Reducing Power in High-performance Microprocessors

Vivek Tiwari, Deo Singh, Suresh Rajgopal, Gaurav Mehta, Rakesh Patel, Franklin Baez

Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95052
408-765-6395

Email contact: vtiwari@scdt.intel.com

ABSTRACT

Power consumption has become one of the biggest challenges in high-performance microprocessor design. The rapid increase in the complexity and speed of each new CPU generation is outstripping the benefits of voltage reduction and feature size scaling. Designers are thus continuously challenged to come up with innovative ways to reduce power, while trying to meet all the other constraints imposed on the design. This paper presents an overview of the issues related to power consumption in the context of Intel CPUs. The main trends that are driving the increased focus on design for low power are described. System and benchmarking issues, and sources of power consumption in a high-performance CPU are briefly described. Techniques that have been tried on real designs in the past are described. The role of CAD tools and their limitations in this domain will also be discussed. In addition, areas that need increased research focus in the future are also pointed out.

1. INTRODUCTION

The drive towards increasing levels of performance has pushed frequencies higher and has increased the complexity of microprocessors. This has come at the cost of higher power consumption. The costs associated with packaging, cooling and power delivery have thus jumped to the forefront in the microprocessor industry. There is even concern that power consumption may set the limit to how much can be integrated on a chip, and how fast it can be clocked [5].

The challenges for power reduction in high-performance general-purpose CPUs are unique. First, the instruction-set and system architecture are designed for a wide market and for a wide range of applications. This restricts the search space for low-power solutions. Second, it is necessary that proposed solutions remain robust and scale gracefully across multiple technology generations. And finally, while significant power savings are desired, they must come at *little or no* performance impact.

The aim of this paper is to highlight the key issues associated with design for low power in this domain. The paper is organized as follows. Section 2 motivates the need for low power and shows the historical trends with respect to power consumption. It also explains some metrics associated with CPU power that are of interest from the system's perspective. In Section 3, the power consumption of a recent CPU is broken down by circuit type. This power breakdown provides a quantitative basis for judging the effectiveness of different power reduction techniques. Section 4 presents an overview of the techniques that have been used in the past. Among these, voltage scaling has been the primary weapon in the battle against power consumption. But it is shown that the limitations of this approach get amplified as technology pushes towards smaller geometries. Thus, it is also important to incorporate power as a design metric at all levels of the design process, and examples from the domains of circuit, logic, CAD tools and software are presented. Section 5 concludes with a discussion of directions that need increased research focus in the future in order to deal with the power challenge.

2. THE CPU POWER PROBLEM

Figure 1 shows the power consumption for Intel CPUs. The X-axis shows the technology generation and the Y-axis the maximum power consumption. As indicated by the dashed line in the main part of the curve, power consumption has been increasing for each new CPU generation. The points to the side of the main curve indicate newer versions of each processor family. These are implemented in newer semiconductor processes with smaller geometries than the lead processor in that family.

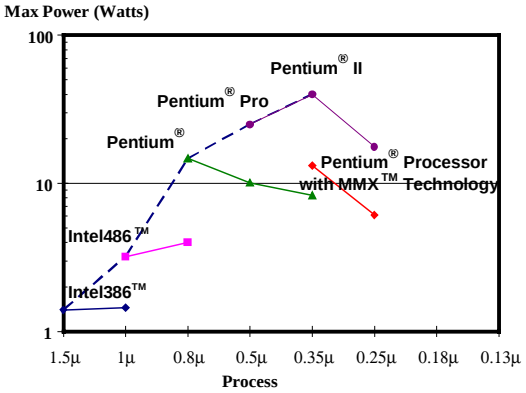

Figure 1: Power consumption for Intel CPUs.

Smaller feature sizes in conjunction with lower supply voltages lead to lower power consumption in the newer versions. However, moving to a new CPU generation in the same process is associated with an increase in the power consumption.

