Low-Power CMOS Digital Design

Anantha P. Chandrakasan, Samuel Sheng, and Robert W. Brodersen, *Fellow, IEEE*

Abstract—**Motivated by emerging battery-operated applications that demand intensive computation in portable environments, techniques are investigated which reduce power consumption in CMOS digital circuits while maintaining computational throughput. Techniques for low-power operation are shown which use the lowest possible supply voltage coupled with architectural, logic style, circuit, and technology optimizations. An architectural-based scaling strategy is presented which indicates that the optimum voltage is much lower than that determined by other scaling considerations. This optimum is achieved by trading increased silicon area for reduced power consumption.**

I. Introduction

With much of the research efforts of the past ten years directed toward increasing the speed of digital systems, present-day technologies possess computing capabilities that make possible powerful personal workstations, sophisticated computer graphics, and multimedia capabilities such as real-time speech recognition and real-time video. High-speed computation has thus become the expected norm from the average user, instead of being the province of the few with access to a powerful mainframe. Likewise, another significant change in the attitude of users is the desire to have access to this computation at any location, without the need to be physically tethered to a wired network. The requirement of portability thus places severe restrictions on size, weight, and power. Power is particularly important since conventional nickel-cadmium battery technology only provides 20 W · h of energy for each pound of weight [1]. Improvements in battery technology are being made, but it is unlikely that a dramatic solution to the power problem is forthcoming; it is projected that only a 30% improvement in battery performance will be obtained over the next five years [2].

Although the traditional mainstay of portable digital applications has been in low-power, low-throughput uses such as wristwatches and pocket calculators, there are an ever-increasing number of portable applications requiring low power and high throughput. For example, notebook and laptop computers, representing the fastest growing segment of the computer industry, are demanding the same computation capabilities as found in desktop machines. Equally demanding are developments in personal communications services (PCS's), such as the current generation of digital cellular telephony networks which employ complex speech compression algorithms and sophisticated radio modems in a pocket-sized device. Even more dramatic are the proposed future PCS applications, with universal portable multimedia access supporting full-motion digital video and control via speech recognition [3]. In these applications, not only will voice be transmitted via wireless inks, but data as well. This will facilitate new services such as multimedia database access (video and audio in addition to text) and supercomputing for simulation and design, through an intelligent network which allows communication with these services or other people at any place and time. Power for video compression and decompression and for speech recognition must be added to the portable unit to support these services—on top of the already lean power budget for the analog transceiver and speech encoding. Indeed, it is apparent that portability can no longer be associated with low throughput; instead, vastly increased capabilities, actually in excess of that demanded of fixed workstations, must be placed in a low-power portable environment.

Even when power is available in nonportable applications, the issue of low-power design is becoming critical. Up until now, this power consumption has not been of great concern, since large packages, cooling fins, and fans have been capable of dissipating the generated heat. However, as the density and size of the chips and systems continue to increase, the difficulty in providing adequate cooling might either add significant cost to the system or provide a limit on the amount of functionality that can be provided.

Thus, it is evident that methodologies for the design of high-throughput, low-power digital systems are needed. Fortunately, there are clear technological trends that give us a new degree of freedom, so that it may be possible to satisfy these seemingly contradictory requirements. Scaling of device feature sizes, along with the development of high-density, low-parasitic packaging, such as multichip modules [4]–[6], will alleviate the overriding concern with the numbers of transistors being used. When MOS technology has scaled to 0.2-μm minimum feature size it will be possible to place from 1 to 10×10^9 transistors in an area of 8 in $\times$ 10 in if a high-density packaging technology is used. The question then becomes how can this increased capability be used to meet a goal of low-power operation. Previous analyses on the question of how to best utilize increased transistor density at the chip level concluded that for high-performance microprocessors the best use is to provide increasing amounts of

Manuscript received September 4, 1991; revised November 18, 1991. This work was supported by DARPA.

The authors are with the Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA 94720.

IEEE Log Number 9105976.

on-chip memory [7]. It will be shown here that for computationally intensive functions that the best use is to provide additional circuitry to parallelize the computation.

Another important consideration, particularly in portable applications, is that many computation tasks are likely to be real-time; the radio modem, speech and video compression, and speech recognition all require computation that is always at near-peak rates. Conventional schemes for conserving power in laptops, which are generally based on power-down schemes, are not appropriate for these continually active computations. On the other hand, there is a degree of freedom in design that is available in implementing these functions, in that once the real-time requirements of these applications are met, there is no advantage in increasing the computational throughput. This fact, along with the availability of almost "limitless" numbers of transistors, allows a strategy to be developed for architecture design, which if it can be followed, will be shown to provide significant power savings.

II. Sources of Power Dissipation

There are three major sources of power dissipation in digital CMOS circuits, which are summarized in the following equation:

$$P_{\text{total}} = p_t\,(C_L \cdot V \cdot V_{dd} \cdot f_{\text{clk}}) + I_{sc} \cdot V_{dd} + I_{\text{leakage}} \cdot V_{dd}. \quad (1)$$

The first term represents the switching component of power, where C_L is the loading capacitance, f_{clk} is the clock frequency, and p_t is the probability that a power-consuming transition occurs (the activity factor). In most cases, the voltage swing V is the same as the supply voltage V_{dd}; however, in some logic circuits, such as in single-gate pass-transistor implementations, the voltage swing on some internal nodes may be slightly less [8]. The second term is due to the direct-path short circuit current I_{sc}, which arises when both the NMOS and PMOS transistors are simultaneously active, conducting current directly from supply to ground [9], [10]. Finally, leakage current I_{leakage}, which can arise from substrate injection and subthreshold effects, is primarily determined by fabrication technology considerations [11] (see Section III-C). The dominant term in a "well-designed" circuit is the switching component, and low-power design thus becomes the task of minimizing p_t, C_L, V_{dd}, and f_{clk}, while retaining the required functionality.

The power–delay product can be interpreted as the amount of energy expended in each switching event (or transition) and is thus particularly useful in comparing the power dissipation of various circuit styles. If it is assumed that only the switching component of the power dissipation is important, then it is given by

$$\text{energy per transition} = P_{\text{total}}/f_{\text{clk}} = C_{\text{effective}}\,V_{dd}^2 \quad (2)$$

where $C_{\text{effective}}$ is the effective capacitance being switched to perform a computation and is given by $C_{\text{effective}} = p_t \cdot C_L$.

III. Circuit Design and Technology Considerations

There are a number of options available in choosing the basic circuit approach and topology for implementing various logic and arithmetic functions. Choices between static versus dynamic implementations, pass-gate versus conventional CMOS logic styles, and synchronous versus asynchronous timing are just some of the options open to the system designer. At another level, there are also various architectural/structural choices for implementing a given logic function; for example, to implement an adder module one can utilize a ripple-carry, carry-select, or carry-lookahead topology. In this section, the trade-offs with respect to low-power design between a selected set of circuit approaches will be discussed, followed by a discussion of some general issues and factors affecting the choice of logic family.

A. Dynamic Versus Static Logic

The choice of using static or dynamic logic is dependent on many criteria than just its low-power performance, e.g., testability and ease of design. However, if only the low-power performance is analyzed it would appear that dynamic logic has some inherent advantages in a number of areas including reduced switching activity due to hazards, elimination of short-circuit dissipation, and reduced parasitic node capacitances. Static logic has advantages since there is no precharge operation and charge sharing does not exist. Below, each of these considerations will be discussed in more detail.

