Power Reduction Techniques For Microprocessor Systems

VASANTH VENKATACHALAM AND MICHAEL FRANZ

University of California, Irvine

Power consumption is a major factor that limits the performance of computers. We survey the "state of the art" in techniques that reduce the total power consumed by a microprocessor system over time. These techniques are applied at various levels ranging from circuits to architectures, architectures to system software, and system software to applications. They also include holistic approaches that will become more important over the next decade. We conclude that power management is a multifaceted discipline that is continually expanding with new techniques being developed at every level. These techniques may eventually allow computers to break through the "power wall" and achieve unprecedented levels of performance, versatility, and reliability. Yet it remains too early to tell which techniques will ultimately solve the power problem.

Categories and Subject Descriptors: C.5.3 [**Computer System Implementation**]: Microcomputers—*Microprocessors*; D.2.10 [**Software Engineering**]: Design—*Methodologies*; I.m [**Computing Methodologies**]: Miscellaneous

General Terms: Algorithms, Design, Experimentation, Management, Measurement, Performance

Additional Key Words and Phrases: Energy dissipation, power reduction

1. INTRODUCTION

Computer scientists have always tried to improve the performance of computers. But although today's computers are much faster and far more versatile than their predecessors, they also consume a lot of power; so much power, in fact, that their power densities and concomitant heat generation are rapidly approaching levels comparable to nuclear reactors (Figure 1). These high power densities impair chip reliability and life expectancy, increase cooling costs, and, for large

Parts of this effort have been sponsored by the National Science Foundation under ITR grant CCR-0205712 and by the Office of Naval Research under grant N00014-01-1-0854.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and should not be interpreted as necessarily representing the official views, policies or endorsements, either expressed or implied, of the National Science foundation (NSF), the Office of Naval Research (ONR), or any other agency of the U.S. Government.

The authors also gratefully acknowledge gifts from Intel, Microsoft Research, and Sun Microsystems that partially supported this work.

Authors' addresses: Vasanth Venkatachalam, School of Information and Computer Science, University of California at Irvine, Irvine, CA 92697-3425; email: vvenkata@uci.edu; Michael Franz, School of Information and Computer Science, University of California at Irvine, Irvine, CA 92697-3425; email: franz@uci.edu.

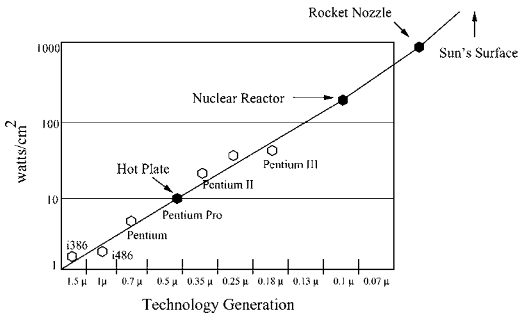

Fig. 1. Power densities rising. Figure adapted from Pollack 1999.

data centers, even raise environmental concerns.

At the other end of the performance spectrum, power issues also pose problems for smaller mobile devices with limited battery capacities. Although one could give these devices faster processors and larger memories, this would diminish their battery life even further.

Without cost effective solutions to the power problem, improvements in micro-processor technology will eventually reach a standstill. Power management is a multidisciplinary field that involves many aspects (i.e., energy, temperature, reliability), each of which is complex enough to merit a survey of its own. The focus of our survey will be on techniques that reduce the total power consumed by typical microprocessor systems.

We will follow the high-level taxonomy illustrated in Figure 2. First, we will define power and energy and explain the complex parameters that dynamic and static power depend on (Section 2). Next, we will introduce techniques that reduce power and energy (Section 3), starting with circuit (Section 3.1) and architectural techniques (Section 3.2, Section 3.3, and Section 3.4), and then moving on to two techniques that are widely applied in hardware and software, *dynamic voltage scaling* (Section 3.5) and *resource hibernation* (Section 3.6). Third, we will examine what compilers can do to manage power (Section 3.7). We will then discuss recent work in application level power management (Section 3.8), and recent efforts (Section 3.9) to develop a holistic solution to the power problem. Finally, we will discuss some commercial power management systems (Section 3.10) and provide a glimpse into some more radical technologies that are emerging (Section 3.11).

