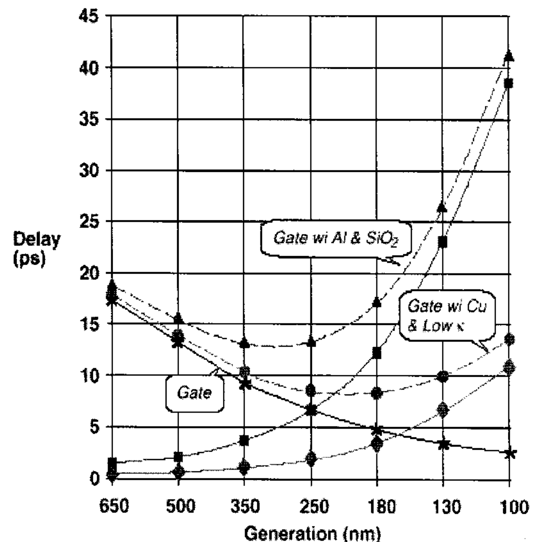


Fig. 1. Gate and wire scaling, from 1997 roadmap.

I. INTRODUCTION

At first glance, the future of wires in integrated circuit technologies appears grim. Even optimistic projections with copper technologies and low-κ dielectrics show that the delay through a fixed-length wire increases as the base fabrication technology scales to smaller dimensions. Since gate delays decrease under scaling, we see an ever-increasing disparity between wire and gate delays. The popular graph shown in Fig. 1, taken from the 1997 SIA roadmap [1], illustrates this wire problem. The growing gap between the wire and the gate performance trajectories has motivated a number of papers that predict the demise of conventional wires or wiring methodologies and that call for new interconnection methods.

However, this plot can be somewhat misleading. For example, the "gate" delays shown are for unloaded single transistors (and thus claim 5-pS delays in a 0.18-μm technology), not for real logical devices. Also, the wire delays shown are for fixed lengths, but as technologies scale, most wires shrink in length. To help understand the real issues with wire scaling, this paper presents a set of performance metrics for wires and gates and then explores how these metrics change with scaling. The results will show that there is indeed a wire problem, although one not as simple as that implied by the SIA plot. As designs scale to newer technologies, they get smaller and their wires get shorter, and the relative change in the speed of wires to the speed of gates is modest. Depending on the assumptions used for transistor and wire performance, the delay ratio is close to unity, and scaled designs should continue to improve with technologies. This reasoning has led some researchers to claim that there is no wire problem [2].

Manuscript received July 25, 2000; revised December 15, 2000. This work was supported in part by the Defense Advanced Research Projects Agency under Contract MDA904-98-R-S855 and by the MARCO Interconnect Focus Center.

The authors are with the Computer Systems Laboratory, Stanford University, Stanford, CA 94305 USA.

Publisher Item Identifier S 0018-9219(01)03200-5.

0018–9219/01$10.00 © 2001 IEEE

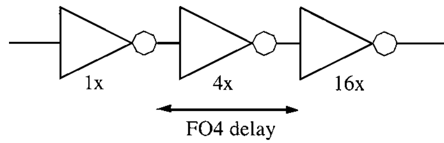


Fig. 2. A fanout-of-four inverter delay.

But this positive scenario is for wires that span a fixed number of gate pitches: as technologies scale, these wires get shorter. The real wire problem arises with increasing chip complexity and global communication costs. First, as technologies scale, designers can pack more and more modules on a chip. Each of these modules has manageable wire problems, and these problems grow slowly, if at all, with technology scaling. Yet, as the number of modules per chip grows exponentially, the accumulation of wire problems will quickly become unmanageable, unless the number of problems per module decreases. Second, some wires will not scale in length, and global communication delays over these wires will indeed increase, though perhaps not as dramatically as in the SIA plot. Even this slower delay growth can present problems, since wire performance, relative to gates, will continue to worsen. While designers need to account for this multicycle chip-length wire delay, these wires will not limit the cycle times of future chips. In fact, compared to the board-level interconnects that they replace, these on-chip global wires are still quite fast. To understand these conclusions better, we start by looking at performance metrics for gates and wires.

II. Metrics for Gates and Wires

Wires affect a circuit in three ways: wire capacitance adds load to driving gates; wire resistance, capacitance, and inductance all add signal delay; and inductive and capacitive coupling between wires adds signal noise. The significance of these effects depends on gate characteristics, since only if wire delays change relative to gate delays, or if signal noise changes relative to gate noise margins, will designers need to change the way they look at wire design. This section starts by characterizing gates, and then moves to metrics for wires. Once we have the basic parameters, we use them to evaluate noise coupling issues in Section II-B3 and overall performance in Section II-D.

A. Gate Metrics

Because transistors are very complicated devices, we want a simpler set of metrics to use in this study. Designers use transistors in a very limited set of topologies; static and dynamic CMOS gates dominate digital designs, so metrics that characterize these gates will suffice. As a measure of gate delay, we use the delay through an inverter driving four identical copies of itself, shown in Fig. 2. Since this gate has a capacitive fanout of four, we will call this delay a "fanout-of-four inverter delay," or simply an *FO4*. In a 0.18-μm technology, an FO4 is about 90 pS under worst case environmental conditions (high temperature and low V_{dd}).

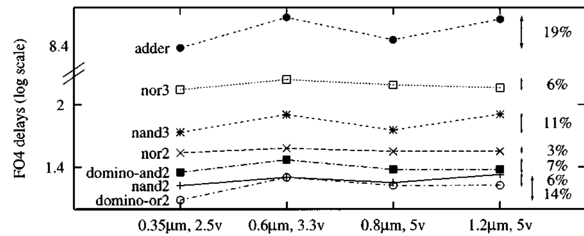


Fig. 3. Gate delays, normalized to FO4s.

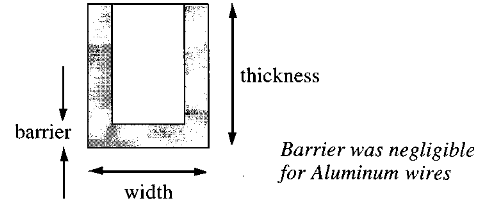


Fig. 4. Calculating resistance.

The utility of FO4 metrics is that any combinational delay, composed of many different static and dynamic CMOS gate delays, can be divided by an FO4, and this normalized delay holds constant over a wide range of process technologies, temperatures, and voltages. Fig. 3 shows the delay of different CMOS circuits over a number of recent process technologies. Thus, to understand how gate delays will scale, we need only estimate how the delay of a loaded inverter will scale, a much simpler task.

B. Wire Metrics

Wires have three important electrical characteristics: resistance, capacitance, and inductance. For the forseeable future, their delay and noise behavior, including transmission-line effects, can be well modeled using these three characteristics. All three depend on the wire's geometry and its position relative to other surrounding structures. In this section, we will briefly describe each of these characteristics.

1) Resistance: All wires have resistance, representing the ability of the wire to carry a charge flow. Aluminum wires have a resistivity of 3.3 mΩ·cm, while thin-film copper wires have a resistivity of 2.2 mΩ·cm. Resistance (per unit length) is simply calculated as the material resistivity divided by the conductor's cross-sectional area. Because of a thin barrier layer on three sides needed to prevent copper from diffusing into surrounding oxide (see Fig. 4), a wire's resistance as technologies migrated from Al to Cu did not quite decrease by 50%, although it did drop significantly. The barrier thickness for today's 0.18-μm generation is 17 nm [3]. Given the simple relationship between resistance and geometry, it is the easiest wire parameter to calculate; the following equation assumes a conformal barrier layer whose thickness on the sides equals that on the bottom:

$$R_{\text{wire}} = \frac{\rho}{(\text{thickness} - \text{barrier})(\text{width} - 2\ \text{barrier})}. \quad (1)$$

Skin effects for the vast majority of on-chip wires are not significant, since wires are less than a few skin depths thick

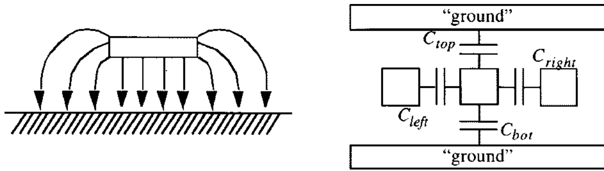


Fig. 5. Isolated and realistic capacitance models.

and wide. At 1.5 GHz, a frequency that corresponds to a signal transition time of about 100 pS, copper's skin depth is 1.7 μm, which exceeds most wire dimensions.

Interconnections between metal layers (plugs or vias) for aluminum wires were made of tungsten, and tended to be fairly resistive; in a 0.25-μm process, a M1–M2 via resistance was about 5 Ω and vias from M5 down to the substrate added up to more than 20 Ω. This may seem large considering a 1-μm-wide, 1-mm-long M5 line itself had a total resistance of only 20 Ω, but designers usually arrayed many vias together to reduce plug resistance at the cost of interlayer congestion. In most cases, electromigration checks required long wires to have arrayed vias anyway. Copper processes improve on this situation by pouring the vias out of copper at the same time as the wires are deposited. These copper vias are much less resistive and do not need to be as aggressively arrayed, although some recent experience has shown that copper vias have their own electromigration concerns [4].