1) Spurious Transitions: Static designs can exhibit spurious transitions due to finite propagation delays from one logic block to the next (also called critical races and dynamic hazards [12]), i.e., a node can have multiple transitions in a single clock cycle before settling to the correct logic level. For example, consider a static N-bit adder, with all bits of the summands going from ZERO to ONE, with the carry input set to ZERO. For all bits, the resultant sum should be ZERO; however, the propagation of the carry signal causes a ONE to appear briefly at most of the outputs. These spurious transitions dissipate extra power over that strictly required to perform the computation. The number of these extra transitions is a function of input patterns, internal state assignment in the logic design, delay skew, and logic depth. To be specific about the magnitude of this problem, an 8-b ripple-carry adder with a uniformly distributed set of random input patterns will typically consume an extra 30% in energy. Though it is possible with careful logic design to eliminate these transitions, dynamic logic intrinsically does not have this problem, since any node can undergo at most one power-consuming transition per clock cycle.

2) Short-Circuit Currents: Short-circuit (direct-path) currents, I_{sc} in (1), are found in static CMOS circuits.

However, by sizing transistors for equal rise and fall times, the short-circuit component of the total power dissipated can be kept to less than 20% [9] (typically < 5–10%) of the dynamic switching component. Dynamic logic does not exhibit this problem, except for those cases in which static pull-up devices are used to control charge sharing [13] or when clock skew is significant.

3) Parasitic Capacitance: Dynamic logic typically uses fewer transistors to implement a given logic function, which directly reduces the amount of capacitance being switched and thus has a direct impact on the power–delay product [14], [15]. However, extra transistors may be required to insure that charge sharing does not result in incorrect evaluation.

4) Switching Activity: The one area in which dynamic logic is at a distinct disadvantage is in its necessity for a precharge operation. Since in dynamic logic every node must be precharged every clock cycle, this means that some nodes are precharged only to be immediately discharged again as the node is evaluated, leading to a higher activity factor. If a two-input N-tree (precharged high) dynamic NOR gate has a uniform input distribution of high and low levels, then the four possible input combinations (00,01,10,11) will be equally likely. There is then a 75% probability that the output node will discharge immediately after the precharge phase, implying that the activity for such a gate is 0.75 (i.e., $P_{\text{NOR}} = 0.75\, C_L V_{dd}^2 f_{\text{clk}}$). On the other hand, the activity factor for the static NOR counterpart will be only 3/16, excluding the component due to the spurious transitions mentioned in Section III-A-1 (power is only drawn on a ZERO-to-ONE transition, so $p_{0\to 1} = p(0)p(1) = p(0)\,(1 - p(0))$). In general, gate activities will be different for static and dynamic logic and will depend on the type of operation being performed and the input signal probabilities. In addition, the clock buffers to drive the precharge transistors will also require power that it not needed in a static implementation.

5) Power-Down Modes: Lastly, power-down techniques achieved by disabling the clock signal have been used effectively in static circuits, but are not as well-suited for dynamic techniques. If the logic state is to be preserved during shutdown, a relatively small amount of extra circuitry must be added to the dynamic circuits to preserve the state, resulting in a slight increase in parasitic capacitance and slower speeds.

B. Conventional Static Versus Pass-Gate Logic

A more clear situation exists in the use of transfer gates to implement logic functions, as is used in the complementary pass-gate logic (CPL) family [8], [10]. In Fig. 1, the schematic of a typical static CMOS logic circuit for a full adder is shown along with a static CPL version [8]. The pass-gate design uses only a single transmission NMOS gate, instead of a full complementary pass gate to reduce node capacitance. Pass-gate logic is attractive as fewer transistors are required to implement important logic functions, such as XOR's which only require two pass transistors in a CPL implementation. This particularly efficient implementation of an XOR is important since it is key to most arithmetic functions, permitting adders and multipliers to be created using a minimal number of devices. Likewise, multiplexers, registers, and other key building blocks are simplified using pass-gate designs.

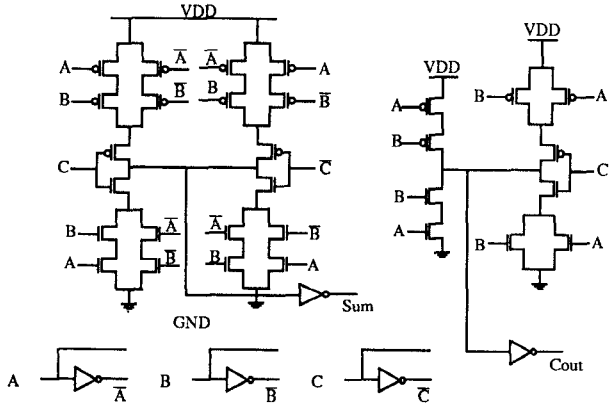


Fig. 1. Comparison of a conventional CMOS and CPL adders [8].

However, a CPL implementation as shown in Fig. 1 has two basic problems. First, the threshold drop across the single-channel pass transistors results in reduced current drive and hence slower operation at reduced supply voltages; this is important for low-power design since it is desirable to operate at the lowest possible voltages levels. Second, since the "high" input voltage level at the regenerative inverters is not V_{dd}, the PMOS device in the inverter is not fully turned off, and hence direct-path static power dissipation could be significant. To solve these problems, reduction of the threshold voltage has proven effective, although if taken too far will incur a cost in dissipation due to subthreshold leakage (see Section III-C) and reduced noise margins. The power dissipation for a pass-gate family adder with zero-threshold pass transistors at a supply voltage of 4 V was reported to be 30% lower than a conventional static design, with the differ-

ence being even more significant at lower supply voltages [8].

C. Threshold Voltage Scaling

Since a significant power improvement can be gained through the use of low-threshold MOS devices, the question of how low the thresholds can be reduced must be addressed. The limit is set by the requirement to retain adequate noise margins and the increase in subthreshold currents. Noise margins will be relaxed in low-power designs because of the reduced currents being switched, however, the subthreshold currents can result in significant static power dissipation. Essentially, subthreshold leakage occurs due to carrier diffusion between the source and the drain when the gate–source voltage V_{gs} has exceeded the weak inversion point, but is still below the threshold voltage V_t, where carrier drift is dominant. In this regime, the MOSFET behaves similarly to a bipolar transistor, and the subthreshold current is exponentially dependent on the gate–source voltage V_{gs}, and approximately independent of the drain–source voltage V_{ds}, for V_{ds} approximately larger than 0.1 V. Associated with this is the subthreshold slope S_{th}, which is the amount of voltage required to drop the subthreshold current by one decade. At room temperature, typical values for S_{th} lie between 60 and 90 mV/decade current, with 60 mV/decade being the lower limit. Clearly, the lower S_{th} is, the better, since it is desirable to have the device "turn off" as close to V_t as possible. As a reference, for an $L = 1.5$-μm, $W = 70$-μm NMOS device, at the point where V_{gs} equals V_t, with V_t defined as where the surface inversion charge density is equal to the bulk doping, approximately 1 μA of leakage current is exhibited, or 0.014 μA/μm of gate width [16]. The issue is whether this extra current is negligible in comparison to the time-average current during switching. For a CMOS inverter (PMOS: $W = 8$ μm, NMOS: $W = 4$ μm), the current was measured to be 64 μA over 3.7 ns at a supply voltage of 2 V. This implies that there would be a 100% power penalty for subthreshold leakage if the device were operating at a clock speed of 25 MHz with an activity factor of $p_t = 1/6$th, i.e., the devices were left idle and leaking current 83% of the time. It is not advisable, therefore, to use a true zero threshold device, but instead to use thresholds of at least 0.2 V, which provides for at least two orders of magnitude of reduction of subthreshold current. This provides a good compromise between improvement of current drive at low supply voltage operation and keeping subthreshold power dissipation to a negligible level. This value may have to be higher in dynamic circuits to prevent accidental discharge during the evaluation phase [11]. Fortunately, device technologists are addressing the problem of subthreshold currents in future scaled technologies, and reducing the supply voltages also serves to reduce the current by reducing the maximum allowable drain–source voltage [17], [18]. The design of future circuits for lowest power operation should therefore explicitly take into account the effect of subthreshold currents.

