Low-Power Digital Design

Mark Horowitz, Thomas Indermaur, and Ricardo Gonzalez

Center for Integrated Systems, Stanford University, Stanford, CA 94305
{horowitz,tni,ricardog}@chroma.stanford.edu

Introduction

Recently there has been a surge of interest in low-power devices and design techniques. While many papers have been published describing power-saving techniques for use in digital systems, trade-offs between the methods are rarely discussed. We address this issue by using an energy-delay metric to compare many of the proposed techniques. Using this metric also provides insight into some of the basic trade-offs in low-power design.

The next section describes the energy-loss mechanisms that are present in CMOS circuits, which provides the parameters that must be changed to lower the power dissipation. With these factors in mind, the rest of the paper reviews the energy saving techniques that have been proposed. These proposals fall into one of three main strategies: trade speed for power, don't waste power, and find a lower power problem.

CMOS Power Dissipation

Power dissipation in CMOS circuits arises from two different mechanisms: static power, which results from resistive paths from the power supply to ground, and dynamic power, which results from switching capacitive loads between two different voltage states. Dynamic power is frequency dependent, since no power is dissipated if the node values don't change, while static power is independent of frequency and exists whenever the chip is powered on. For uses where the electronics will be inactive for much of the time (most portable applications), the static power must be made very low in the inactive state.

Even if there are no explicit circuits using static current, the chip will dissipate some static power. This power is the result of leakage current through nominally off transistors. The leakage is set by the sub-threshold current of the transistor,

$$I_{ds} \sim \frac{W}{L} I_s \times \exp\left(\frac{V_{gs} - V_{th}}{a v_T}\right) \qquad (1)$$

where v_T (kT/q) is around 26mV at room temperature, 'a' is a constant slightly larger than 1, and I_s is roughly $\mu C_{ox}(av_T)^2$ or 0.3μA/μ. This leakage current and the allowable static power limit how low one can make the threshold voltage. The situation is made worse by the fact that the threshold voltage is not perfectly controlled, and thus the nominal value must guarantee that the leakage is acceptable in the worst-case situation.

Some numbers will make this limit clearer. A 10mm square chip will generally contain a few meters of transistor width. If the static current limit for this chip is 100μA, then the leakage current of an off transistor must be under 0.1nA/μ. To achieve this leakage requires Vth to be around 8 av_T in the worst-case situation, which would be high temperature and a low-threshold fabrication run. If the fab control on Vth is 100mV, the nominal value of the threshold would be around 0.35V.

With small static power, charging and discharging capacitors generally consumes most of the power on a CMOS circuit.* In charging a load capacitor C up ΔV volts, and then discharging it to its original voltage, a gate pulls C ΔV from the Vdd supply to charge up the capacitor, and then sinks this charge to Gnd to discharge the node. So at the end of a cycle, the gate / capacitor combination has moved C ΔV of charge from Vdd to Gnd, which uses C ΔV Vdd of energy and is independent of the cycle time. The dynamic power of this node is the energy per cycle, times the number of cycles it makes a second, or

$$P = C\,\Delta V\,Vdd\;\alpha F \qquad (2)$$

where α is the number of times this node cycles each clock cycle and is usually called the activity ratio. The dynamic power for the whole chip is the sum of (eq. 2) over all the nodes in the circuit.

From this formula it is clear what we need to do to reduce the dynamic power. We can either reduce the capacitance being switched, the voltage swing, the power-supply voltage, the activity ratio, or the operating frequency. The power-saving techniques described in the following sections provide a number of ways to reduce these parameters.

Low-Power Design Techniques

Until relatively recently, power was an afterthought in the design process. Designers would optimize their design to meet performance and area constraints, and then talk with the packaging and system designers to figure out how they were going to deal with the power of the chip. Probably the most important low-power design method is simply to make low power a key objective in the design process. Once this is done, a lot of power can be saved by not doing "stupid" things – by simply not wasting power. For example, lowering the power supply from 5V to 3.3V, rather than using internal

Funding for this research was provided by ARPA under contract J-FBI-92-194.

* Shunt current that occurs when both devices are on is usually a small percent of the dynamic power (5-10%) and will be ignored in this paper.

voltage regulation, is an obvious design decision if low power is an objective. Removing circuits that dissipate static power and powering down inactive blocks are other examples of how wasted power can be saved.

To help find wasted power we need a metric that allows us to compare two designs to see which is more efficient. The obvious choices for a low-power metric, power and energy, turn out to have serious flaws. Using power as the metric has the problem that CMOS circuits use energy mostly when they switch their outputs. One can always reduce the power by reducing the operating frequency, which is not a useful result.