2) Capacitance: All wires have capacitance, representing charge that must be added or removed to change the electric potential on the wire. Many analytical models approximate the capacitance of a wire over a plane; more accurate ones combine a bottom-plate term with a fringing term to account for field lines emerging from the edge and top of the wire. However, wires today are often taller than they are wide, and will grow even taller to reduce resistance as technologies scale. At minimum pitch their side-to-side capacitances are a significant and growing portion of the total. Capacitance is thus better modeled by four parallel-plate capacitors for the top, bottom, right, and left sides, plus a constant term for fringing capacitance, as shown in Fig. 5 [5]. The vertical and horizontal capacitors may have different relative dielectrics in technologies that use low-κ materials [6]

$$C_{\text{wire}} = \epsilon_0 \left(2K\epsilon_{\text{horiz}} \frac{\text{thick}}{\text{spacing}} + 2\epsilon_{\text{vert}} \frac{\text{width}}{ILD_{\text{thick}}} \right) + \text{fringe}(\epsilon_{\text{horiz}}, \epsilon_{\text{vert}}). \quad (2)$$

The "far" plates for the top and bottom capacitors are typically modeled as being grounded, since they represent a collection of orthogonally routed conductors that, averaged over the length of the wire, maintain a constant voltage.[1] Capacitors to the left and right, on the other hand, have data-dependent effective capacitances that can vary: if the left and right neighbors switch in the opposite direction as the wire, the effective sidewall capacitances double, and if they switch with the wire, the effective sidewall capacitances approach zero. This effect, known as "Miller multiplication," is modeled by varying the K parameter in (2) between zero and two. These left and right neighbors are also the worst offenders for noise injection. The fringe term depends weakly on geometry and for today's 0.18-μm technologies is about 40 fF/μm. For the very top layers of metal with no upper layers, we can use three parallel plates with extra fringing terms on the two horizontal capacitors.

3) Inductance: No handy closed form models exist for on-chip wire inductance, as they do for resistance or capacitance. Extracting inductance is a complicated task; we usually think of the inductance of a loop, and a changing magnetic flux through it induces a current on the loop. This view of inductance cannot be applied directly to on-chip wires, however, since we do not always know what wires will form the "loop." If we send current down an on-chip wire, for example, the return currents may flow in adjacent wires, parallel power supply buses, or even the substrate. In fact, because return currents will flow in the paths of least impedance, the actual return paths will change with the frequency content of the signal. At low frequencies, low-resistance power buses, even if relatively far away, are low-impedance ($Z = R + j\omega L$) and return currents will use them, creating fairly large loops and implying higher inductance. At high frequencies, far-away return paths have higher impedances, and return currents will bypass them to return in local, capacitively coupled wires, implying lower inductance.

To get around this problem of return path ambiguity, today's tools define return paths to be at a fixed common reference,[2] and the resultant "partial inductances," when combined with wire capacitances, can yield accurate results inside circuit simulation [7], [8]. Most of these tools still overestimate inductance since they assume that all of the current uniformly flows to the end of the wire, while in very large scale integration (VLSI) circuits, current actually returns through distributed and end-load capacitances [9]. The greater problem with inductance extraction is data explosion: since inductance falls with distance very slowly inside the return loop, wires that are separated by several pitches—or by several wires—can couple inductively. So for each extracted wire we must calculate the mutual inductance to all neighbors within several pitches, and the amount of data to extract and simulate quickly becomes unmanageable. Various sparsification schemes try to reduce this data without making the resultant coupling matrices unstable [10], [11].

A number of publications have proposed criteria for whether or not inductive effects are important; they essentially boil down to whether or not the signal near-end rise time is much faster than the propagation velocity down the wire, and whether the attenuation constant ($Z_0/2R_{\text{wire}}$) is

[1]This capacitance would be multiplied by an appropriate factor if the orthogonal wires switched simultaneously and monotonically, as with a precharge bus.

[2]The common return path can be arbitrarily picked, so long as it is consistent. The most (mathematically) convenient return path is at infinity; however, visualizing the resultant loop can be challenging.

greater than one [12]–[14]. For the vast majority of on-chip wires, these criteria show that self-inductance is negligible. However, because they all focus on the self-inductance of a single wire, they ignore the much worse problem of noise coupling. Noise, which depends on $M(\delta i/\delta t)$, is not as easily characterized and will be discussed in more detail later.

C. Signal Coupling

As was mentioned earlier, wires affect both circuit delay and robustness. This section looks at some of the coupling issues for VLSI wires, and the following section looks at delay and wire bandwidth. Coupling noise is a serious problem for a chip designer, since both mutual capacitance and inductance terms for wires can be large.

To understand the magnitude of coupling noise problems, we need to compare the induced noise to the noise margins of the receiving gate. Static and dynamic CMOS gates are voltage controlled—they switch their output voltage when the input voltage exceeds some threshold. Thus, we are concerned about the voltage noise on the wire relative to the voltage margins of the receiving gates.

Capacitance noise coupling is a larger effect so we will look at it first. The large aspect ratios of modern wires mean that for a wire surrounded by neighboring wires on either side, the cross-capacitance to these sideways neighbors dominates the total capacitance; sideways cap can exceed 70% of the total. When these sideways neighbors (the "attackers") switch, the current that flows through the coupling capacitors must then flow through the center wire (the "victim"), inducing noise on it. The familiar model of $V_{\text{noise}} = V_{\text{swing}} C_{\text{coupling}} / C_{\text{total}}$ gives a pessimistic upper bound on the noise, since this is the noise voltage only if the victim line is left floating. Many recent papers have modeled this noise more carefully and have shown that the noise voltage depends on both the coupling capacitance to total capacitance ratio as well as on the ratio of the strengths of the gates driving the two wires [15]–[17]. A convenient model simple enough for first-order hand calculations is

$$V_{\text{noise}} = V_{\text{swing}} \cdot \frac{C_{\text{coupling}}}{C_{\text{total}}} \cdot \frac{1}{1 + \frac{\tau_{\text{att}}}{\tau_{\text{vic}}}} \tag{3}$$

where τ_{att} and τ_{vic} are the time constants of the attacker and victim drivers, respectively. If the attacker has a much smaller time constant than the victim (and is hence much stronger), the noise approaches the pessimistic worst case. Typically, however, the transition times of different gates are matched, which gives an attacker-to-victim time constant ratio that is greater than one. If the two wires are identical, with identical drivers, the time constant ratio will be set by the difference between the effective resistance of a MOS transistor in the saturated region, driving the aggressor wire, and a transistor in the linear region, trying to hold the value of the victim wire stable. This ratio is usually between two and four,[3] which greatly reduces capacitive coupled noise for most nodes.

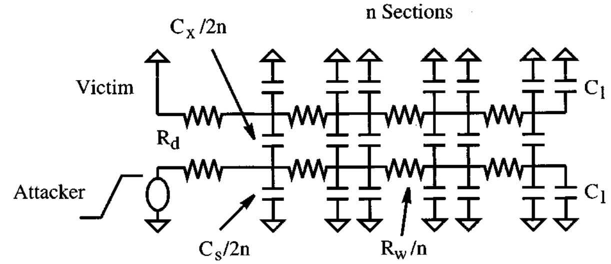


Fig. 6. Bus coupling noise model.

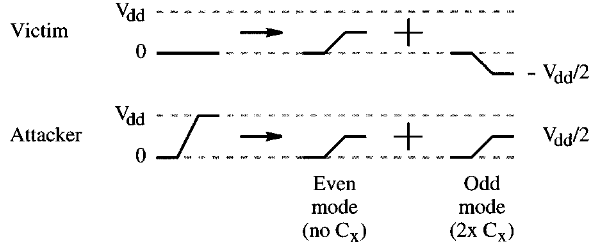


Fig. 7. Attacker and victim inputs, decomposed.

However, the limitation of this model is that it does not account for distributed line resistance. Adding this effect makes deriving analytical results difficult, leading researchers to use approximations like lumping the wire resistance with the driver resistance [16]. However, for the special case where the wires are identical, the most common case where coupling is a problem, there is a way to view the problem using superposition that gives a simple and intuitive view of coupling. This model starts by assuming that the driver resistances are the same, as shown in Fig. 6.

The key to the analysis is to break the driving input into a symmetric, or even mode, input (both sides are driven by a $V/2$ ramp), and an antisymmetric, or odd mode, input [attacker driven by $V/2$ ramp and the victim driven by $-(V/2)$]. In the even component, both attacker and victim see a half-amplitude input, and since the two lines now have identical responses, the coupling capacitors conduct no current and can be zeroed out. In this case, the response at the end of the victim is the same as that of a single wire in isolation, with total line capacitance $C_{\text{total}} = nC_s + C_l$.

For the odd component, the attacker sees a positive half-amplitude step, and the victim sees a negative half-amplitude step. In this case, the two lines have exactly opposite responses, so the coupling capacitors see twice the voltage difference and can be replaced by double-size capacitors referred to ground. Thus, we can once again treat the victim wire as an isolated single wire, with total line capacitance $C_{\text{total}} = n(C_s + 2C_c) + C_l$.

The combination of the even and odd modes, as in Fig. 7, will place a full step on the attacker driver and hold the victim

[3]The ratio hinges on the degree of velocity saturation of the attacking transistor. Since nMOS gates suffer more from velocity saturation, the ratio for nMOS gates is generally closer to 4.

driver to ground, so we need only add the two decoupled responses to get the true victim waveform. In other words, the victim response can be written as the sum of two isolated wire responses, one with no coupling, and the other with double coupling. These two isolated responses can be derived from a number of models, ranging from simple single time-constant exponentials to more complicated moment-matched asymptotic waveforms [18]. The key idea is that symmetry properties allow us to break the highly coupled circuit into two isolated circuits that are more easily handled.

Although this model requires identical driving resistances for attacker and victim, we can avoid this limitation by observing that a driving resistor that sees a step input can be transformed into a larger (weaker) resistor by using a slower exponential input. In other words, from the perspective of the downstream wire, a properly chosen exponential input driven into a resistor is almost indistinguishable from a step input driven into a larger (weaker) resistor. Thus, if we use an appropriate exponential input instead of a step input, and the smaller (stronger) victim resistance for both of the wire models, we will effectively increase the attacker driving resistance while maintaining the proper victim resistance.