D. Power-Down Strategies

In synchronous designs, the logic between registers is continuously computing every clock cycle based on its new inputs. To reduce the power in synchronous designs, it is important to minimize switching activity by powering down execution units when they are not performing "useful" operations. This is an important concern since logic modules can be switching and consuming power even when they are not being actively utilized [19].

While the design of synchronous circuits requires special design effort and power-down circuitry to detect and shut down unused units, self-timed logic has inherent power-down of unused modules, since transitions occur only when requested. However, since self-timed implementations require the generation of a completion signal indicating the outputs of the logic module are valid, there is additional overhead circuitry. There are several circuit approaches to generate the requisite completion signal. One method is to use dual-rail coding, which is implicit in certain logic families such as the DCVSL [13], [20]. The completion signal in a combinational macrocell made up of cascading DCVSL gates consists of simply ORing the outputs of only the last gate in the chain, leading to small overhead requirements. However, for each computation, dual-rail coding guarantees a switching event will occur since at least one of the outputs must evaluate to zero. We found that the dual-rail DCVSL family consumes at least two times more in energy per input transition than a conventional static family. Hence, self-timed implementations can prove to be expensive in terms of energy for data paths that are continuously computing.

IV. Voltage Scaling

Thus far we have been primarily concerned with the contributions of capacitance to the power expression CV^2f. Clearly, though, the reduction of V should yield even greater benefits; indeed, reducing the supply voltage is the key to low-power operation, even after taking into account the modifications to the system architecture, which is required to maintain the computational throughput. First, a review of circuit behavior (delay and energy characteristics) as a function of scaling supply voltage and feature sizes will be presented. By comparison with experimental data, it is found that simple first-order theory yields an amazingly accurate representation of the various dependencies over a wide variety of circuit styles and architectures. A survey of two previous approaches to supply-voltage scaling is then presented, which were focused on maintaining reliability and performance. This is followed by our architecture-driven approach, from which an "optimal" supply voltage based on technology, architecture, and noise margin constraints is derived.

A. Impact on Delay and Power–Delay Product

As noted in (2), the energy per transition or equivalently the power–delay product in "properly designed" CMOS circuits (as discussed in Section II) is proportional

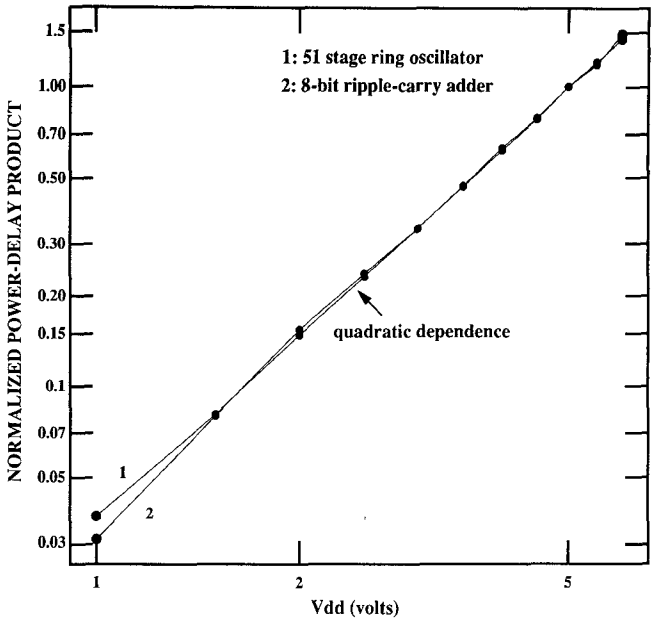


Fig. 2. Power-delay product exhibiting square-law dependence for two different circuits.

to V^2. This is seen from Fig. 2, which is a plot of two experimental circuits that exhibit the expected V^2 dependence. Therefore, it is only necessary to reduce the supply voltage for a *quadratic* improvement in the power-delay product of a logic family.

Unfortunately, this simple solution to low-power design comes at a cost. As shown in Fig. 3, the effect of reducing V_{dd} on the delay is shown for a variety of different logic circuits that range in size from 56 to 44 000 transistors spanning a variety of functions; all exhibit essentially the same dependence (see Table I). Clearly, we pay a speed penalty for a V_{dd} reduction, with the delays drastically increasing as V_{dd} approaches the sum of the threshold voltages of the devices. Even though the exact analysis of the delay is quite complex if the nonlinear characteristic of a CMOS gate are taken into account, it is found that a simple first-order derivation adequately predicts the experimentally determined dependence and is given by

$$T_d = \frac{C_L \times V_{dd}}{I} = \frac{C_L \times V_{dd}}{\mu C_{ox}\,(W/L)\,(V_{dd} - V_t)^2}. \quad (3)$$

We also evaluated (through experimental measurements and SPICE simulations) the energy and delay performance for several different logic styles and topologies using an 8-b adder as a reference; the results are shown on a log–log plot in Fig. 4. We see that the power–delay product improves as delays increase (through reduction of the supply voltage), and therefore it is desirable to operate at the *slowest* possible speed. Since the objective is to reduce power consumption while maintaining the overall system throughput, compensation for these increased delays at low voltages is required. Of particular interest in this figure is the range of energies required for a transition at a given amount of delay. The best logic family we ana-

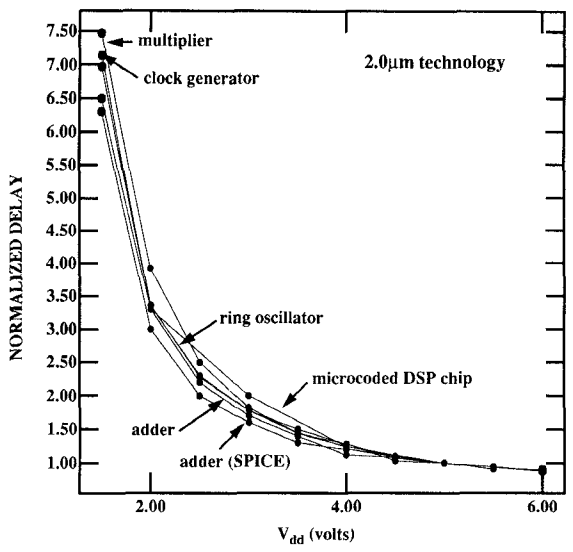


Fig. 3. Data demonstrating delay characteristics follow simple first-order theory.