1-58113-049-x-98/0006/$3.50

Why is this cause for concern? The reason is that increased power consumption directly impacts CPU and system cost. This cost has two components. The first is *thermal cost*, which is associated with keeping the devices below the specified operating temperature limits. Maintaining the integrity of packaging at higher temperatures also requires expensive solutions. The second component of the cost of power consumption is the cost of power delivery, i.e., the on-chip, on-package, and on-board decoupling capacitances and interconnect associated with the power distribution network. Increased power consumption at lower voltages increases the magnitude of the current drawn by the CPU. In addition, today's design trends are such that the variability in the amount of current drawn from cycle to cycle is also increasing. These factors combine to make resistive and inductive noise a big problem. Dealing with this is becoming increasingly costly.

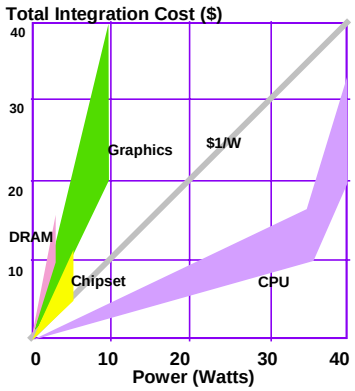

Figure 2: Thermal and power delivery cost in a PC

Figure 2 gives an idea of the range of dollar amounts associated with the above costs for different system components. As can be seen, when the CPU power is in the 35-40W range, the cost of each additional Watt goes above $1/W per chip. An interesting observation is that the power cost of the other system components (DRAM, chipsets, graphics) is on a steeper curve than the CPU. This is because the spatial layout of today's system chassis designs is such that these components are harder to cool. This is likely to change with new designs, further increasing the relative importance of the CPU power cost.

CPU activity is a new design attribute that is needed for power estimation as well as for the choice of power savings techniques. What kind of tests must be used to characterize this activity? They have to come from traces of real applications. But power dissipation inside a CPU is a complex scenario. Figure 3 shows the need for defining multiple power specifications in this regard, using illustrative power traces. The first trace, labeled *Max Power*, is associated with the power consumption while running an artificial piece of code specifically written to generate maximum CPU activity. Modern CPUs with increasingly sophisticated implementations have multiple parallel units that are not generally fully active simultaneously. Hence there is an increasing gap between artificial worst-case power and the peak demands required by *practical* applications, as shown by the second trace, labeled *Worst-Case App Trace*.

There are other power components that are of interest. The *Thermal Power* is the running average of the worst-case application power over several seconds, consistent with the component/system thermal time constant. It is used to determine cooling options and system integration and thermal design costs.

The *Standby Power* (not shown here) is power consumption under a predefined idle condition. For battery life, the criterion is *Energy*, which is the product of the *Average Power* and the running time for a given task. In the most general case, *Average Power* is often taken as the long-term average power of typical applications, measured over intervals of hours. In addition, multiple on-chip units and power down techniques have increased the variability in power consumption. This is quantified by the *Transient Power*, whose time constant is one or few CPU clock cycles. This is an important parameter in designing the power delivery system.

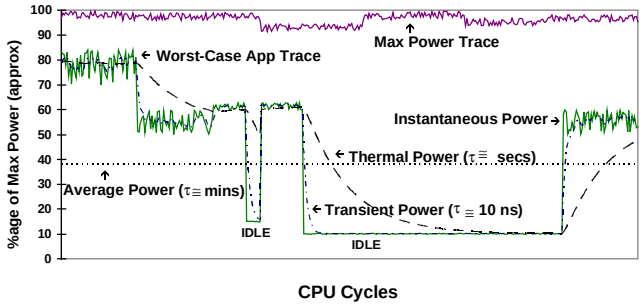

Figure 3: CPU power terminology

For each of the power components discussed above, a different trace may be needed in order to capture the desired nature of the CPU activity. Generating these traces for today's complex CPUs is itself a challenge. Another challenge is to translate these traces into test vectors for RTL and schematic power estimation. Since real traces are usually millions of clock cycles long, selection and compression are required. The selection/compression must ensure that the characteristics of an application trace that track a given power component (*viz*. worst-case, average, etc.) are maintained.