The mathematical derivation, using simple single-time constant models for the wire responses, is unrewarding and not shown here, but it reduces to a peak noise given by[4]

$$V_{\text{peaknoise}} = \frac{C_{\text{coupling}}}{C_{\text{total}}} \cdot \left(\frac{1+M}{k+M}\right)^{(k+M)/(k-1)} \tag{4}$$

where $M = nR_{\text{wire}}/2R_{\text{att}}$ and k is the ratio of attacker to victim driving resistances (typically between two and four). For reasonable wire lengths, the driver resistance ratio does a good job of attenuating the noise pulse, making it a small issue for static CMOS circuits. However, capacitance coupling is a large problem for weakly driven nodes, and computer-aided design (CAD) tools must be used to check for coupling on such weakly driven or dynamic nodes.

Noise from inductive coupling can also present problems for VLSI wires. The current flowing down the aggressor wire generates a magnetic field which causes a backward return current to flow in the victim wire. Inductive coupling pushes the victim in the opposite direction from capacitive coupling: a rising attacker capacitively couples a victim up, but inductively couples the victim down. While capacitive coupling is mostly a "nearest neighbor" phenomenon, inductive coupling has a much larger range. Inductive noise becomes a problem only when a large number of wires switch at the same time in bus-like situations [19]–[21]. The worst case noise vector would have multiple wires switching, with near neighbors switching in one direction, and far neighbors switching in the opposite direction. This causes the capacitive and inductive noises to add, and the accumulated noise can be enough to cause failures [20].

Designers cope with inductive coupling by adding power planes or densely gridded power supplies to reduce the number of wires that can couple into a victim. Power planes, or dense power grids, effectively reduce both self and mutual inductances for wires in the direction of the grid, since they provide very nice return paths within a few micrometers of the wire itself and thus limit the extent of the magnetic coupling [22]. Most companies have design rules for buses to limit the inductive noise to acceptable levels.

[4]Note that this formula reduces to a slightly different result than (3) when the wire resistance is 0 (i.e., when $M = 0$). In these cases, this equation gives a better result.

D. Delay and Bandwidth

Now that we have discussed wire characteristics, we can summarize the wire performance with delay and bandwidth metrics. The delay of a gate driving a wire comes from a simple RC formulation

$$\begin{aligned} \text{Delay} \propto\ & R_{\text{gate}}(C_{\text{diffusion}} + C_{\text{wire}} + C_{\text{load}}) \\ & + R_{\text{wire}}(\tfrac{1}{2}\,C_{\text{wire}} + C_{\text{load}}). \end{aligned} \tag{5}$$

This is only an approximation, since it ignores slew rates: if a preceding wire is long enough that its end voltage slews very slowly, it will degrade the delay of the next gate.

This model does not include any explicit inductive terms and assumes that delays are dominated by RC effects. Inductance can have four effects on delay, some more important than others. First, signal propagation on a wire, or the local speed of light, is set by $\sqrt{LC}$, in ps/mm. Thus, when the front end of a wire switches, the far end of the wire cannot begin to switch until at least $l\sqrt{LC}$. With typical on-chip inductance numbers around 2–5 nH/cm [23], however, RC terms dwarf this effect. Second, over-driving wires with too-large gates (a common failure of some synthesis tools) can cause LRC wires to become underdamped and the resulting waveshape to have a delay poorly predicted by a dominant RC time constant. However, keeping driver fanouts reasonable (i.e., at least four and higher for resistive lines) keeps wire responses well within RC domains and the "sharpening" effect of inductance to within a small percentage of total line delay. Third, inductive coupling, much like capacitive coupling, can push out delay by forcing a victim to absorb induced transients before swinging. Designers can very roughly approximate this effect by modulating the C_{wire} term in a manner analogous to capacitive Miller-multiplication; this zeroth-order approximation is extremely crude and does not scale, but it has the virtue of being easily integrated into existing tool flows. Fourth, inductance can force return currents into tighter loops with higher resistivity than wider loops. This extra "return path resistance," often overlooked, can be significant, and designers can include it by increasing the R_{wire} term. These last two effects are the most important, but they are also very geometry-dependent, so we will not include them in the discussions below.

The first term in the delay equation above is about 1FO4, because simple sizing heuristics aim for gate sizes to have a fanout of about four for optimal delay [24]; such sizing rules avoid huge gates for really long wires since wire resistance will shield downstream capacitance. We will also assume that $C_{\text{wire}} \gg C_{\text{load}}$ for long wires in excess of 1 mm. Our metric for delay is therefore simply 1FO4$+(1/2)R_{\text{wire}}C_{\text{wire}}$. These assumptions do not hold for wires driving large or many gate loads, such as repeated wires (which we will consider later)

Table 1
Sample $1/2R_{\text{wire}}C_{\text{wire}}$ Delays, 0.18 μm Technology

0.18μm tech	Local	Semi-global	Global
Wire delay (Cu), FO4/mm^2	0.56	0.22	0.05

or control wires driving each bit of a wide datapath. Representative delay numbers for a 0.18-μm technology are shown in Table 1; this table lists a worst case delay capacitance number.

As Section III-B describes in more detail, modern technologies optimize their metal layers for three different tasks. The lowest level metals are used for local interconnections; the semiglobal wires, on midlevel layers of metal, typically run within functional units; the global wires, on the top layers of metal, route power, ground, and global signals. The wire delay for all three classes of wires are given in the table.

For a copper 0.18-μm technology, long unbuffered wires with small loads are not too slow. A 10 mm route takes $1 + 56 = 57$ FO4s on local wires, but $1 + 22 = 23$ FO4s on semiglobal lines, and only $1 + 5 = 6$ FO4s on global wires. Significant gate loads can increase this delay, and using repeaters can decrease it; repeaters will be discussed in more detail later.

We can also estimate the bandwidth of an unbuffered wire by asking how long we must wait between successive transitions on a wire. If we switch a wire once, we need to wait until residual currents from that transition have mostly died away, or else we will see intersymbol interference when we switch it again. We can do this by waiting for three propagation delays before sending the next signal. This is enough time for the output to transition past 90% of its final value. In (6), we assume the propagation delay to be a gate delay (1FO4) plus the distributed wire delay. Increasing a wire's pitch will monotonically increase that wire's bandwidth, since it decreases the wire RC product, leading to the misleading result that fatter wires are always better. Therefore, we will actually examine the bandwidth across a routing area. In this case, making wires excessively fat will reduce the number of wires available, and hence potentially reduce bandwidth over that area:

$$BW_{\text{area}} = \frac{1}{3(\text{1FO4} + D_{\text{wire}})} \cdot \frac{\text{Blockwidth}}{\text{Wirepitch}}. \quad (6)$$

This formulation allows us to examine unrepeated bandwidth in both local and global contexts. For module-length wires, we run semiglobal layer metals across a square that holds around 50 000 gates. For global wires, we run top-level metals across the entire die and thus consider the bandwidth across a die-sized square.

Fig. 8 shows module and global unrepeated bandwidth. In (6), the left-hand 1/delay term rises with increasing wire pitch, but the right-hand "number-of-wires" term falls with increasing pitch. Whether or not designers should increase the wire pitch depends on the wire length: if the wire is short enough that its delay is dominated by gate delay, then the bandwidth improvement from increased pitch tends to be less than the bandwidth degradation from fewer wires. If the wire is long enough that its delay dominates gate delay, then bandwidth is improved by increasing pitch. In Fig. 8, we see that increasing wire width does not improve local bandwidth, but it slightly improves global bandwidth.

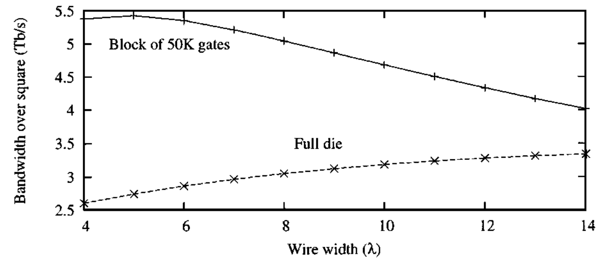


Fig. 8. Unrepeated BW, 0.18-μm technology.

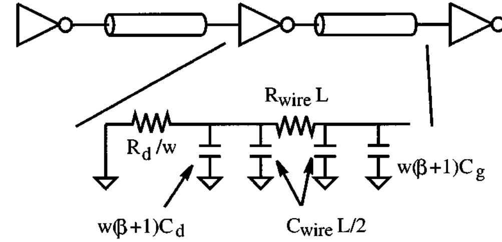


Fig. 9. First-order repeater model.

The long delay and low bandwidth of the global wires clearly indicates a problem caused by the large resistance of these wires. Fortunately, there is a simple way to dramatically reduce the effect this resistance has on circuit performance—we can break these long wires into a number of shorter segments by adding gain stages between the segments. These stages are called repeaters.

1) Repeaters: Since the delay of an uninterrupted wire grows quadratically with wire length, designers can add repeating elements periodically along the wire. This makes total wire delay equal to the number of repeated segments multiplied by the individual segment delay; total wire delay is hence linear with total wire length. A first-order model of repeaters is shown in Fig. 9, where R_d is the driver resistance in $\Omega \cdot \mu$m, w is the width of the driver transistor, C_d and C_g are diffusion and gate resistances per unit width, R_{wire} and C_{wire} are wire resistance and capacitance per unit length, l is the repeater segment length, and β is the pmos-to-nmos sizing ratio [25].