TABLE I
DETAILS OF COMPONENTS USED FOR THE STUDY IN FIG. 3

Component (all in 2 μm)	# of Transistors	Area	Comments
Microcoded DSP Chip [21]	44 802	94 mm^2	20-b data path
Multiplier	20 432	12.2 mm^2	24 × 24 b
Adder	256	0.083 mm^2	conventional static
Ring Oscillator	102	0.055 mm^2	51 stages
Clock Generator	56	0.04 mm^2	cross-coupled NOR

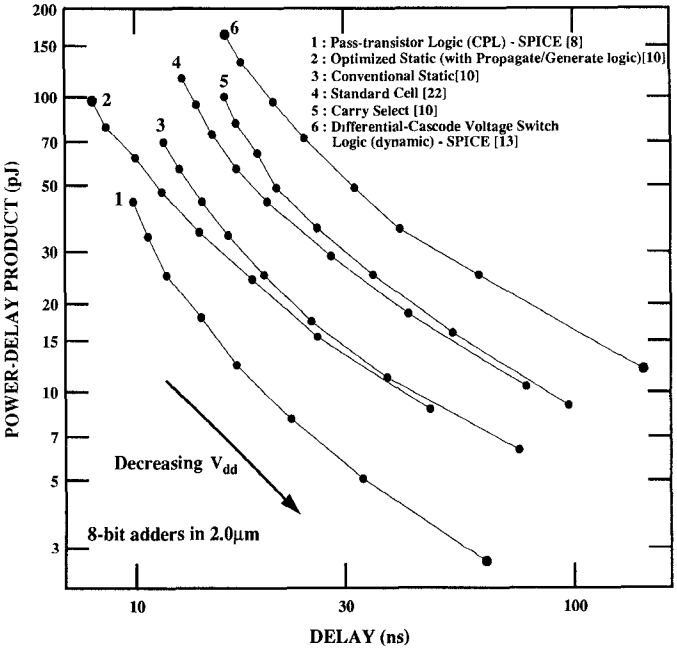


Fig. 4. Data showing improvement in power-delay product at the cost of speed for various circuit approaches.

lyzed (over 10 times better than the worst that we investigated) was the pass-gate family, CPL, (see Section III-B) if a reduced value for the threshold is assumed [8].

Figs. 2, 3, and 4 suggest that the delay and energy behavior as a function of V_{dd} scaling for a given technology is "well-behaved" and relatively independent of logic style and circuit complexity. We will use this result during our optimization of architecture for low-power by treating V_{dd} as a free variable and by allowing the architectures to vary to retain constant throughput. By exploiting the monotonic dependencies of delay and energy versus supply voltage that hold over wide circuit variations, it is possible to make relatively strong predictions about the types of architectures that are best for low-power design. Of course, as mentioned previously, there are some logic styles such as NMOS pass-transistor logic without reduced thresholds whose delay and energy characteristics would deviate from the ones presented above, but even for these cases, though the quantitative results will be different, the basic conclusions will still hold.

B. Optimal Transistor Sizing with Voltage Scaling

Independent of the choice of logic family or topology, optimized transistor sizing will play an important role in reducing power consumption. For low power, as is true for high-speed design, it is important to equalize all delay paths so that a single critical path does not unnecessarily limit the performance of the entire circuit. However, beyond this constraint, there is the issue of what extent the W/L ratios should be uniformly raised for all the devices, yielding a uniform decrease in the gate delay and hence allowing for a corresponding reduction in voltage and power. It is shown in this section that if voltage is allowed to vary, that the optimal sizing for low-power operation is quite different from that required for high speed.

In Fig. 5, a simple two-gate circuit is shown, with the first stage driving the gate capacitance of the second, in addition to the parasitic capacitance C_p due to substrate coupling and interconnect. Assuming that the input gate capacitance of both stages is given by NC_{ref}, where C_{ref} represents the gate capacitance of a MOS device with the smallest allowable W/L, then the delay through the first gate at a supply voltage V_{ref} is given by

$$T_N = K\frac{(C_p + NC_{\text{ref}})}{(NC_{\text{ref}})}\frac{V_{\text{ref}}}{(V_{\text{ref}} - V_t)^2}$$

$$= K(1 + \alpha/N)\frac{V_{\text{ref}}}{(V_{\text{ref}} - V_t)^2} \tag{4}$$

where α is defined as the ratio of C_p to C_{ref}, and K represents terms independent of device width and voltage. For a given supply voltage V_{ref}, the speedup of a circuit whose W/L ratios are sized up by a factor of N over a reference circuit using minimum-size transistors ($N = 1$) is given by $(1 + \alpha/\text{N})/(1 + \alpha)$. In order to evaluate the energy performance of the two designs at the same speed, the voltage of the scaled solution is allowed to vary as to keep delay constant. Assuming that the delay scales as $1/V_{dd}$ (ignoring threshold voltage reductions in signal swings), the supply voltage V_N, where the delays of the scaled design and the reference design are equal, is given by

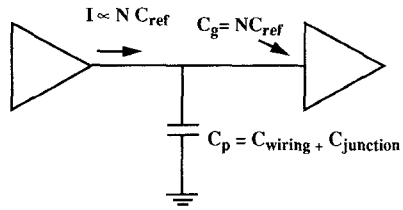


Fig. 5. Circuit model for analyzing the effect of transistor sizing.

$$V_N = \frac{(1 + \alpha/N)}{(1 + \alpha)}V_{\text{ref}}. \tag{5}$$

Under these conditions, the energy consumed by the first stage as a function of N is given by

$$\text{Energy}(N) = (C_p + NC_{\text{ref}})V_N^2$$

$$= \frac{NC_{\text{ref}}(1 + \alpha/N)^3 V_{\text{ref}}^2}{(1 + \alpha)^2}. \tag{6}$$

After normalizing against E_{ref} (the energy for the minimum size case), Fig. 6 shows a plot of Energy (N)/Energy (1) versus N for various values of α. When there is no parasitic capacitance contribution (i.e., $\alpha = 0$), the energy increases linearly with respect to N, and the solution utilizing devices with the smallest W/L ratios results in the lowest power. At high values of α, when parasitic capacitances begin to dominate over the gate capacitances, the power decreases temporarily with increasing device sizes and then starts to increase, resulting in an optimal value for N. The initial decrease in supply voltage achieved from the reduction in delays more than compensates the increase in capacitance due to increasing N. However, after some point the increase in capacitance dominates the achievable reduction in voltage, since the incremental speed increase with transistor sizing is very small (this can be seen in (4), with the delay becoming independent of α as N goes to infinity). Throughout the analysis we have assumed that the parasitic capacitance is independent of device sizing. However, the drain and source diffusion and perimeter capacitances actually increase with increasing area, favoring smaller size devices and making the above a worst-case analysis.

Also plotted in Fig. 6 are simulation results from extracted layouts of an 8-b adder carry chain for three different device W/L ratios ($N = 1$, $N = 2$, and $N = 4$). The curve follows the simple first-order model derived very well, and suggests that this example is dominated more by the effect of gate capacitance rather than parasitics. In this case, increasing devices W/L's does not help, and the solution using the smallest possible W/L ratios results in the best sizing.