3. WHERE DOES THE POWER GO

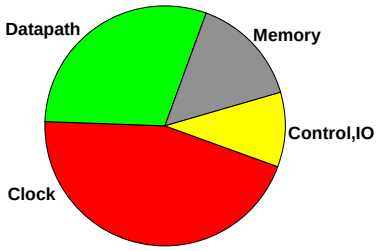

Figure 4: Power breakdown in a high-performance CPU

Figure 4 shows the power breakdown for a recent high-performance CPU, as obtained by detailed switch-level power simulation. As can be seen, the clock is the largest power consuming component. This includes the clock generator, the clock drivers, the clock distribution tree, the latches, and the clock loading due to all the clocked elements. The clock loading is actually the largest component of clock power. As shown in Figure 5, even a simple latch presents a certain amount of capacitive load to the clock network (gate capacitance of 4 clocked devices in this example). This capacitance switches on every clock tick, causing significant power consumption even when the data inputs have low activity factor (AF), or are even totally stable (AF = 0).

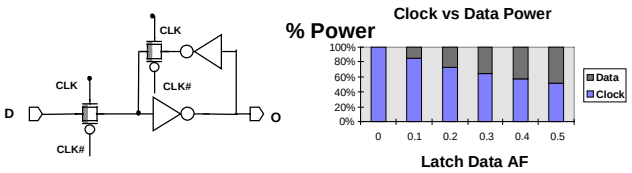

Figure 5: Power consumption in latches

After the clock, the datapath is the highest power consumer. This includes the large execution units, buses and the register files on the die. The datapath circuits in high-performance microprocessors typically have tight timing margins and are thus custom-designed. Fast and power-hungry dynamic logic is often used here. The power contribution of on-chip memories (caches, TLBs, branch buffers) is determined by the size as well as the circuit and physical structure of the memory arrays. The controller and I/O is usually a small contributor to the full-chip power. The logic blocks that constitute the control functionality are mostly non-critical from a timing standpoint. The logic and physical layout for these is thus mostly synthesized.

4. POWER REDUCTION TECHNIQUES

4.1 Voltage Scaling

Power is proportional to the square of the supply voltage (Vcc). This makes Vcc reduction as the most effective way for reducing power, and the industry has thus steadily moved to lower Vcc. This trend should and will continue. However, the drive for higher performance is outstripping the benefits of voltage scaling, as illustrated in Figure 6. The figure shows the power, feature size, voltage, frequency, and relative die size of some recent CPUs. Starting from a 5V part, there was an initial decrease in power when moving to a smaller technology at 3.3V. However, the power came back to original levels when the frequency was increased. The Pentium® Pro with its aggressive microarchitecture, saw an additional increase in power, even at 3.3V and a smaller technology.

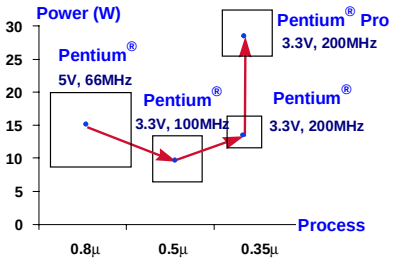

Figure 6: Power and Vcc for some recent CPUs

Another issue with voltage scaling is that to maintain performance, threshold voltage (Vt) also needs to be scaled. At low Vt, leakage power starts becoming a bigger factor [3]. A first order analysis, using constant electric field scaling of the process parameters, illustrates this as shown in Figure 7 [2]. At sub-0.1µm feature sizes, the leakage power starts eating into the benefits for lower Vcc. In addition, design of dynamic circuits, caches, sense-amps, PLAs, etc. becomes difficult at higher leakage currents. Lower Vcc also exacerbates noise and reliability concerns. Therefore, voltage scaling alone cannot be relied upon to solve the power problem. Design for low power is also necessary.