2. DEFINING POWER

Power and energy are commonly defined in terms of the work that a system performs. *Energy* is the total amount of work a system performs over a period of time, while *power* is the rate at which the system performs that work. In formal terms,

$$P = W/T \tag{1}$$

$$E = P * T, \tag{2}$$

where P is power, E is energy, T is a specific time interval, and W is the total work performed in that interval. Energy is measured in *joules*, while power is measured in *watts*.

These concepts of work, power, and energy are used differently in different contexts. In the context of computers, work involves activities associated with running programs (e.g., addition, subtraction, memory operations), power is the rate at which the computer consumes electrical energy (or dissipates it in the form of heat) while performing these activities, and energy is the total electrical energy the computer consumes (or dissipates as heat) over time.

This distinction between power and energy is important because techniques that reduce power do not necessarily reduce energy. For example, the power consumed by a computer can be reduced by halving the clock frequency, but if the computer then takes twice as long to run the same programs, the total energy consumed will be similar. Whether one should reduce power or energy depends on the context. In mobile applications, reducing energy is often more important because it increases the battery lifetime. However, for other systems (e.g., servers), temperature is a larger issue. To keep the temperature within acceptable limits, one would need to reduce instantaneous power regardless of the impact on total energy.

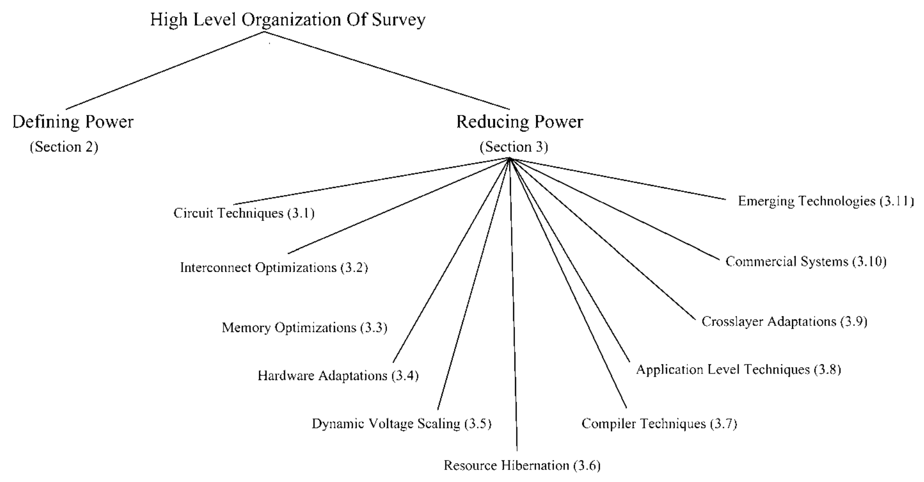
High Level Organization Of Survey
Defining Power (Section 2)
Reducing Power (Section 3)
Circuit Techniques (3.1)
Interconnect Optimizations (3.2)
Memory Optimizations (3.3)
Hardware Adaptations (3.4)
Dynamic Voltage Scaling (3.5)
Resource Hibernation (3.6)
Compiler Techniques (3.7)
Application Level Techniques (3.8)
Crosslayer Adaptations (3.9)
Commercial Systems (3.10)
Emerging Technologies (3.11)

Fig. 2. Organization of this survey.

2.1. Dynamic Power Consumption

There are two forms of power consumption, *dynamic power consumption* and *static power consumption*. Dynamic power consumption arises from circuit activity such as the changes of inputs in an adder or values in a register. It has two sources, switched capacitance and short-circuit current.

Switched capacitance is the primary source of dynamic power consumption and arises from the charging and discharging of capacitors at the outputs of circuits.