From this section, it is clear that the determination of

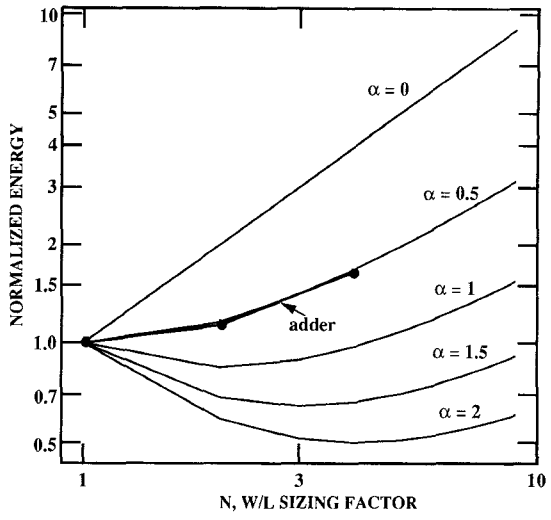


Fig. 6. Plot of energy versus transistor sizing factor for various parasitic contributions.

an "optimal" supply voltage is the key to minimizing power consumption; hence we focus on this issue in the following sections. First, we will review the previous work dealing with choice of supply voltage which were based on reliability and speed considerations [23], [24], followed by an architecturally driven approach to supply voltage scaling.

C. Reliability-Driven Voltage Scaling

One approach to the selection of an optimal power supply voltage for deep-submicrometer technologies is based on optimizing the trade-off between speed and reliability [23]. Constant-voltage scaling, the most commonly used technique, results in higher electric fields that create hot carriers. As a result of this, the devices degrade with time (including changes in threshold voltages, degradation of transconductance, and increase in subthreshold currents), leading to eventual breakdown [11]. One solution to reducing the number of hot carriers is to change the physical device structure, such as the use of lightly doped drain (LDD), usually at the cost of decreased performance. Assuming the use of an LDD structure and a constant hot-carrier margin, an optimal voltage of 2.5 V was found for a 0.25-μm technology by choosing the minimum point on the delay versus V_{dd} curve [23]. For voltages above this minimum point, the delay was found to increase with increasing V_{dd}, since the LDD structure used for the purposes of reliability resulted in increased parasitic resistances.

D. Technology-Driven Voltage Scaling

The simple first-order delay analysis presented in Section IV-A is reasonably accurate for long-channel devices. However, as feature sizes shrink below 1.0 μm, the delay characteristics as a function of lowering the supply voltage deviate from the first-order theory presented since it does not consider carrier velocity saturation under high electric fields [11]. As a result of velocity saturation, the current is no longer a quadratic function of the voltage but linear; hence, the current drive is significantly reduced and is approximately given by $I = WC_{ox}(V_{dd} - V_t)v_{\max}$ [4]. Given this and the equation for delay in (3), we see that the delay for submicrometer circuits is relatively independent of supply voltages at high electric fields.

A "technology"-based approach proposes choosing the power supply voltage based on maintaining the speed performance for a given submicrometer technology [24]. By exploiting the relative independence of delay on supply voltage at high electric fields, the voltage can be dropped to some extent for a velocity-saturated device with very little penalty in speed performance. This implies that there is little advantage to operating above a certain voltage. This idea has been formalized by Kakumu and Kinugawa, yielding the concept of a "critical voltage" which provides a lower limit on the supply voltage [24]. The critical voltage is defined as $V_c = 1.1E_cL_{\text{eff}}$, where E_c is the critical electric field causing velocity saturation; this is the voltage at which the delay versus V_{dd} curve approaches a $\sqrt{V_{dd}}$ dependence. For 0.3-μm technology, the proposed lower limit on supply voltage (or the critical voltage) was found to be 2.43 V.

Because of this effect, there is some movement to a 3.3-V industrial voltage standard since at this level of voltage reduction there is not a significant loss of circuit speed [1], [25]. This was found to achieve a 60% reduction in power when compared to a 5-V operation [25].

E. Architecture-Driven Voltage Scaling

The above-mentioned "technology"-based approaches are focusing on reducing the voltage while maintaining device speed, and are not attempting to achieve the minimum possible power. As shown in Figs. 2 and 4, CMOS logic gates achieve lower power–delay products (energy per computation) as the supply voltages are reduced. In fact, once a device is in velocity saturation there is a further degradation in the energy per computation, so in minimizing the energy required for computation, Kakumu and Kinugawa's critical voltage provides an *upper* bound on the supply voltage (whereas for their analysis it provided a *lower* bound!). It now will be the task of the architecture to compensate for the reduced circuit speed that comes with operating below the critical voltage.

To illustrate how architectural techniques can be used to compensate for reduced speeds, a simple 8-b data path consisting of an adder and a comparator is analyzed assuming a 2.0-μm technology. As shown in Fig. 7, inputs A and B are added, and the result compared to input C. Assuming the worst-case delay through the adder, comparator, and latch is approximately 25 ns at a supply voltage of 5 V, the system in the best case can be clocked with a clock period of $T = 25$ ns. When required to run at this maximum possible throughput, it is clear that the operating voltage cannot be reduced any further since no extra delay can be tolerated, hence yielding no reduction in power. We will use this as the reference data path for

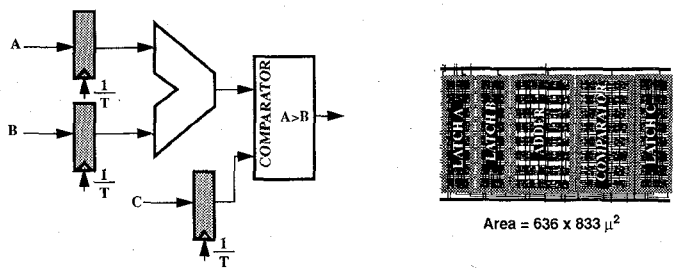


Fig. 7. A simple data path with corresponding layout.

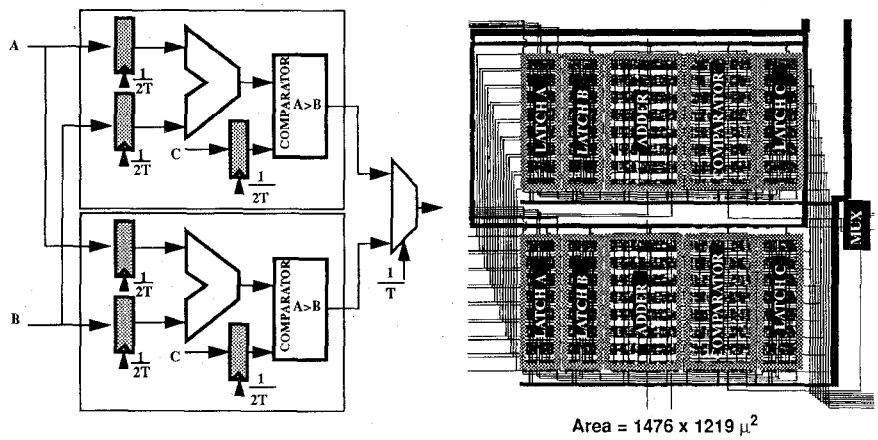


Fig. 8. Parallel implementation of the simple data path.

our architectural study and present power improvement numbers with respect to this reference. The power for the reference data path is given by

$$P_{\text{ref}} = C_{\text{ref}} V_{\text{ref}}^2 f_{\text{ref}} \tag{7}$$

where C_{ref} is the total effective capacitance being switched per clock cycle. The effective capacitance was determined by averaging the energy over a sequence of input patterns with a uniform distribution.