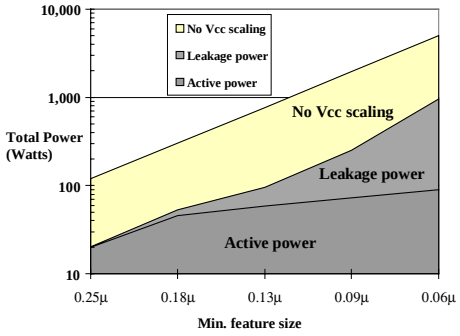

Figure 7: First order analysis for leakage power trends

4.2 Clock Gating

As discussed in Section 3, the clock is the largest contributor to the CPU power. Reducing the switched capacitance on the clocks will thus have the most impact on total power. A practical and effective way to do this is to partition the clock network and allow only those portions to toggle that are needed on each cycle. This is achieved through *clock gating* . It is implemented by qualifying the different clock partitions by special “enable” signals. It is well suited for CPUs since it can often be easily integrated into existing clock networks as illustrated in Figure 8. A regular clock buffer can be changed into a qualifying gate at low area and performance overhead.

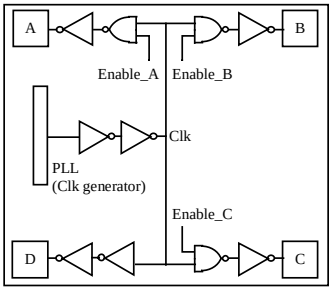

Figure 8: Clock gating and clock networks

However, clock gating cannot be used indiscriminately since there are some issues that need to be considered. An important concern is that the disabled block may not power up in time, or that modified clocks may generate glitches. This imposes strict timing constraints on the enable signals and calls for careful timing verification. Functional validation is becoming a greater challenge with each new CPU generation, and gated clocks make the problem even more difficult. In addition to this, at clock frequencies of 100MHz and above, clock skew becomes critical and every extra gate used to qualify the clock can potentially introduce timing critical skews. Thus, the granularity at which clock gating can be applied becomes a tradeoff against overall clock network design time and complexity.

Another concern with clock gating is the impact on current variations when large blocks of logic are switched on and off. A CPU may be at peak current levels for some cycles, when few blocks can be clock gated. But it may rapidly transition to low values of current if something like a stall of pipeline flush causes a number of units to be powered off. This increases the variation in transient power (*cf.* Figure 3). Switching between a normal operation mode and a standby mode, in which most of the internal clocks are turned off, also causes the same problem. This is illustrated in Figure 9 for the case of the Pentium® Processor

[7]. Notice how the differences between peak and idle current are larger when power saving techniques are enabled. Thus, the power delivery system may have to be designed for higher inductive noise specifications, increasing the CPU and system cost. This power supply noise problem will get worse with higher frequencies and lower voltages.

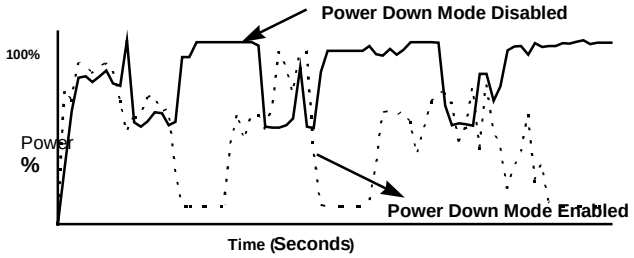

Figure 9: Power vs. time running a real application

A wide variety of techniques exist for reducing power at the logic and circuit level in a high-performance CPU. The most important of these target areas which have a large impact on the overall power of the chip (*cf* Figure 4). Redesign of latches and master slave flip flops, for example, has a large impact on full chip power. Other techniques like transistor sizing, low power logic synthesis, and specific circuit techniques also produce power savings. These are discussed below.