This first-order model leads to a total wire delay of

$$D = 0.7n \left[\frac{R_d}{w} [w(\beta+1)(C_d + C_g) + lC_{\text{wire}}] + l^2 \frac{R_{\text{wire}} C_{\text{wire}}}{2} + lR_{\text{wire}} w(\beta+1) C_g \right]. \quad (7)$$

Taking the derivative with respect to n, after setting $l = L/n$, leads to segments that are long enough that their intrinsic wire delay equals a repeater stage delay. For unloaded wires and inverter-repeaters, this implies a propagation latency of $2.13\sqrt{R_{\text{wire}} C_{\text{wire}} \text{FO1}}$ pS/mm, with a $\text{FO1} = R_d(\beta+1)(C_d+C_g)$ equal to the delay of an inverter driving an identical copy of itself (with typical diffusion capacitances, a FO1 is one-third a FO4). The repeaters

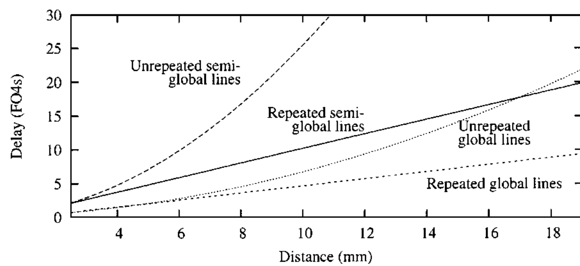
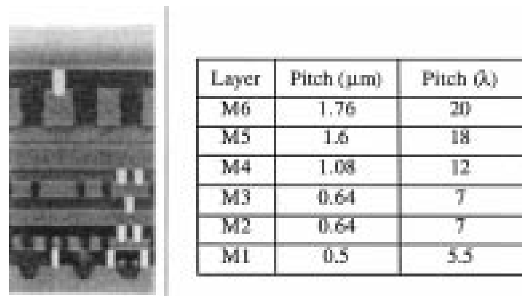


Fig. 10. Repeated and nonrepeated lines, 0.18-μm technology.



Layer	Pitch (μm)	Pitch (λ)
M6	1.76	20
M5	1.6	18
M4	1.08	12
M3	0.64	7
M2	0.64	7
M1	0.5	5.5

Fig. 11. An Intel 0.18-μm technology [26].

tend to be area- and power-hungry, however, since this derivation (which is independent of technology generation) leads to repeaters whose gate capacitances are 60% of each segment's wire capacitance, for a driver fanout of 2.7. In a 0.18-μm technology, for example, these are global wires 2.5 mm (28 kλ) long, driven by 100 μm (1100 λ) gates.[5]

A better repeater strategy would optimize the delay–power product (or, equivalently, the delay–capacitance product). This leads to segments that are longer by 1.7$\times$ and gates that are smaller by 0.6$\times$. The repeaters thus have gate capacitances that are 20% of the wire segment capacitance, for a more reasonable driver fanout of 5.8. The propagation latency rises by about 13%, but the total capacitance falls by 30%. Again, the derivation is independent of technology. In our 0.18-μm example, these are wire segments 4.25 mm (48 kλ) long, driven by 60 μm (660 λ) gates. Sometimes, however, noise considerations prevent us from running such long segments between repeaters.

Fig. 10 shows example delays of unloaded semiglobal and global wires in this technology. In this figure, the wires terminate in very small loads.

From a design perspective, inserting repeaters into a design can be complicated. First, using inverting elements requires an even number of repeaters to avoid logic inversions on the wire.[6] Second, repeaters for global wires require many via cuts from the upper-layer wires all the way down to the substrate, potentially congesting routes on intervening layers. Third, designers are rarely afforded the luxury of placing repeaters in their optimal locations; since they require active area, designers usually floorplan repeaters in clusters. Finally, even with delay-power optimizations, repeaters are still large devices, and repeating an entire bus takes an impressive amount of silicon area. Fortunately for these last two complications, delay and capacitance curves for repeater insertion have fairly shallow optimizations, so that adding or removing a single repeater stage, moving repeaters back and forth, or resizing repeaters have fairly small costs.

Repeated wires offer substantially increased bandwidth. After sending one signal down a wire, we only need wait until that signal fully transitions on the first repeater segment before we send the next signal; the bandwidth of a repeated wire does not depend on the entire wire length. Also, increasing the wire width makes the segment length longer but does not change the segment delay, so wider wires only reduce the number of available routing tracks and hence do not improve bandwidth. In our previous example of bandwidth, using repeaters increases local bandwidth by 1.6$\times$, and global bandwidth by almost a factor of 10.

III. Technology Scaling

Before we look at how technologies will scale, we will first look more closely at a contemporary 0.18-μm technology to set some of the geometry assumptions that we will use when we explore scaled technologies. Our 0.18-μm baseline technology has six layers of copper interconnect, with upper layers wider and taller than lower ones. The lowest metal layer, M1, has the finest pitch and hence the highest resistivity, and it predominantly connects nets within gates or between relatively close gates. The middle layers, M2 through M4, have a wider pitch than M1 and connect both short- and long-haul routes, typically within functional units. The top layers, M5 and M6, have the widest pitch and hence the lowest resistivity and they usually carry global routes, power and ground, and clock. Fig. 11 shows typical pitches for these various layers in technology-independent λ's, where a λ is half of the drawn gate length. The features at the upper layers are clearly much grosser than the comparatively tiny features at the bottom of the picture.

In our baseline technology, local wires have a pitch of 5 λ, semiglobal wires a pitch of 8 λ, and global wires a pitch of 16 λ. Details are shown in Table 2. We will use these pitches (in λ) for our scaled technologies.

We see two different approaches to looking forward. First, one could consider technological limitations, and forecast wire and gate performance from projected roadblocks. The 1994 SIA roadmap did this, and used limitations such as oxide thickness and clock frequency scaling to arrive at projections of wires and gates [27]. The risk with this approach is that clever people will figure a way around some of these limitations and exceed the projections; a glance back at the 1994 roadmap shows that this indeed happened, and the industry has advanced far beyond the 1994 predictions.

Second, one could simply project from current trends, without regard to potential looming technical obstacles. The 1997 SIA roadmap followed this strategy and extrapolated from recent history to guess at future wire and gate per-

[5]Here, a λ represents half of a gate length. Describing distances in λ's is convenient because λs are technology-independent units.

[6]Designers may opt to use buffered repeaters, which are two inverters back-to-back. The delay-optimal design for such repeating elements shifts a bit: segments are 87% longer since the repeating element has more delay, and overall propagation delay is about 14% worse. However, inserting buffers is logically easier.

Table 2
Dimensions for Our Example 0.18-μm Technology

0.18μm tech	Local	Semi-global	Global
Width, μm	0.27	0.36	0.72
Spacing, μm	0.18	0.36	0.72
Height, μm	0.378	0.720	1.584
Resistance, Ω/mm	258	96	20
Capacitance, fF/mm (ε_{horiz}=3.72, ε_{vert}=3.9)	431	414	430
% of Cap is Xcap	75%	76%	77%
Delay ($R_{wire}C_{wire}/2$) FO4s/mm^2	0.56	0.22	0.05
Repeated propagation velocity FO4/mm	--	1.2	0.6
Repeated bandwidth (block and chip), Tb/s	--	8.75 (block)	32.1 (chip)

formance [1]. Such a strategy often requires miracles and, indeed, the 1997 roadmap forecast clock frequencies that will be difficult, if not impossible, to achieve.

In our approach, we hedge our bets by making not a single prediction of technological scalings for wire characteristics but rather a range of predictions. We will use both aggressive and conservative scaling projections to bound future parameters, and hope that by doing so, we will encompass a broad enough range that actual wire performance will fall within these bounds. In the discussions below, we will show results for both aggressive and conservative scaling; not only does this give us a better chance of predicting future performance, it also helps us determine the sensitivity of these trends.

A. Gate Delay Scaling

Historically, gates have scaled linearly with technology, and an accurate model of recent FO4 delays has been 360*L_{gate} pS at typical and 500*L_{gate} pS under worst case environmental conditions (typical devices, low V_{dd}, high temperature). Fig. 12 shows FO4 delays for a number of different process technologies running at the worst case environment corner. This trend may continue for future generations of transistors, since devices seem scalable down to drawn dimensions of 0.018 μm [28]. Whether or not such devices obey the above delay model is uncertain, because of issues in scaling gate oxide, V_{dd} and V_{th}. These concerns mean 500*L_{gate} pS is a lower limit for future FO4 delays. Since we are considering wire delays relative to gate delays, faster gates provide the worst case for wire issues, and thus we will use this model as our gate delay projection.

Other device parameters, such as gate and diffusion capacitance, are assumed to scale nicely. We assume gate capacitance, now around 1.5–2 fF/μm, will stay constant; although this would seem to demand too-thin gate oxides, high-κ dielectrics may allow more aggressive scaling of the effective T_{ox} [29]–[31]. We project diffusion capacitance to stay at about half gate capacitance for legged devices, although trench technologies and/or SOI can reduce this dramatically [32].

To predict chip clock cycle times under process scaling, we can examine the number of FO4s per cycle. Fig. 13 shows some historical data from Intel microprocessors for various

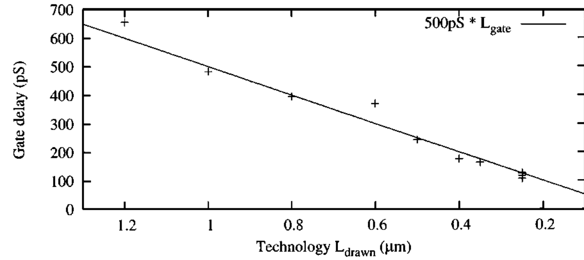


Fig. 12. FO4 scaling (typical, 90% V_{dd}, 125°C).

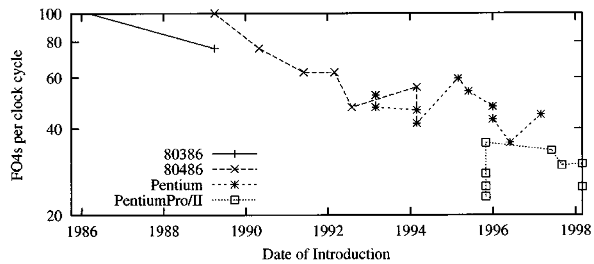


Fig. 13. Historical FO4s per clock, ×86 machines.

microarchitectures ranging from the nonpipelined 80 386 to the out-of-order execution PentiumPro [33].