One way to maintain throughput while reducing the supply voltage is to utilize a parallel architecture. As shown in Fig. 8, two identical adder–comparator data paths are used, allowing each unit to work at half the original rate while maintaining the original throughput. Since the speed requirements for the adder, comparator, and latch have decreased from 25 to 50 ns, the voltage can be dropped from 5 to 2.9 V (the voltage at which the delay doubled, from Fig. 3). While the data-path capacitance has increased by a factor of 2, the operating frequency has correspondingly decreased by a factor of 2. Unfortunately, there is also a slight increase in the total "effective" capacitance introduced due to the extra routing, resulting in an increased capacitance by a factor of 2.15. Thus the power for the parallel data path is given by

$$P_{\text{par}} = C_{\text{par}} V_{\text{par}}^2 f_{\text{par}}$$

$$= (2.15C_{\text{ref}})\,(0.58V_{\text{ref}})^2 \left(\frac{f_{\text{ref}}}{2}\right) \approx 0.36\, P_{\text{ref}}. \tag{8}$$

This method of reducing power by using parallelism has the overhead of increased area, and would not be suitable for area-constrained designs. In general, the parallelism will have the overhead of extra routing (and hence extra power), and careful optimization must be performed to minimize this overhead (for example, partitioning techniques for minimal overhead). Interconnect capacitance will especially play a very important role in deep-submicrometer implementations, since the fringing capacitance of the interconnect capacitance ($C_{\text{wiring}} = C_{\text{area}} + C_{\text{fringing}} + C_{\text{wiring}}$) can become a dominant part of the total capacitance (equal to $C_{\text{gate}} + C_{\text{junction}} + C_{\text{wiring}}$) and cease to scale [4].

Another possible approach is to apply pipelining to the architecture, as shown in Fig. 9. With the additional pipeline latch, the critical path becomes $\max[T_{\text{adder}}, T_{\text{comparator}}]$, allowing the adder and the comparator to operate at a slower rate. For this example, the two delays are equal, allowing the supply voltage to again be reduced from 5 V used in the reference data path to 2.9 V (the voltage at which the delay doubles) with no loss in throughput. However, there is a much lower area overhead incurred by this technique, as we only need to add pipeline registers. Note that there is again a slight increase in hardware due to the extra latches, increasing the "effective" capacitance by approximately a factor of 1.15. The power consumed by the pipelined data path is

$$P_{\text{pipe}} = C_{\text{pipe}} V_{\text{pipe}}^2 f_{\text{pipe}}$$

$$= (1.15C_{\text{ref}})\,(0.58V_{\text{ref}})^2 f_{\text{ref}} \approx 0.39\, P_{\text{ref}}. \tag{9}$$

With this architecture, the power reduces by a factor of approximately 2.5, providing approximately the same

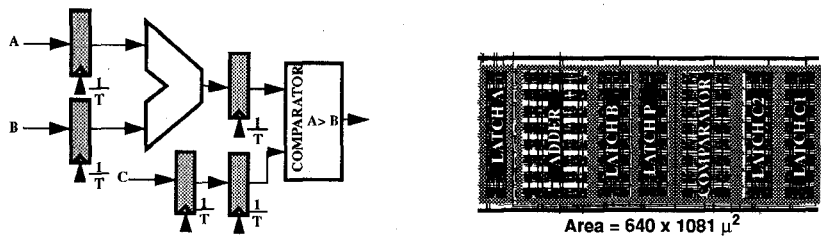


Fig. 9. Pipelined implementation of the simple data path.

TABLE II
ARCHITECTURE SUMMARY

Architecture Type	Voltage	Area	Power
Simple data path (no pipelining or parallelism)	5 V	1	1
Pipelined data path	2.9 V	1.3	0.39
Parallel data path	2.9 V	3.4	0.36
Pipeline parallel	2.0 V	3.7	0.2

power reduction as the parallel case with the advantage of lower area overhead. As an added bonus, increasing the level of pipelining also has the effect of reducing logic depth and hence power contributed due to hazards and critical races (see Section III-A-1).

Furthermore, an obvious extension is to utilize a combination of pipelining and parallelism. Since this architecture reduces the critical path and hence speed requirement by a factor of 4, the voltage can be dropped until the delay increases by a factor of 4. The power consumption in this case is

$$P_{\text{parpipe}} = C_{\text{parpipe}} V^2_{\text{parpipe}} f_{\text{parpipe}} = (2.5C_{\text{ref}})(0.4V_{\text{ref}})^2 \left(\frac{f_{\text{ref}}}{2}\right) \approx 0.2P_{\text{ref}}. \quad (10)$$

The parallel-pipeline implementation results in a 5 times reduction in power. Table II shows a comparative summary of the various architectures described for the simple adder–comparator data path.

From the above examples, it is clear that the traditional time-multiplexed architectures, as used in general-purpose microprocessors and DSP chips, are the least desirable for low-power applications. This follows since time multiplexing actually increases the speed requirements on the logic circuitry, thus not allowing reduction in the supply voltage.

V. Optimal Supply Voltage

In the previous section, we saw that the delay increase due to reduced supply voltages below the critical voltage can be compensated by exploiting parallel architectures. However, as seen in Fig. 3 and (3), as supply voltages approach the device thresholds, the gate delays increase rapidly. Correspondingly, the amount of parallelism and overhead circuitry increases to a point where the added overhead dominates any gains in power reduction from further voltage reduction, leading to the existence of an "optimal" voltage from an architectural point of view. To determine the value of this voltage, the following model is used for the power for a fixed system throughput as a function of voltage (and hence degree of parallelism):

$$\text{Power}(N) = NC_{\text{ref}} V^2 \frac{f_{\text{ref}}}{N} + C_{ip} V^2 \frac{f_{\text{ref}}}{N} + C_{\text{interface}} V^2 f_{\text{ref}} \quad (11)$$

where N is the number of parallel processors, C_{ref} is the capacitance of a single processor, C_{ip} is the interprocessor communication overhead introduced due to the parallelism (due to control and routing), and $C_{\text{interface}}$ is the overhead introduced at the interface which is not decreased in speed as the architecture is made more parallel. In general, C_{ip} and $C_{\text{interface}}$ are functions of N, and the power improvement over the reference case (i.e., without parallelism) can be expressed as

$$P_{\text{normalized}} = \left(1 + \frac{C_{ip}(N)}{NC_{\text{ref}}} + \frac{C_{\text{interface}}(N)}{C_{\text{ref}}}\right)\left(\frac{V}{V_{\text{ref}}}\right)^2. \quad (12)$$

At very low supply voltages (near the device thresholds), the number of processors (and hence the corresponding overhead in the above equation) typically increases at a faster rate than the V^2 term decreases, resulting in a power increase with further reduction in voltage.

Reduced threshold devices tend to lower the optimal voltage; however, as seen in Section III-C, at thresholds below 0.2 V, power dissipation due to the subthreshold current will soon start to dominate and limit further power improvement. An even lower bound on the power supply voltage for a CMOS inverter with "correct" functionality was found to be 0.2 V ($8kT/q$) [26]. This gives a limit on the power–delay product that can be achieved with CMOS digital circuits; however, the amount of parallelism to retain throughput at this voltage level would no doubt be prohibitive for any practical situation.