An alternative metric is the energy needed to complete an operation. This is an improvement over power because running the part slower does not directly change the energy used in an operation, it simply spreads the same energy use over a longer time. The problem with this metric is that the energy an operation requires can be made smaller by reducing the supply voltage since the energy is roughly nCV^2, where nC is the sum of capacitance times transitions that are needed to complete the operation. However, the lower supply voltage also affects performance, and dramatically increases the delay of the operation. Thus the lowest energy solution also will run very slowly.

To avoid these problems, we use the metric of delay/op x energy/op. Smaller energy-delay values imply a lower energy solution at the same level of performance – a more energy-efficient design. The following sections will discuss various low-power design techniques, and show how they affect the energy-delay product. The first three methods (voltage scaling, transistor sizing, and adiabatic circuits) only have a small effect on the energy-delay product and are really methods for trading speed for power. The next two sections describe ways of not using energy needlessly. Finally, the last two sections describe how reformulating the problem at the system level can yield large improvements in the energy-delay product.

Voltage Scaling

In a given technology the energy per operation can be reduced by lowering the power-supply voltage. However, since both capacitance and threshold voltage are constant, the speed of the basic gates will also decrease with this voltage scaling. We can use a charge control model to estimate the delay of a gate by dividing the charge needed to transition the node by the transistor current. As other researchers have shown [3], using a quadratic model of a transistor leads to:

$$t_d = k\frac{CV}{(V - V_{th})^2} \qquad (3)$$

Figure 1 plots energy / operation, delay and energy-delay as the supply voltage is scaled. At large voltages, reducing the supply reduces the energy for a modest change in delay

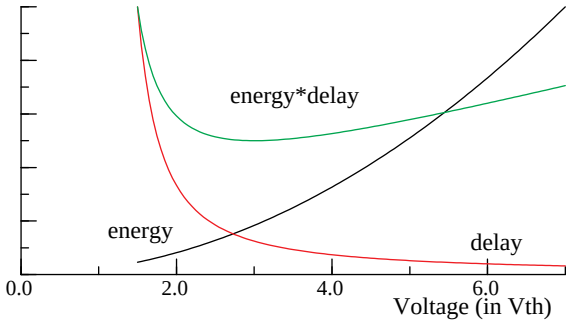

Figure 1. Energy and Delay vs. Voltage

(especially in the velocity saturated case, where the delay change is even less than shown in the figure). At voltages near the device threshold, small supply changes cause a large change in delay for a modest change in energy. While there is a minima at Vdd = $3V_{th}$, it is pretty flat. Around this point changing the supply voltage does not strongly affect the energy-delay product, allowing one to trade delay for energy. From the 3 V_{th} point, there is a factor of about 4 in energy in either direction (from 1.5 V_{th} to 6 V_{th}) that can be traded for delay without greatly changing the energy-delay product. Below 1.5 V_{th} the surplus performance would be better spent in some other way, like reducing the transistor sizes.

Transistor Sizing

Like supply voltage, sizing gates mostly presents the designer an opportunity to trade speed for power, rather than reducing their product. Since some of the load capacitance is caused by the gate capacitance of other transistors, one can reduce the energy of an operation by making all the transistors smaller. However, decreasing the size of the transistors also decreases their current drive, and thus makes the gates slower. This trade-off can be easily seen using a chain of uniformly loaded inverters, which are shown in Figure 2.

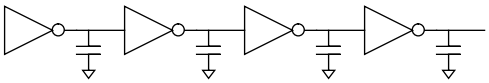

Figure 2. Simple Inverter Chain

Figure 3 graphs the delay, energy, and energy-delay of a stage as a function of the transistor's capacitance contribution to the total load. The load will be mostly load capacitance for small transistor, and will be mostly gate for large devices. For very small transistors, energy is dominated by switching the load capacitance, while the delay is inversely proportional to the transistor width, so increasing the transistors improves the energy-delay product. For large transistors, the gates are limited by self loading, so decreasing the transistor size improves the metric. The optimal operating point is when the transistor loading is the same as the wire loading.

Obviously, real circuits are more complex. The gate and wire capacitance is different for different gates, nodes transition at

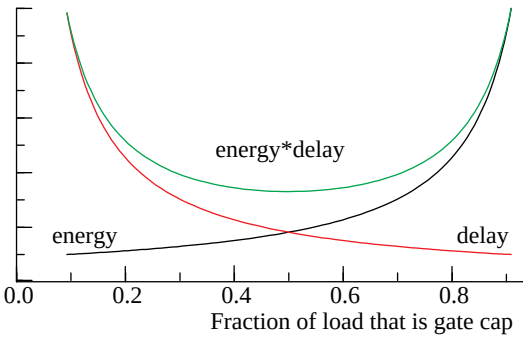

Figure 3. Energy, Delay vs. Transistor Width

different frequencies, and not all gates are on the critical path. While this problem is difficult to solve precisely, the structure of the solution remains roughly the same as the simple inverter chain: making the critical path transistors much smaller than their loads will greatly increase the delay without reducing the power, and making the transistors much larger than their loads will greatly increase the energy without having a large effect on the delay.