Current machines cycle between 20 and 30 FO4s per clock, and the upcoming Pentium4 microarchitecture and the aggressive Compaq/DEC Alpha chips sit at around 14 to 16 FO4s per clock [34], [35]. This may look misleading, since the Pentium4 processor, at 1.4 GHz in an 0.18 μm process, would appear to have a cycle time of 714 pS and an FO4 of 90 pS, or 8 FO4s per clock. However, some technologies have an L_{gate} that is significantly smaller than the base technology feature size. For example, the Intel 0.18-μm process, because of poly profile engineering, ends up with an L_{gate} of 100 nm [36]. (This is not the same as saying that "electrical gate length is smaller than physical gate length," since the narrowing of L_{gate} is due not to diffusion undercut, but rather to poly notches. In fact, the electrical gate length for this process will be smaller still, although L_{elec} is irrelevant to our FO4 model, which uses physical gate length.) Hence, our model would more properly estimate FO4 delays for the Intel 0.18-μm process

Table 3
Optimistic Gate Delay (Typical Devices, Low V_{dd}, High Temperature) and Clock Scaling

L_{drawn}	0.18μm	0.13μm	0.10μm	0.07μm	0.05μm	0.035μm
FO4, pS	90	65	50	35	25	17.5
Frequency, GHz	0.7	1	1.25	1.8	2.5	3.6

Table 4
Wire Dimensions Scaling

L_{drawn}	0.18μm	0.13μm	0.10μm	0.07μm	0.05μm	0.035μm
Semi-global pitch, μm	0.36	0.26	0.20	0.14	0.10	0.07
Global pitch, μm	0.72	0.52	0.40	0.28	0.20	0.14
Chip edge, mm	19	20.7	22.8	24.9	27.4	30.1

as $500^*0.100 = 50$ pS, giving the Pentium4 processor 14 FO4s per clock.

However, extrapolating future clock cycle times from Fig. 13 can lead to unrealistic predictions: a linear fit would lead us to expect clock cycles of 6–8 FO4s per clock within a few generations. Such fast-cycling machines pose two circuit design problems. First, timing overhead for latch-based designs becomes a prohibitive fraction of the clock cycle when the system runs faster than 16 FO4s per clock. Although some circuit strategies exist for mitigating clock skew overhead [33], synchronization penalties will still represent an ever-increasing percentage of the available time as the cycle time shrinks. Second, and more importantly, generating a clock that spins at 8 FO4s per clock is extremely difficult, since the rise and fall times of a clock wave take more than 2 FO4s to fully transition. A clock that tries to rise and fall within 8 FO4s will appear sinusoidal in nature and be susceptible to power supply-induced jitter and other timing uncertainties. In the following discussions, we will set the number of FO4s per clock cycle to be 16, as shown in Table 3.

B. Wire Scaling

Wire scaling is difficult to predict because of the greater number of physical parameters that govern wire electrical characteristics. To capture a reasonable range of possible wire futures, we will choose a set of conservative and aggressive scaling parameters. The conservative projections assume limited technological improvements, such as low-κ dielectrics scaling at 0.9× and hence at the 0.035 μm technology reaching $\epsilon_r = 2.2$, and that thin-film copper is the lowest resistance metal available. The aggressive projections, assuming ϵ_r scaling down to 1.4 and bulk copper resistivity, will be more in line with the 1997 SIA roadmap. The recent 1999 SIA roadmap's more middle-of-the-road projections fall in between these two extremes [3].

In both sets of scaling projections, we will maintain the semiglobal pitch to be 8 λ and the global pitch to be 16 λ. The chip edge length is growing but slower than the scale factor S. These dimensions, along with chip edge length, are shown in Table 4. However, because of performance and power delivery constraints, designers may choose to give the very top layers of metal a thickness and pitch that stays constant in micrometers. These global wires thus scale upwards in size relative to the rest of the metal layers, and will have superior current-carrying and delay characteristics, enabling global delays to scale with gate delays. Such "superwires" were first envisioned by Song and Glasser [37] for electromigration and voltage drop considerations. Our discussion does not assume their usage.

1) Resistance: Resistance grows under scaling, since the width and height both scale down, although the height does so more slowly as the aspect ratio grows. Our optimistic scaling pushes the aspect ratio at the expense of coupling capacitance, and introduces bulk copper resistivity at the 70-nm generation. The conservative scaling limits the aspect ratio to control coupling and assumes the best metal to be thin film copper. Concerns about future scaled resistances are twofold [38]. First, the barrier layers, if not well conformed to the edges of the conductor, may end up much thicker on the bottom than on the sides, and thus dramatically decrease the available cross section. Recent advances in atomic layer deposition, however, may enable very conformal barrier layers at a cost of decreased fabrication throughput. Second, as wires get thinner and thinner, and approach a thickness equal to the mean free path of electrons, their edges present scattering targets for the electrons. These carrier collisions effectively reduce the mobility, up to around 10% in copper [38]. The numbers in Table 5 and Fig. 14 show a dramatic increase in resistance under technology scaling.[7]

2) Capacitance and Inductance: Capacitance decreases very slowly with technology due to projected advances in low-κ dielectrics. The conservative scaling projections cap the aspect ratios to keep the sidewall capacitance less than 75% of the total capacitance. The aggressive scaling projections also keep this ratio under 75% despite an aspect ratio that approaches 3, due to the aggressive low-κ dielectrics placed in between wires and not in between wire layers [6]. The fringe terms for the different process generations came from fitting the capacitance equation (2) to results from a field solver [39]. The numbers shown here in Table 6 and Fig. 15 are for worst case delay; the side-to-side capacitances are "Miller-multiplied" by a factor of 2 to simulate the simultaneous switching of adjacent wires. The table and figure show that capacitance does not change much over time, and that the aggressive and conservative scalings are not that different.

Like capacitance, inductance per length should be roughly constant with scaling. In fact, the rising aspect ratios of the wires will cause the value to slightly decrease. More important than the wire aspect ratio is how the power and ground networks scale, since current returns limit the inductive coupling of the wires. While the design of the supply is chip-dependent, the trend is for denser power distribution networks to lower the supply impedance for each technology shrink [40]. About a 2× reduction in supply impedance is needed

[7]One way to reduce wire resistance significantly is to actively cool the chip. Although currently expensive, refrigeration can lower copper resistance by almost an order of magnitude as temperatures drop from 300 K to 77 K.

Table 5
Resistance Scaling

		0.18μm	0.13μm	0.10μm	0.07μm	0.05μm	0.035μm
conservative	Global aspect ratio	2.2	2.3	2.5	2.5	2.5	2.5
conservative	Semi-global (Ω/μm)	96	184	307	627	1220	2509
conservative	Global (Ω/μm)	20	37	58	118	231	473
aggressive	Global aspect ratio	2.2	2.5	2.7	2.8	3	3
aggressive	Semi-global (Ω/μm)	96	168	260	340	600	1224
aggressive	Global (Ω/μm)	20	35	54	82	150	306

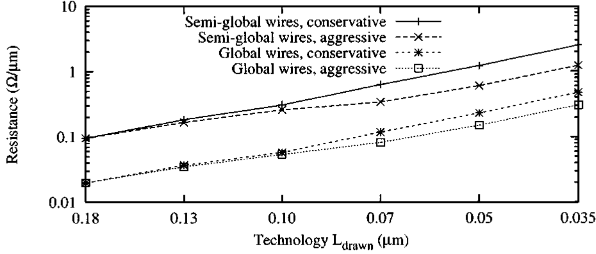


Fig. 14. Resistance scaling.

in each generation to maintain the same relative amount of supply noise [41].

3) Noise: Noise coupling, both capacitive and inductive, should be mostly unchanged under scaling as long as the wires scale in length. In both the conservative and aggressive scaling scenarios, the ratio of sidewall to total capacitance is held to at most 75%. The overall capacitive coupling noise thus depends on the scaling of the ratio of the wire resistance to the driver resistance. If the wire lengths scale, the wire resistance scales down either slowly (conservative) or more rapidly. Since the driver resistance is relatively constant with scaling, this leads to a coupling noise relative to the power supply (and hence to gate noise margins) which is either constant or slowly scaling down. If the wire lengths remain constant, the increase in wire resistance will cause the coupling noise to increase slightly. This increase in noise for long wires is another reason to use repeaters.

Inductive noise, depending as it does on a superposition of $M \cdot (\delta i/\delta t)$ terms, stays constant relative to the power supply for scaled-length wires. This is because the mutual inductance, M, stays constant per unit length, so that the total mutual inductance scales down with shorter wires. The total capacitance, and thus total current, also scales down, so the $\delta i/\delta t$ term stays constant. Thus the product of the two scales downward, along with the power supply V_{dd}.

For wires that do not scale in length, inductive noise can grow relative to the power supply, but more likely than not, these wires will be repeated. Repeaters break up the current return paths effectively, making each repeated segment independent from the rest, and preventing inductive noise from growing over technologies.

C. Delay and Bandwidth

In discussions about wire delays under technology scaling, we need to make an important distinction between two kinds of wires, shown in Fig. 16. The first kind of wire connects gates locally within blocks, and when devices (and blocks) get smaller, these wires get shorter. The second kind of wire connects blocks together and usually spans a significant part of a die; when devices and blocks get smaller, these wires typically do not shrink.