So far, we have seen that parallel and pipelined architectures can allow for a reduction in supply voltages to the "optimal" level; this will indeed be the case if the algorithm being implemented does not display any recursion (feedback). However, there are a wide class of applications inherently recursive in nature, ranging from the simple ones, such as infinite impulse response and adap-

tive filters, to more complex cases such as systems solving nonlinear equations and adaptive compression algorithms. There is, therefore, also an algorithmic bound on the level to which pipelining and parallelism can be exploited for voltage reduction. Although the application of data control flowgraph transformations can alleviate this bottleneck to some extent, both the constraints on latency and the structure of computation of some algorithms can prevent voltage reduction to the optimal voltage levels discussed above [27], [28].

Another constraint on the lowest allowable supply voltages is set by system noise margin constraints ($V_{\text{noise margin}}$). Thus, we must lower-bound the "optimal" voltage by

$$V_{\text{noise margin}} \leq V_{\text{optimal}} \leq V_{\text{critical}} \tag{13}$$

with V_{critical} defined in Section IV-D. Hence, the "optimal" supply voltage (for a fixed technology) will lie somewhere between the voltage set by noise margin constraints and the critical voltage.

Fig. 10 shows power (normalized to 1 at $V_{dd} = 5$ V) as a function of V_{dd} for a variety of cases for a 2.0-μm technology. As will be shown, there is a wide variety of assumptions in these various cases and it is important to note that they all have roughly the same optimum value of supply voltage, approximately 1.5 V. Curve 1 in this figure represents the power dissipation which would be achieved if there were no overhead associated with increased parallelism. For this case, the power is a strictly decreasing function of V_{dd} and the optimum voltage would be set by the minimum value allowed from noise margin constraints (assuming that no recursive bottleneck was reached). Curve 5 assumes that the interprocessor capacitance has an N^2 dependence while curve 6 assumes an N^3 dependence. It is expected that in most practical cases the dependence is actually less than N^2, but even with the extremely strong N^3 dependence an optimal value around 2 V is found.

Curves 2 and 3 are obtained from data from actual layouts [22], and exhibit a dependence of the interface capacitance which lies between linear and quadratic on the degree of parallelism, N. For these cases, there was no interprocessor communication. Curves 2 and 3 are extensions of the example described in Section IV-E in which the parallel and parallel-pipeline implementations of the simple data path were duplicated N times. Curve 4 is for a much more complex example, a seventh-order IIR filter, also obtained from actual layout data. The overhead in this case arose primarily from interprocessor communication. This curve terminates around 1.4 V, because at that point the algorithm has been made maximally parallel, reaching a recursive bottleneck. For this case, at a supply voltage of 5 V, the architecture is basically a single hardware unit that is being time multiplexed and requires about 7 times more power than the optimal parallel case which is achieved with a supply of around 1.5 V. Table III summarizes the power reduction and normalized areas that were obtained from layouts. The increase in

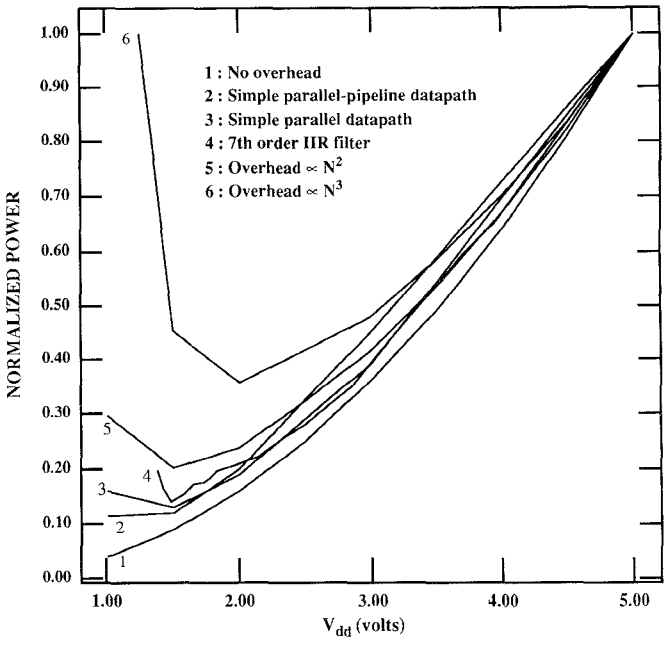


Fig. 10. Optimum voltage of operation.

TABLE III
NORMALIZED AREA/POWER FOR VARIOUS SUPPLY VOLTAGE FOR PLOTS 2, 3, AND 4 IN FIG. 10

Voltage	Parallel Area/Power	Parallel-Pipeline Area/Power	IIR Area/Power
5	1/1	1/1	1/1
2	6/0.19	3.7/0.2	2.6/0.23
1.5	11/0.13	7/0.12	7/0.14
1.4	15/0.14	10/0.11	Recursive bottleneck reached

areas gives an indication of the amount of parallelism being exploited. The key point is that the optimal voltage was found to be relatively independent over all the cases considered, and occurred around 1.5 V for the 2.0-μm technology; a similar analysis using a 0.5-V threshold, 0.8-μm process (with an L_{eff} of 0.5 μm) resulted in optimal voltages around 1 V, with power reductions in excess of a factor of 10. Further scaling of the threshold would allow even lower voltage operation, and hence even greater power savings.

VI. CONCLUSIONS

There are a variety of considerations that must be taken into account in low-power design which include the style of logic, the technology used, and the logic implemented. Factors that were shown to contribute to power dissipation included spurious transitions due to hazards and critical race conditions, leakage and direct path currents, precharge transitions, and power-consuming transitions in unused circuitry. A pass-gate logic family with modified threshold voltages was found to be the best performer for low-power designs, due to the minimal number of transistors required to implement the important logic func-

tions. An analysis of transistor sizing has shown that minimum-sized transistors should be used if the parasitic capacitances are less than the active gate capacitances in a cascade of logic gates.

With the continuing trend of denser technology through scaling and the development of advanced packaging techniques, a new degree of freedom in architectural design has been made possible in which silicon area can be traded off against power consumption. Parallel architectures, utilizing pipelining or hardware replication, provide the mechanism for this trade-off by maintaining throughput while using slower device speeds and thus allowing reduced voltage operation. The well-behaved nature of the dependencies of power dissipation and delay as a function of supply voltage over a wide variety of situations allows optimizations of the architecture. In this way, for a wide variety of situations, the optimum voltage was found to be less than 1.5 V, below which the overhead associated with the increased parallelism becomes prohibitive.

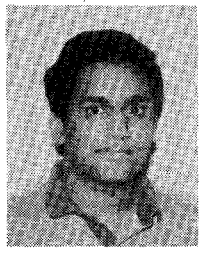
There are other limitations which may not allow the optimum supply voltage to be achieved. The algorithm that is being implemented may be sequential in nature and/or have feedback which will limit the degree of parallelism that can be exploited. Another possibility is that the optimum degree of parallelism may be so large that the number of transistors may be inordinately large, thus making the optimum solution unreasonable. However, in any case, the goal in minimizing power consumption is clear: operate the circuit as slowly as possible, with the lowest possible supply voltage.

ACKNOWLEDGMENT

The authors wish to thank A. Burstein, M. Potkonjak, M. Srivastava, A. Stoelzle, and L. Thon for providing us with examples. We also would like to thank Prof. P. Ko, Prof. T. Meng, Prof. J. Rabaey, and Prof. C. Sodini for their invaluable feedback. Lastly, we would also wish to acknowledge the support of S. Sheng by the Fannie and John Hertz Foundation.