Short-circuit current is a secondary source of dynamic power consumption and accounts for only 10-15% of the total power consumption. It arises because circuits are composed of transistors having opposite polarity, negative or *NMOS* and positive or *PMOS*. When these two types of transistors switch current, there is an instant when they are simultaneously on, creating a short circuit. We will not deal further with the power dissipation caused by this short circuit because it is a smaller percentage of total power, and researchers have not found a way to reduce it without sacrificing performance.

As the following equation shows, the more dominant component of dynamic power, switched capacitance ($P_{dynamic}$), depends on four parameters namely, supply voltage (V), clock frequency (f), physical capacitance (C) and an activity factor (a) that relates to how many $0 \rightarrow 1$ or $1 \rightarrow 0$ transitions occur in a chip:

$$P_{dynamic} \sim aCV^2 f. \qquad (3)$$

Accordingly, there are four ways to reduce dynamic power consumption, though they each have different tradeoffs and not all of them reduce the total energy consumed. The first way is to reduce the physical capacitance or stored electrical charge of a circuit. The physical capacitance depends on low level design parameters such as transistor sizes and wire lengths. One can reduce the capacitance by reducing transistor sizes, but this worsens performance.

The second way to lower dynamic power is to reduce the switching activity. As computer chips get packed with increasingly complex functionalities, their switching activity increases [De and Borkar 1999], making it more important to develop techniques that fall into this category. One popular technique, *clock gating*, gates the clock signal from reaching idle functional units. Because the clock network accounts for a large fraction of a chip's total energy consumption, this is a very effective way of reducing power and energy throughout a processor and is implemented in numerous commercial systems

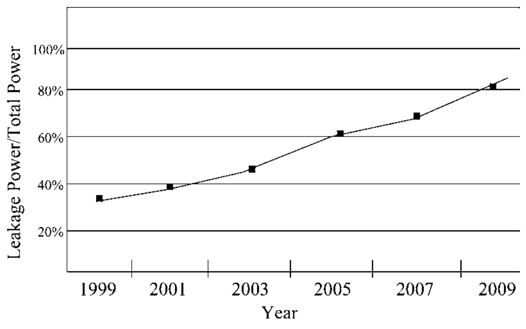

Fig. 3. ITRS trends for leakage power dissipation. Figure adapted from Meng et al., 2005.

including the Pentium 4, Pentium M, Intel XScale and Tensilica Xtensa, to mention but a few.

The third way to reduce dynamic power consumption is to reduce the clock frequency. But as we have just mentioned, this worsens performance and does not always reduce the total energy consumed. One would use this technique only if the target system does not support voltage scaling and if the goal is to reduce the peak or average power dissipation and indirectly reduce the chip's temperature.

The fourth way to reduce dynamic power consumption is to reduce the supply voltage. Because reducing the supply voltage increases gate delays, it also requires reducing the clock frequency to allow the circuit to work properly.

The combination of scaling the supply voltage and clock frequency in tandem is called *dynamic voltage scaling* (DVS). This technique should ideally reduce dynamic power dissipation cubically because dynamic power is quadratic in voltage and linear in clock frequency. This is the most widely adopted technique. A growing number of processors, including the Pentium M, mobile Pentium 4, AMD's Athlon, and Transmeta's Crusoe and Efficieon processors allow software to adjust clock frequencies and voltage settings in tandem. However, DVS has limitations and cannot always be applied, and even when it can be applied, it is nontrivial to apply as we will see in Section 3.5.

2.2. Understanding Leakage Power Consumption

In addition to consuming dynamic power, computer components consume *static power*, also known as *idle power* or *leakage*. According to the most recently published industrial roadmaps [ITRSRoadMap], leakage power is rapidly becoming the dominant source of power consumption in circuits (Figure 3) and persists whether a computer is active or idle. Because its causes are different from those of dynamic power, dynamic power reduction techniques do not necessarily reduce the leakage power.

As the equation that follows illustrates, leakage power consumption is the product of the supply voltage (V) and *leakage current* (I_{leak}), or parasitic current, that flows through transistors even when the transistors are turned off.

$$P_{leak} = V I_{leak}. \tag{4}$$

To understand how leakage current arises, one must understand how transistors work. A transistor regulates the flow of current between two terminals called the *source* and the *drain*. Between these two terminals is an insulator, called the channel, that resists current. As the voltage at a third terminal, the *gate*, is increased, electrical charge accumulates in the channel, reducing the channel's resistance and creating a path along which electricity can flow. Once the gate voltage is high enough, the channel's polarity changes, allowing the normal flow of current between the source and the drain. The threshold at which the gate's voltage is high enough for the path to open is called the *threshold voltage*.