1) Wires that Scale in Length: Since wires that scale in length have both resistance and capacitance multiplied by a length scaling factor, they show essentially a constant wire resistance and a falling wire capacitance. The delay of these kinds of wires thus scales with technology, as shown in Fig. 17. This figure shows the delay of a wire that spans a block of 50 K gates. For the aggressive projections, the wire delays scale with gate delays until the 0.05-μm generation, and then grow slowly. For the conservative projections, wire delays get 4$\times$ worse than gate delays over six technology generations. This increase in wire delay will of course be smaller if gates do not follow the aggressive scaling that we have assumed.

2) Wires that Do Not Scale in Length: These kinds of wires do show an increasing delay disparity with gates; over technology scaling, the wire delay $R_{\mathrm{wire}}C_{\mathrm{wire}}$, relative to gate delays, roughly doubles each generation, for both aggressive and conservative scaling trends. Fig. 18 shows the delay of 1 cm wires relative to gate delays on a log scale. Fortunately, designers avoid such long wires running across a chip, and use various mitigating techniques, such as repeaters, wider wires, or wire layer promotion (pushing critical wires to a higher, and thus better, wire layer). The next section provides data for the delay of a repeated fixed length wire.

3) Repeated Wires: From Section II-D1, we saw that the propagation delay of a repeated wire is proportional to the geometric mean of wire delay ($R_{\mathrm{wire}}C_{\mathrm{wire}}$) and a FO1 delay. Under scaling, wire capacitance is largely unchanged, and resistance grows just slightly faster than gate delays fall. Thus the repeated propagation delay is basically constant under technology scaling: for global wires, this delay changes from about 55 pS/mm in a 0.18-μm technology to around 80 pS/mm in a 0.035-μm technology, a change of less than 1.5$\times$ over six technology generations (see Fig. 19). That propagation delay is unchanged in pS/mm under scaling often surprises designers, and it highlights the notion that wires themselves are not degrading in performance as much as chip complexity and performance are outpacing what wires can offer.

Table 6
Capacitance Scaling

		0.18μm	0.13μm	0.10μm	0.07μm	0.05μm	0.035μm
conservative	Sideways ε_r	3.75	3.375	3.038	2.734	2.460	2.214
conservative	Semi-global (fF/μm)	414	387	359	333	311	295
conservative	Global (fF/μm)	440	423	413	381	355	334
aggressive	Sideways ε_r	3.75	2.5	2	1.75	1.5	1.4
aggressive	Semi-global (fF/μm)	414	343	314	307	296	287
aggressive	Global (fF/μm)	440	370	335	312	296	287

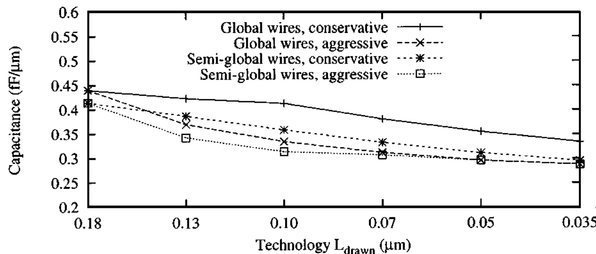


Fig. 15. Capacitance scaling.

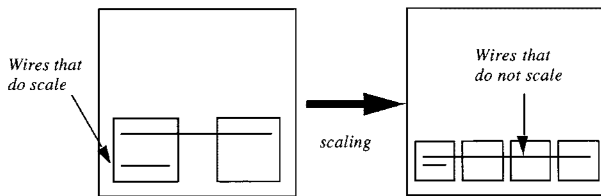


Fig. 16. Some wires scale in length; some do not.

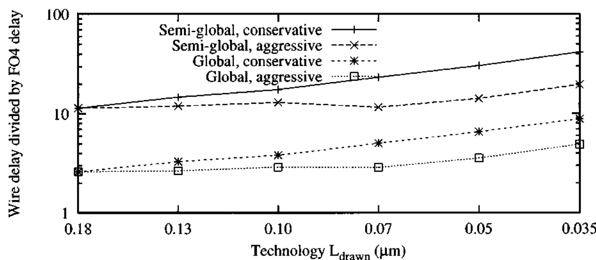


Fig. 17. Wire delays (in FO4s) for scaled-length wires spanning 50 K gates.

Designers will rarely use global wires without repeaters, so the reachable distance per clock will be set by repeated wire velocity. Fig. 20 shows how far a signal can reach within a clock. The importance of this distance lies in implicit versus explicit architectural latencies: spans that lie within the reachable distance per clock need not be broken into pipestages or otherwise synchronized across cycles, while spans that cannot be crossed within a clock will have architecturally explicit latencies. On the left side of Fig. 20, we show this reachable span in micrometers, while in the right side we show it in λs. Notice that although the reachable span is decreasing in absolute distances, the logical span in λ is essentially constant over many technology generations, supporting the earlier conclusion that designs that shrink will have nicely-scaled performance. Again, the problem is one of growing complexity; as technologies scale, each die will have many, many more λs per edge.

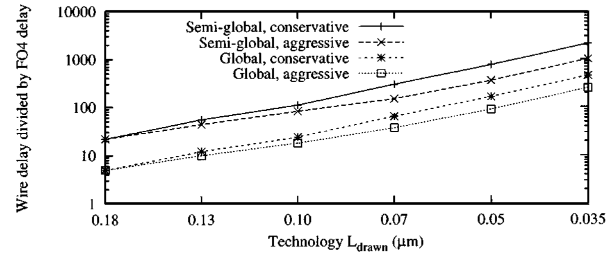


Fig. 18. Wire delays (in FO4s) for fixed-length wires 1 cm long.

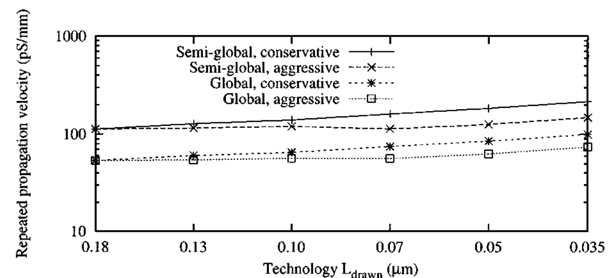


Fig. 19. Repeated propagation velocity (optimal delay-cap).

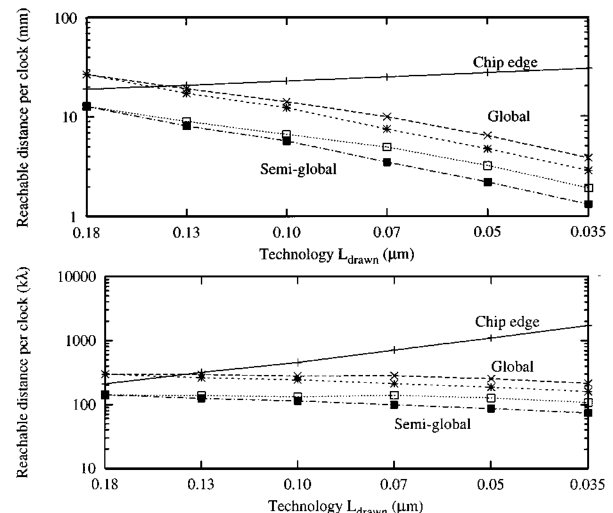


Fig. 20. Reachable distance per clock, repeated global wires.

The bandwidth of repeated semiglobal and global wires over a range of technologies is shown in Fig. 21. Conservative and aggressive projections make no difference in this graph since under optimal repeating, the segment delay is a

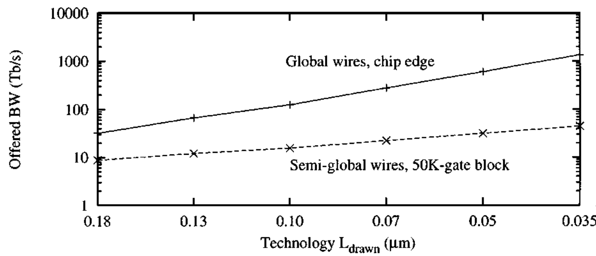


Fig. 21. Bandwidth of repeated wires over scaling.

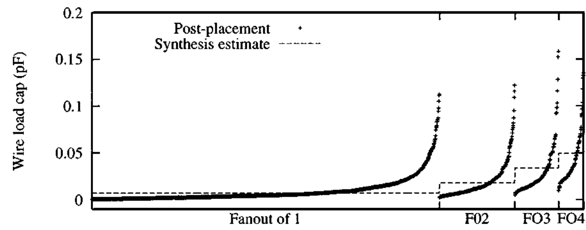


Fig. 22. Estimated and actual wire loads for a sample 0.5-μm design [42].

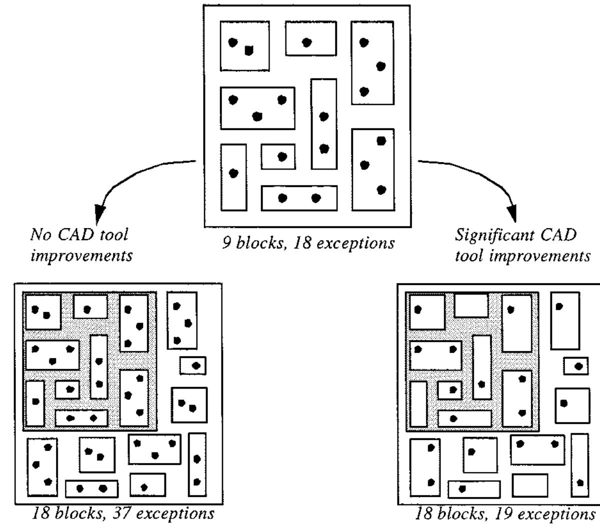


Fig. 23. Scaling with and without CAD tool improvements.

fixed number of FO4 delays, so we only show one set of projections.[8]

IV. Implications

One interesting view of wire scaling is that the real problem is not with the wire, but rather with the increasing complexity that scaling enables. The previous section showed that if the length of the wires scales with technology, the magnitude of the "wire problem" is small. The main problem is with wires with increasing logical span—wires need to communicate across more and more gates as technology scales, and these wires cannot keep up with the scaling gate delays. Wire delays are greatly improved by using repeaters, yet at best this makes the delay constant, which still means that compared to a gate these wires are getting effectively slower. It is these slower wires and the increasing total number of wires on a chip that will force changes in the way we design chips and the tools used to support design.