4.3 Libraries

Power savings from the redesign of cell libraries can come from two sources: device sizing and restructuring of the logic and physical layout of cell. Device sizing for optimizing switching energy vs. delay, ensures that the libraries are designed with power "in mind". Figure 10 shows a power vs. delay curve for two different standard cell libraries. The transistor sizes in Library 1 have been optimized for minimum delay as a target. If one were to change the optimization criteria of transistor sizes to that of energy *and* delay, one would get a lower power library (Library 2). For Library 2, the delay penalty is small but the power reduction is greater, since the original library was in the steep non-optimal part of the curve. It has been seen in practice, that this extra delay can often be absorbed during circuit design, especially for non-critical paths. One can, therefore, get the same performance with Library2 and still have overall lower power.

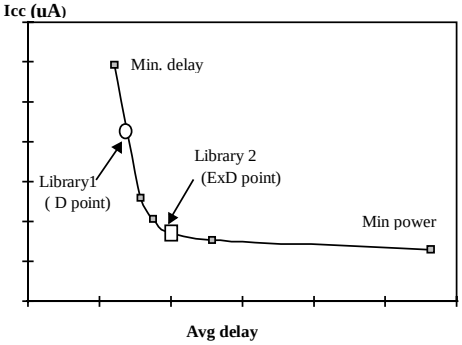

Figure 10: Power-delay curve for differently sized libraries

Figure 11 below shows some of the nodes in a latch cell that may change due to the different optimization criteria.

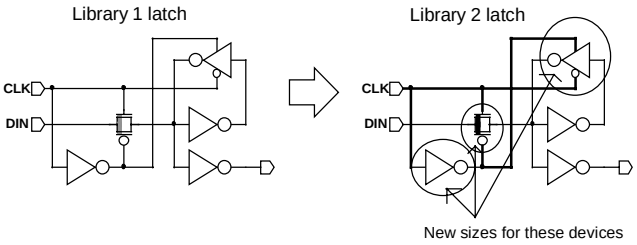

Figure 11: Illustration of energy vs. delay size optimization

The second way to optimize a cell library is to change the schematics of the most commonly used and most power hungry cells in the design. These typically consist of latches and master slave flip flops since these have clock nodes which switch on every clock edge. Figure 12 shows an example of latch redesign that gets rid of clock nodes while still maintaining functionality and performance. Table 1 shows results for some circuits where some sequential cells were replaced with more efficient ones. As can be seen, a small amount of redesign effort on some selected cells can have a significant power impact.

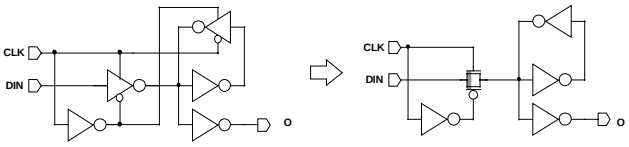

Figure12: Latch redesign example

	Ckt1	Ckt2	Ckt3	Ckt4
Num seq. cells replaced	148	13	18	15
Power change	-13.30%	-9.50%	-12.67%	-3.56%
Area change	0%	8%	1.30%	-1.50%

Table 1: Power Savings from sequential cell redesign

4.4 Power-Delay Curves and Automated Transistor Sizing

As discussed in Section 3, a large part of high-performance CPUs is typically custom designed. These designs typically involve manual tweaking of transistors to upsize drivers in critical paths. If too many transistors are upsized unnecessarily, certain designs can lie on the steep part of a circuit's power-delay curve. In addition, the choice of logic family used, e.g. static vs. dynamic logic, can also greatly influence the circuit's power consumption. Figure 13 illustrates these scenarios for a 32-bit adder. Suppose a designer has the data shown in the figure at his/her disposal. Knowledge of where in the power-delay curve the circuit lies can tell the designer whether he/she can trade a little performance for a larger power savings. In this example, 69% total savings can be gained by transistor sizing and going from domino to static. There is a 23% delay penalty. This extra delay penalty may be overcome by upsizing adjacent blocks at a much less power penalty and still end up with an overall power benefit. Experiments with this methodology have yielded 10% power savings with no delay increase in several circuits. CAD tools that enable this kind of logic and circuit design exploration for custom circuits can thus have a significant impact at the full-chip level.