According to this model, a transistor is similar to a water dam. It is supposed to allow current to flow when the gate voltage exceeds the threshold voltage but should otherwise prevent current from flowing. However, transistors are imperfect. They leak current even when the gate voltage is below the threshold voltage. In fact, there are six different types of current that leak through a transistor. These include reverse-biased-junction leakage, gate-induced-drain leakage, subthreshold leakage, gate-oxide leakage, gate-current leakage, and punch-through leakage. Of these six, subthreshold leakage and gate-oxide leakage dominate the total leakage current.

Gate-oxide leakage flows from the gate of a transistor into the substrate. This type of leakage current depends on the thickness of the oxide material that insulates the gate:

$$I_{ox} = K_2 W \left(\frac{V}{T_{ox}}\right)^2 e^{-\alpha \frac{T_{ox}}{V}}. \qquad (5)$$

According to this equation, the gate-oxide leakage I_{ox} increases exponentially as the thickness T_{ox} of the gate's oxide material decreases. This is a problem because future chip designs will require the thickness to be reduced along with other scaled parameters such as transistor length and supply voltage. One way of solving this problem would be to insulate the gate using a high-k dialectric material instead of the oxide materials that are currently used. This solution is likely to emerge over the next decade.

Subthreshold leakage current flows between the drain and source of a transistor. It is the dominant source of leakage and depends on a number of parameters that are related through the following equation:

$$I_{sub} = K_1 W e^{\frac{-V_{th}}{nT}} \left(1 - e^{\frac{-V}{T}}\right). \qquad (6)$$

In this equation, W is the gate width and K and n are constants. The important parameters are the supply voltage V, the threshold voltage V_{th}, and the temperature T. The subthreshold leakage current I_{sub} increases exponentially as the threshold voltage V_{th} decreases. This again raises a problem for future chip designs, because as technology scales, threshold voltages will have to scale along with supply voltages.

The increase in subthreshold leakage current causes another problem. When the leakage current increases, the temperature increases. But as the Equation (6) shows, this increases leakage further, causing yet higher temperatures. This vicious cycle is known as *thermal runaway*. It is the chip designer's worst nightmare.

How can these problems be solved? Equations (4) and (6) indicate four ways to reduce leakage power. The first way is to reduce the supply voltage. As we will see, *supply voltage reduction* is a very common technique that has been applied to components throughout a system (e.g., processor, buses, cache memories).

The second way to reduce leakage power is to reduce the size of a circuit because the total leakage is proportional to the leakage dissipated in all of a circuit's transistors. One way of doing this is to design a circuit with fewer transistors by omitting redundant hardware and using smaller caches, but this may limit performance and versatility. Another idea is to reduce the effective transistor count *dynamically* by cutting the power supplies to idle components. Here, too, there are challenges such as how to predict when different components will be idle and how to minimize the overhead of shutting them on or off. This, is also a common approach of which we will see examples in the sections to follow.

The third way to reduce leakage power is to cool the computer. Several cooling techniques have been developed since the 1960s. Some blow cold air into the circuit, while others refrigerate the processor [Schmidt and Notohardjono 2002], sometimes even by costly means such as circulating cryogenic fluids like liquid nitrogen [Krane et al. 1988]. These techniques have three advantages. First they significantly reduce subthreshold leakage. In fact, a recent study [Schmidt and Notohardjono 2002] showed that cooling a memory cell by 50 degrees Celsius reduces the leakage energy by five times. Second, these techniques allow a circuit to work faster because electricity encounters less resistance at lower temperatures. Third, cooling eliminates some negative effects of high temperatures, namely the degradation of a chip's reliability and life expectancy. Despite these advantages,

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase®
Smarter legal research.