A. CAD Tools

Wire parameters are very important in determining a gate's performance, since they both increase gate loading as well as add intrinsic wire delays. In a normal CAD design flow, synthesis of the logic for a module occurs before gate placement. Since the CAD tool does not know exact wire lengths and capacitive loads, it synthesizes initial logic structures and netlists using fanout-based wire load models, usually supplied by library cell vendors. These wire load models come from statistical analyses of past designs and represent the median of the wire load distribution for each fanout. However, post-layout wire capacitances have Poisson distributions with a narrow peak around the statistical length and long tails. Fig. 22 shows the discrepancy between post-layout and statistical wire load models for a small design, where nets are sorted by fanout and then by capacitance.

Thus, even though synthesis reasonably estimates the wire load of the multitude of shorter wires, it highly underestimates the longer nets. These longer nets will be driven with underpowered logical structures and have unaccounted intrinsic wire delays. Thus, they may not meet timing closure through layout optimization techniques. These longer nets that potentially break the CAD design flow are called "wire exceptions," and dealing with these exceptions is a time-consuming process for designers.

We have seen already that for a given block that scales to a new process technology, the performance of wires in the block scales nicely, since the wires get shorter. Said differently, the wire exceptions in a block get only slightly worse under technology scaling. Thus, were we to simply scale a design from one technology to another, then the wire problems would be largely unchanged and our existing CAD tools would be sufficient: wire exceptions would still be painful, but they would not be much harder or more complicated.

However, designers rarely simply scale a design. They usually take advantage of the larger number of available transistors and wires to increase functionality or performance. But this increase in design complexity means that if CAD tools are unchanged, and if each block still generates a fixed number of wire exceptions, then the exponentially increasing number of blocks will spawn an unmanageable number of wire exceptions, as illustrated in Fig. 23. Since the cost to fix each wire exception can be measured in designer days, unless the CAD tools improve to reduce the number of exceptions per block, productivity will limit design time.

This need to handle larger designs without growing the design team is driving a number of new CAD tools to better support long wires. These include better wire load predicting, more accurate and explicit wire resistance modeling, and combining synthesis and layout. Much current work explores these areas, especially in layout-driven synthesis [42]–[46].

[8]The segment length is a function of both gate delays and wire parameters, and hence differs between aggressive and conservative scalings. In all cases, the segment length is shorter than the edge of a 50 K gate block, so it does not change the offered bandwidth.

These tools can deal with wires with significant wire delay, thus decreasing the number of exceptions, and will be able to handle the potential slow growth in wire delay relative to gate delay. With these new CAD tools, module level wires should not be an issue.

B. Architecture

The previous sections described an interesting set of constraints for digital systems architects of the future. Architects can view their job as translating vast amounts of silicon real estate, filled with transistors and wires, into system performance. However, technology scaling changes the nature of the fundamental building substrate. So architects themselves must adapt to these changes in the underlying technology.

In many ways, wire quality is not degrading. The number of logic gates reachable in a cycle will not change significantly,[9] and wires will continue to provide ever-increasing on-chip bandwidth. But this picture ignores the exponentially growing number of gates on a chip; we are reaching or have reached a point where more gates can fit on a chip than can communicate in one cycle. Said differently, the absolute distance that a signal can travel in a clock cycle has been decreasing exponentially for a long time, but it has never mattered since this distance has been larger than a chip—until recently.

That on-chip communication has been cheap for a long time has driven a number of architectural models relying on low-latency communication to shared global resources. Programmers find these models attractive, since they provide the most uniform computational framework and the best functional unit utilization. This focus on function rather than communication is pervasive and is the fundamental conceptual roadblock to overcome in the future. If we take our current architectures and try to increase their complexity, those designs will encounter problems. These problems will arise because our current architectures are function-centric and implicitly assume that global communication is extremely cheap or free [49], [50].

In older technology generations (around 2.0-μm processes), few functional units actually fit on a die, so maximal use of these few functional units was paramount. Fitting all the needed functions on the chip was the critical design goal. The number of gates that fit on a chip was less than 50 K, the size of today's synthesized blocks. Global wires were not a problem—the logical span of the longest wires were quite short. Wire resistance was not an issue.

As technologies improved, wire resistance remained small, but the increasing capacitance of long wires became significant. Floorplanning of high-performance designs became an important design step, so that proper device sizing could keep communication costs low. Designers not needing maximum performance could still ignore wires, and they used ASIC tool flows which did synthesis first, followed by placing and routing of the design. Continued technology scaling led to the situation where global wire delays were nontrivial but still much less than a clock cycle. In this design period, the programming model remained one of globally shared resources, but with microarchitectures increasingly partitioned. For example, the instruction fetch unit, while logically part of the datapath, physically migrated to the cache to minimize branch latencies. The address adder in many machines was duplicated: one in the datapath where it logically belonged, and a smaller version near the data cache to generate the cache index, again to reduce latency. Wires delays were still modest (much less than a cycle), and these microarchitecture changes were mostly invisible to the user.

Designers developed many tools and methodologies to handle the increasing importance of wires during this period (the beginning of submicrometer design), including analytical wire RC models and more accurate AWE simulation; floorplanning techniques such as delay-driven segregation of local and global routing; and local circuit generation techniques such as layout-driven synthesis. Today's 0.18-μm technology designs utilize some or all of the above techniques. While local routing within reasonably sized blocks has negligible wire delays, global routes between such blocks are closer to half a cycle. The cost of communication is becoming more explicit. Chips are partitioned early in the design process, and the delays of global lines are rolled into timing models.

As the complexity of digital systems has continued to increase, architects have responded to higher communication costs by further partitioning the internal microarchitecture and adding internal latency (internal pipe stages) in locations that they think will least damage machine performance. While some researchers imply that the delay of global wires sets cycle times [23], [47], [48], in high-performance machines this is clearly not the case; communication on these global wires is simply pipelined. This additional internal latency allows the machines to absorb the penalty of on-chip wires while still taking advantage of their offered high bandwidth. These added latencies are now visible to the user, but have small effects on the programming model. For example, in the Alpha 21 264 processor, the integer unit is partitioned into two clusters, and the latency for communicating between these clusters takes an additional cycle [51].

What will happen as on-chip wire delay takes multiple cycles is still an open question. A recent publication [52] gives a number of different visions of billion-transistor chips and shows the active debate in the computer architecture field about whether or not increasing communication costs can be hidden in machine microarchitecture. We believe that this will not be possible, and that more explicitly parallel machines, in which communication is expressed explicitly at the architectural level, will migrate on-chip.

Building machines that have better scaling properties is already an active area of research. These machines are often constructed from processing nodes that do not grow in complexity with technology. Instead, as technology scales, the number of these processing nodes on the chip grows, along with an on-chip communication network. The design

[9] As mentioned in Section III-A, the number of FO4s per cycle will be difficult to scale below 8. Thus it is likely that the scaling of FO4s per cycle will slow considerably from the current rate. This will further keep the change in the number of reachable gates per clock small.

of each processing node is similar to current designs where some attention is paid to communication issues, but the primary focus is still on functionality. However, at the chip level, the communication between the processing nodes is the main focus. This shift from monolithic architectures to more modular ones is also attractive from the standpoint of complexity management and design costs [53]. Examples of such research projects include the UC Berkeley IRAM and IDisk programs, where large servers are built from large numbers of small processors and disks [54], [55]; the MIT RAW project, which exposes the computation and communication costs directly to the compiler for it to schedule operations [56]; and the Stanford Smart Memories project, which is building a flexible collection of processing nodes, memory, and interconnect fabric to support a wider variety of programming models efficiently [53]. These are only a few of the many research projects in this area.

V. Conclusion

The view of wire scaling presented is not completely surprising; VLSI designers have long known that while local wires scale in performance, global and fixed-length wires do not. However, the CAD and architectural implications of wire scaling are often misunderstood or overlooked. Design tools and methodologies, despite the fact that scaled designs scale in performance, will still be pushed hard for improved long-wire-handling capabilities, or else the exponential scaling of chip complexity—and wire exceptions—will destroy design productivity. Architectures, rather than being pushed away from synchronous systems, will be driven toward explicit accounting for global latencies and modularity in computation and design.

In some ways the problem that future designers face is not that scaled wires are fundamentally bad, but that our expectations of wires are unreasonable. There is an effective signal speed for on-chip wires, and chip designers need to learn how to deal with it. Since wires have never been completely free at the board- or system-level, future chip design will be very similar to board-level design today; however, instead of dealing with chips on a board, we will be dealing with processing cores on a single die.

Acknowledgment

The authors would like to acknowledge useful discussions with J. Hutchby from the SRC, B. Havemann from Sematech, and D. Sylvester from Synopsys.