REFERENCES

[1] T. Bell, "Incredible shrinking computers," *IEEE Spectrum*, pp. 37–43, May 1991.
[2] Eager, "Advances in rechargeable batteries pace portable computer growth," in *Proc. Silicon Valley Personal Comput. Conf.*, 1991, pp. 693–697.
[3] A. Chandrakasan, S. Sheng, and R. W. Brodersen, "Design considerations for a future multimedia terminal," presented at the 1990 WINLAB Workshop (Rutgers Univ., New Brunswick, NJ), Oct. 1990.
[4] H. B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI.* Menlo Park, CA: Addison-Wesley, 1990.
[5] D. Benson *et al.*, "Silicon multichip modules," presented at the Hot Chips Symp. III, Santa Clara, CA, Aug. 1990.
[6] G. Geschwind and R. M. Clary, "Multichip modules—An overview," in *Expo SMT/HiDEP 1990 Tech. Proc.*, (San Jose, CA), 1990, pp. 319–329.
[7] D. A. Patterson and C. H. Sequin, "Design considerations for single-chip computers of the future," *IEEE J. Solid-State Circuits*, vol. SC-15, no. 1, Feb. 1980; also *IEEE Trans. Comput.*, vol. C-29, no. 2, pp. 108–116, Feb. 1980.
[8] K. Yano *et al.*, "A 3.8-ns CMOS 16 × 16 multiplier using complementary pass transistor logic," *IEEE J. Solid-State Circuits*, vol. 25, pp. 388–395, Apr. 1990.
[9] H. J. M. Veendrick, "Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits," *IEEE J. Solid-State Circuits*, vol. SC-19, pp. 468–473, Aug. 1984.
[10] N. Weste and K. Eshragian, *Principles of CMOS VLSI Design: A Systems Perspective.* Reading, MA: Addison-Wesley, 1988.
[11] R. K. Watts, Ed., *Submicron Integrated Circuits.* New York: Wiley, 1989.
[12] D. Green, *Modern Logic Design.* Reading, MA: Addison-Wesley, 1986, pp. 15–17.
[13] G. Jacobs and R. W. Brodersen, "A fully asynchronous digital signal processor using self-timed circuits," *IEEE J. Solid-State Circuits*, vol. 25, pp. 1526–1537, Dec. 1990.
[14] D. Hodges and H. Jackson, *Analysis and Design of Digital Integrated Circuits.* New York: McGraw-Hill, 1988.
[15] M. Shoji, *CMOS Digital Circuit Technology.* Englewood Cliffs, NJ: Prentice-Hall, 1988.
[16] S. Sze, *Physics of Semiconductor Devices.* New York: Wiley, 1981.
[17] M. Aoki *et al.*, "0.1μm CMOS devices using low-impurity-channel transistors (LICT)," in *IEDM Tech. Dig.*, 1990, pp. 939–941.
[18] M. Nagata, "Limitations, innovations, and challenges of circuits & devices into half-micron and beyond," in *Proc. Symp. VLSI Circuits*, 1991, pp. 39–42.
[19] S. C. Ellis, "Power management in notebook PC's," in *Proc. Silicon Valley Personal Comput. Conf.*, 1991, pp. 749–754.
[20] K. Chu and D. Pulfrey, "A comparison of CMOS circuit techniques: Differential cascode voltage switch logic versus conventional logic," *IEEE J. Solid-State Circuits*, vol. SC-22, pp. 528–532, Aug. 1987.
[21] R. W. Brodersen, *Lager: A Silicon Compiler.* Norwell, MA: Kluwer, to be published.
[22] R. W. Brodersen *et al.*, "LagerIV cell library documentation," Electron. Res. Lab., Univ. Calif., Berkeley, June 23, 1988.
[23] B. Davari *et al.*, "A high performance 0.25μm CMOS technology," in *IEDM Tech. Dig.*, 1988, pp. 56–59.
[24] M. Kakumu and M. Kinugawa, "Power-supply voltage impact on circuit performance for half and lower submicrometer CMOS LSI," *IEEE Trans. Electron Devices*, vol. 37, no. 8, pp. 1902–1908, Aug. 1990.
[25] D. Dahle, "Designing high performance systems to run from 3.3 V or lower sources," in *Proc. Silicon Valley Personal Comput. Conf.*, 1991, pp. 685–691.
[26] R. Swansan and J. Meindl, "Ion-implanted complementary MOS transistors in low-voltage circuits," *IEEE J. Solid-State Circuits*, vol. SC-7, pp. 146–153, Apr. 1972.
[27] D. Messerschmitt, "Breaking the recursive bottleneck," in *Performance Limits in Communication Theory and Practice*, 1988, J. K. Skwirzynski, Ed., pp. 3–19.
[28] M. Potkonjak and J. Rabaey, "Optimizing resource utilization using transformations," in *Proc. ICCAD*, Nov. 1991, pp. 88–91.



Anantha P. Chandrakasan received the B.S. (with highest honors) and M.S. degrees in electrical engineering and computer sciences from the University of California, Berkeley, in 1989 and 1990, respectively. He is currently working towards the Ph.D. degree in electrical engineering at Berkeley.

His research interests include low-power techniques for portable real-time applications, video compression, computer-aided design tools for VLSI, and system-level integration.

Mr. Chandrakasan is a member of Eta Kappa Nu and Tau Beta Pi.

Samuel Sheng received the B.S. degree in electrical engineering and the B.A. degree in applied mathematics in 1989 from the University of California, Berkeley, and has just completed his thesis work there for the M.S. degree in electrical engineering on a design of a wide-band portable digital microwave transceiver. He will continue on in the Ph.D. program in the area of integrated RF transceivers, with emphasis on high-speed MOS analog design and low power.

During the summer of 1991, he worked with the Advanced Technologies Group at Apple Computer in Sunnyvale, CA, in the area of digital communications.

Mr. Sheng is a member of Phi Beta Kappa, Eta Kappa Nu, and Tau Beta Pi, and has won the Certificate of Academic Distinction from the University of California and the Departmental Citation in the Department of EECS there. He was also a National Merit Scholar, and since 1989 has been a Fellow of the Fannie and John Hertz Foundation.

Robert W. Brodersen (M'76–SM'81–F'82) received the B.S. degrees in electrical engineering and in mathematics from California State Polytechnic University, Pomona, in 1966. From the Massachusetts Institute of Technology, Cambridge, he received the Engineers and M.S. degrees in 1968, and the Ph.D. degree in 1972.

From 1972 to 1976 he was with the Central Research Laboratory at Texas Instruments Incorporated, Dallas, where he was engaged in the study of operation and applications of charge-coupled devices. In 1976 he joined the Electrical Engineering and Computer Science faculty of the University of California at Berkeley, where he is now a Professor. In addition to teaching, he is involved in research involving new applications of integrated circuits, which is focused in the area of signal processing, and the CAD tools necessary to support this activity.

Dr. Brodersen has won best paper awards at a number of conferences, and in 1979 he received the W. G. Baker award. In 1982 he became a Fellow of the IEEE and in 1983 he was co-recipient of the IEEE Morris Liebmann award. In 1986 and 1991 he received the Technical Achievement awards in the IEEE Circuits and Systems Society and the Signal Processing Society. In 1988 he was elected to be a member of the National Academy of Engineering.