The energy-delay product is roughly constant as the percentage of gate loading changes from 20% to 80%, which is roughly a factor of 5 in speed and power. While using minimum-sized devices can lead to lower power solutions [3], they do not lead to more energy efficient solutions.

Adiabatic Circuits

Adiabatic or charge-recovery circuits, are another method that allow a designer to explicitly trade performance for lower energy requirements [11][7]. These circuits resonate the load capacitance with an inductor, which recovers some of the energy needed to change the capacitor's voltage. The energy loss in switching the load can be reduced to $\tau/T\ CV^2$, where τ is the intrinsic delay of the gate, and T is the delay set by the LC circuit. While this ease in trading energy for delay is attractive, the energy-delay product for these circuits is much worse than normal CMOS gates[6]. Thus adiabatic circuits become attractive only when you need to operate at delays beyond the range viable by voltage scaling and transistor sizing standard CMOS.

Technology Scaling

One way to greatly improve the energy-delay product, and thus save energy, is to improve the technology. In ideal scaling as first described by Dennard[5], all voltages and linear dimensions are reduced by a scale factor, γ (<1). Since the E-fields in the devices and wires remain constant, the device current* and device and wire capacitance all scale as γ. Since the voltage also scales by γ, the energy of an operation scales as γ^3. The delay of each gate also improves by γ, since the delay is roughly $t_d = CV/i$. The energy-delay product decreases by γ^4, implying a 0.7 shrink of a chip can be run at the same performance for roughly 1/4 the power.

The difficulty with ideal scaling is the requirement for Vth to scale along with the supply voltage. As was mentioned earlier, static power caused by leakage current through the off transistors will limit how low the threshold voltage can be scaled.† Even with constant voltage scaling, the reduced capacitance improves both the energy and the delay, so their product scales at least as γ^2.

Transition Reduction

Another way to improve the energy-delay product is to avoid wasting energy – avoid causing node transitions that are not needed. One common approach to solve this problem is to make sure that idle blocks do not use any power. The key to selective activation is to control objects that dissipate a significant amount of power. From our work analyzing power of digital systems, around 70% of the power comes from high-transition count, high-capacitance nodes – like clocks and buses – which comprise less than 20% of the nodes in a given design. While doing selective activation of a set of 64 bus lines might make sense, trying to reduce the number of transitions in the adder that drives the bus does not.

As long as the static power is small, the circuit only uses power when a node switches. Thus an idle section can be powered down simply by preventing its outputs from switching (generally by keeping its inputs stable). At the block level on a chip, the activation is usually done by gating the clock to the function blocks[10]. When the clock is turned off, none of the latch outputs change state, and thus the logic outputs are also stable. Gating the clock has the added advantage that it reduces the clock load that toggles each cycle, since the clocks in the inactive blocks are effectively turned off. On low-power processors, the caches, FPU, and integer unit can all be independently controlled [1]. Generally the performance impact of the clock gating is small, so the energy-delay product decreases by the energy saving.

Reducing unnecessary toggles will reduce the energy-delay product, but it rarely changes it by more than a small integer factor (2 or 3). To get more significant reductions requires examining the problem from the system level.

Parallelism

One can improve the energy-delay product by reducing either the energy or the delay. Voltage and transistor scaling allow a designer to trade excess performance for lower energy operations. The ability to trade delay for energy points out the strong connection between high-speed and low-power designs. One wants to start with a solution with a large

* This relations holds independent of whether the devices are velocity saturated or not.

† There has been some work to allow larger leakage currents and switch the power supply off to these sections using lower leakage (higher threshold) transistors. This might allow slightly lower threshold transistors in the active circuits but requires a sophisticated power management system on chip [8].

amount of excess performance that can then be traded for reduced power. A way of generating this performance is by exploiting parallelism.

When an application has parallelism, one can build N functional units instead of one, and solve N problems at the same time. Doing this increases the performance by nearly N (there is some time needed to distribute the operands, and collect the results), and increases the power by slightly over N (again because of overhead). Thus using parallelism increases the energy/op by only the overhead while the effective delay/op drops by N minus the delay overhead. The energy-delay product of the parallel solution is much lower (roughly N times lower) than the original sequential approach. This argument is independent of how the parallelism is extracted (pipelining, parallel machine, etc.), although the overhead factors will be different. For DSP applications with a large amount of parallelism, the performance gains allow the resulting systems to run at very low power supply voltage, use small transistor sizes, and still meet their performance targets [4].