References

[1] *National Technology Roadmap for Semiconductors*: Semiconductor Industry Association, 1997.
[2] D. Sylvester *et al.*, "Getting to the bottom of deep submicron," in *Proc. ICCAD*, Nov. 1998, pp. 203–211.
[3] *International Technology Roadmap for Semiconductors*: Semiconductor Industry Association, 1999.
[4] T. Williams, private communication, 2000.
[5] M. Bohr, "Interconnect scaling—The real limiter to high performance ULSI," in *Proc. IEDM*, 1995, pp. 241–244.
[6] Y. Nishi, "The trend of on-chip interconnects: An international perspective," presented at the 1998 Spring Seminar Series, Stanford University.
[7] A. Ruehli, "Inductance calculations in a complex integrated circuit environment," *IBM J. Res. Dev.*, no. 5, pp. 470–481, Sept. 1972.
[8] E. Rosa, "The self and mutual inductance of linear conductors," *Bull. National Bureau of Standards*, pp. 301–344, 1908.
[9] M. Kamon *et al.*, "FASTHENRY: A multipole accelerated 3-D inductance extraction program," *IEEE Trans. Microwave Theory Tech.*, vol. 42, pp. 1750–1758, Sept. 1994.
[10] M. Beattie *et al.*, "IC analyses including extracted inductance models," in *Proc. DAC*, June 1999, pp. 915–920.
[11] K. Shepard *et al.*, "Return-limited inductance: A practical approach to on-chip inductance extraction," in *Proc. CICC*, May 1999, pp. 453–456.
[12] A. Deutsch *et al.*, "The importance of inductance and inductive coupling for on-chip wiring," in *Proc. EPEP*, Oct. 1997, pp. 53–56.
[13] P. J. Restle *et al.*, "Designing the best clock distribution network," in *VLSI Circuit Symp. Dig. Tech. Papers*, June 1998, pp. 2–6.
[14] B. Krauter *et al.*, "Including inductance effects in interconnect timing analysis," in *Proc. CICC*, May 1999, pp. 445–452.
[15] A. Vittal *et al.*, "Crosstalk reduction for VLSI," *IEEE Trans. Computer-Aided Design*, vol. 16, pp. 290–298, Mar. 1997.
[16] T. Sato *et al.*, "Accurate in-site measurement of peak noise and signal delay induced by interconnect coupling," in *ISSCC Dig. Tech. Papers*, Feb. 2000, pp. 226–227.
[17] K. Soumyanath *et al.*, "Accurate on-chip interconnect evaluation: A time-domain technique," *IEEE J. Solid-State Circuits*, vol. 34, pp. 623–631, May 1999.
[18] L. Pillage *et al.*, "Asymptotic waveform evaluation for timing analysis," *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 352–366, Apr. 1990.
[19] S. Morton, "On-chip inductance issues in multiconductor systems," in *Proc. DAC*, June 1999, pp. 921–926.
[20] S. Naffziger, "Design methodologies for interconnect in GHz+ ICs," in *Tutorial at ISSCC*, Feb. 1999.
[21] L. He *et al.*, "An efficient inductance modeling for on-chip interconnects," in *Proc. CICC*, May 1999, pp. 457–460.
[22] D. Priore, "Inductance on silicon for sub-micron CMOS VLSI," in *VLSI Circuits Symp. Dig. Tech. Papers*, June 1993, pp. 17–18.
[23] D. Miller, "Rationale and challenges for optical interconnections to electronic chips," Proc. IEEE, to be published.
[24] I. Sutherland *et al.*, *Logical Effort: Designing Fast CMOS Circuits*. San Mateo, CA: Morgan Kaufmann, Jan. 1999.
[25] H. Bakoglu, *Circuits, Interconnections and Packaging for VLSI*. Reading, MA: Addison-Wesley, 1990.
[26] P. Green, "A GHz IA-32 architecture microprocessor implemented on 0.18 mm technology with aluminum interconnect," in *ISSCC Dig. Tech. Papers*, Feb. 2000, pp. 98–99.
[27] *National Technology Roadmap for Semiconductors*: Semiconductor Industry Association, 1994.
[28] C. Hu, "CMOS Transistor scaling limits," *DAC 2000*. invited talk.
[29] T. Sorsch *et al.*, "Ultra-thin, 1.0–3.0nm, gate oxides for high performance sub-100nm technology," in *VLSI Technology Symp. Dig. Tech. Papers*, June 1998, pp. 215–216.
[30] I. C. Kizilyalli *et al.*, "Stacked gate dielectrics with TaO for future CMOS technologies," in *VLSI Technology Symp. Dig. Tech. Papers*, June 1998, pp. 216–217.
[31] M. Khare *et al.*, "Highly robust ultra-thin gate dielectric for giga scale technology," in *VLSI Technology Symp. Dig. Tech. Papers*, June 1998, pp. 218–219.
[32] T. Lin *et al.*, "A fully planarized 6-level-metal CMOS technology for 0.25-0.18 micron foundry manufacturing," in *Proc. IEDM*, Dec 1997, pp. 851–854.
[33] D. Harris, *Skew-Tolerant Circuit Design*. San Mateo, CA: Morgan Kaufman, 2000.
[34] B. Benschneider *et al.*, "A 1 GHz alpha microprocessor," in *ISSCC Dig. Tech. Papers*, Feb. 2000, pp. 86–87.
[35] [Online]. Available: www.intel.com/pressroom/archive/releases/dp031 000.htm
[36] T. Ghani *et al.*, "100 nm gate length high performance/low power CMOS transistor structure," in *Proc. IEDM*, Dec. 1999, pp. 415–419.
[37] W. Song *et al.*, "Power distribution techniques for VLSI circuits," in *Proc. Conf. Advance Research in VLSI*, Jan. 1984, pp. 45–52.
[38] P. Kapur, private communication, 2000.
[39] *XTK-TLC User's Manual*, 1996.

[40] Y. Massoud *et al.*, "Layout techniques for minimizing on-chip interconnect self inductance," in *Proc. DAC*, June 1998, pp. 566–571.
[41] M. Ang *et al.*, "An on-chip voltage regulator using switched decoupling capacitors," in *ISSCC Dig. Tech. Papers*, Feb. 2000, pp. 438–439.
[42] H. Kapadia *et al.*, "Using partitioning to help convergence in the standard-cell design automation methodology," in *Proc. DAC*, June 1999, pp. 592–597.
[43] W. Gosti *et al.*, "Wireplanning in logic synthesis," in *Proc. ICCAD*, Nov. 1998, p. 26.
[44] G. Stenz *et al.*, "Timing driven placement in interaction with netlist transformations," in *Proc. ISPD*, Apr. 1997, pp. 36–41.
[45] M. Lee *et al.*, "Incremental timing optimization for physical design by interacting logic restructuring and layout," in *Proc. ACM/IEEE Int. Workshop on Logic Synthesis*, May 1998, pp. 508–513.
[46] A. Salek *et al.*, "A DSM design flow: Putting floorplanning, technology mapping, and gate placement together," in *Proc. DAC*, June 1998, pp. 287–290.
[47] D. Sylvester *et al.*, "Getting to the bottom of deep submicron II: A global wiring paradigm," in *Proc. ISPD*, Apr. 1999, pp. 193–200.
[48] P. Fisher *et al.*, "Clock cycle estimation and test challenges for future microprocessors," Sematech Technology Transfer document #98033484A-TR, May 1998.
[49] V. Agarwal *et al.*, "Clock rate versus IPC: The end of the road for conventional microarchitectures," in *Proc. 27th Int. Symp. Computer Architecture*, June 2000, pp. 248–259.
[50] W. Dally *et al.*, "VLSI architecture: Past, present, and future," in *Proc. Conf. Advanced Research in VLSI*, Jan. 1999, pp. 232–241.
[51] B. A. Gieseke *et al.*, "A 600 MHz superscalar RISC microprocessor with out-of-order execution," in *ISSCC Dig. Tech. Papers*, Feb. 1997, pp. 176–177.
[52] *IEEE Computer, Special Issue: Future Microprocessors—How to Use a Billion Transistors*, Sept. 1997.
[53] K. Mai *et al.*, "Smart memories: A modular reconfigurable architecture," in *Proc. 27th Int. Symp. Computer Architecture*, June 2000, pp. 161–171.
[54] C. E. Kozyrakis *et al.*, "Scalable processors in the billion-transistor era: IRAM," *IEEE Computer*, pp. 75–78, Sept. 1997.
[55] K. Keeton *et al.*, "The intelligent disk (IDISK): A new computing infrastructure for decision support databases," presented at the National Storage Consortium's Network Attached Storage Device Working Group Meeting, June 8–9, 1998.
[56] E. Waingold *et al.*, "Baring it all to software: Raw machines," *IEEE Computer*, pp. 86–93, Sept. 1997.

Ron Ho (Member, IEEE) received the B.S. degree in electrical engineering and the A.B. degree in science, technology, and society in 1992, and the M.S. degree in electrical engineering in 1993, all from Stanford University, Stanford, CA. He is currently pursuing the Ph.D. degree in electrical engineering there.

In 1993, he joined Intel Corporation, where he has worked on microprocessor designs and design methodologies. His research interests are in high-performance and high-efficiency circuit design.

Mr. Ho is a member of Tau Beta Pi and Phi Beta Kappa. He was the 1992–1993 IEEE Fortescue Scholar.

Kenneth W. Mai (Student Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from Stanford University, Stanford, CA, in 1993 and 1997, respectively. He is pursuing the Ph.D. degree in electrical engineering at the same institution.

His research interests include low-power and high-performance circuit design.

Mr. Mai is a member of Tau Beta Pi and Phi Beta Kappa.

Mark A. Horowitz (Fellow, IEEE) received the B.S. and M.S. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, MA, and the Ph.D. degree from Stanford University, Stanford, CA.

He is Yahoo Founder's Professor of Electrical Engineering and Computer Sciences, and Director of the Computer Systems Laboratory at Stanford University. He is well-known for his research in integrated circuit design and VLSI systems. He is also co-founder of Rambus, Inc., Mountain View, CA. His current research includes multiprocessor design, low-power circuits, memory design, and high-speed links.

Dr. Horowitz received the Presidential Young Investigator Award and an IBM Faculty Development Award in 1985. In 1993, he was awarded Best Paper at the International Solid-State Circuits Conference.