In some applications, the available parallelism is smaller and harder to extract. In processors the cost of issuing multiple instructions is not small, and does not yield a performance gain for all code sequences. As a result, as shown in Table 1, parallel execution neither helps or hurts a processor's energy-delay product (Watts/SPEC^2). Fabrication technology seems more important for the energy-delay product than whether the machine is superscalar (21064, PPC604) or not.

Table 1 Energy-Delay for some Recent Processors

μP	DEC 21064	MIPS R4200	IDT R4600	PPC 604	PPC 603
SPECavg	155	42.5	64	162.5	80
Power	30W	1.8W	3W	13W	3W
SPEC^2/W	800	1000	1400	2000	2100
Min L	0.75μ	0.64μ	0.64μ	0.5μ	0.5μ

Redefine the Problem

So far we have looked at ways to more efficiently implement the tasks needed to complete some operation. Yet this discussion missed the most important method of reducing system energy – reduce the number/complexity of tasks that the operation requires. It is at this level that the designer can make the largest changes to the energy-delay product, since simplifications often reduce both the energy and the delay of the operation. The key point to realize is that the energy-delay product measures the energy to complete some user operation and the delay to complete that operation. If we can simplify the operation, we reduce the number of primitive steps required, and thus reduce both the energy and the delay.

As a simple example of the saving that is possible, consider a operation that is implemented as a program on a microcontroller. The initial code for this operation takes N micro instructions to execute, so the energy for the operation is N times the instruction energy, and the delay is N times the instruction delay. If another approach can perform the same operation in M instruction, the energy-delay product will change by $(M/N)^2$, since both the delay and energy decrease by (M/N).

This strategy works for hardware designs as well, with similar quadratic gains. Often a reformulation of a problem can lead to a solution that requires less computation to accomplish the same task [2][9]. Orders of magnitude gains are possible at this level. Unfortunately the optimizations used tend to be tied to the specific application that is being optimized. The good news is that this process is similar to the ones used to increase system performance. The bad news is that these system level optimizations generally require some creative insight.

Conclusions

Good design has always required one to make careful trade-offs, and low-power design simply means one needs to consider energy dissipation in addition to the normal concerns of speed, area, and design-time. The energy-delay product is a useful guide for making these trade-offs. It allows a designer to find optimizations that provide the largest reduction in energy for the smallest change in performance. It also makes clear the strong coupling between performance and power which is the reason that many high-performance techniques are useful for low-power design.

References

[1] R. Bechade, et al., "A 32b 66MHz 1.8W Microprocessor, ISSCC, Feb 1994, pg 208-209.

[2] B. Brandt, B. Wooley, "A Low Power, Area-Efficient Digital Filter for Decimation and Interpolation," IEEE Journal of Solid State Circuits, SC29, June 1994.

[3] A. Chandrakasan, et al. "Low-power CMOS digital design." IEEE Journal of Solid-state Circuits Vol 27 pg 473-484.

[4] A. Chandrakasan, et al, "A Low Power Chipset for Portable Multimedia Applications," ISSCC, Feb 1994, pg 82-83.

[5] R. Dennard et al., "Design of Ion Implanted MOSFET's with Very Small Dimensions," IEEE Journal of Solid State Circuits, SC9, pg 256-267, 1974.

[6] T. Indermaur, et al., "Evaluation of Charge Recovery Circuits and Adiabatic Switching for Low Power CMOS Design," Symposium on Low-Power Electronics, Oct 1994.

[7] J. Koller, W. Athas, "Adiabatic Switching, Low Energy Computing, and the Physics of Storing and Erasing Information," Proceedings of Physics of Computation Workshop, Oct. 1992.

[8] D. Takashima, et al., "Standby/Active Model Logic for Sub-1V Operating ULSI Memory," IEEE Journal of Solid State Circuits, Vol 29, pg 441-447, 1994.

[9] E. Tsern, et. al., "Video Compression For Portable Communication Using Pyramid Vector Quantization of Subband Coefficients," IEEE Workshop on VLSI Signal Processing, Oct 1993.

[10] N. Yeung et al., "The Design of a 55 SPECint92 RISC Processor under 2W," ISSCC, Feb 1994, pg 206-207.

[11] S. Younis, T. Knight, "Practical Implementation of Charge Recovering Asymptotically Zero Power CMOS," Proceedings of the 1993 Symp. on Integrated Sys., MIT Press, pg. 234-250, 1993.