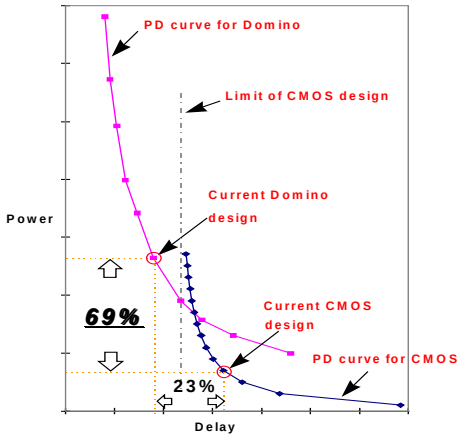

Figure 13: Power-Delay curves for two different 32 bit adders

Figure 14 shows an example of another kind of intelligent tradeoff for power/performance. It shows the results for a logic block whose 7 sub-blocks can either be implemented as PLAs or as synthesized random logic. Increasing the number of synthesized blocks leads to some increase in delay but for much larger power savings.

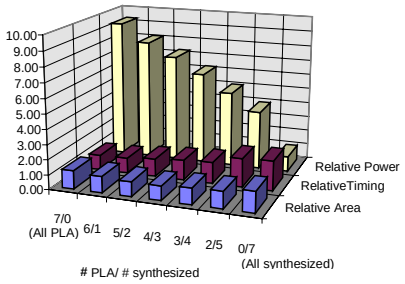

Figure 14: Power/Area/Delay tradeoffs for PLA vs. synth. logic

The traditional emphasis on performance often leads to over-design that is wasteful for power. An emphasis on power, however, motivates identification of such sources of power wastage. An example of this is paths that are designed faster than they ultimately need to be. For synthesized blocks, such paths can be automatically downsized by the synthesis tool. For manually designed blocks, on the other hand, downsizing may not always get done. Automated downsizing tools can thus have a big impact. Transistor width savings (with no delay increase) from the use of one such tool are shown in Table 2. The benefit of such tools is power savings, as well as productivity enhancement for manual designs.

	Ckt1	Ckt2	Ckt3	Ckt4
No of elements	4853	1953	18300	19756
Width Savings	40.00%	42.00%	17.00%	3.00%

Table 2: Transistor width savings with a sizing tool

4.5 Low Power Logic Synthesis

A lot of the CAD research in low power has been in the area of low power logic synthesis. Technology dependent phases of synthesis [8,10] are particularly suited for practical applications, since they have access to low-level circuit information. It is possible to save power with these methods, even when they are constrained to not increase the delay of the circuit. Table 3 shows the results of applying these techniques on a couple of sample circuits from a high-performance CPU. Column 2 shows the power savings when only the combinational part of the circuit is allowed to be changed. Column 3 shows the results when the sequential elements are also allowed to be changed. The area impact is low (Column 5) and there is no increase in delay. The results show ~10% power savings for synthesized blocks, but this translates into only 1% full-chip power savings. This is because only 10% of total power is from synthesized logic for CPUs such as the one whose power breakdown was shown in Figure 4.

Circuit	After LP Synth	After seq. dnsizing	Total	Area
1	1.14%	7.39%	8.52%	-1.85%
2	0.76%	4.89%	5.65%	-0.69%
3	8.40%	1.53%	9.92%	1.99%
4	6.13%	3.31%	9.44%	7.12%
5	11.03%	1.07%	12.09%	1.09%
Total	**5.16%**	**3.81%**	**8.97%**	**0.53%**

Table 3: Low power logic synthesis results

It is pertinent to note that the system power consumption problem also encompasses chipsets, i.e., devices such as the memory, I/O, and graphics controllers. These operate at a fraction of the CPU clock frequency, and large portions of these are well-suited to be implemented as ASICs. Low power synthesis thus has a much larger impact in this domain.

4.6 Specific Circuit Level Techniques

In addition to the techniques described above, specific optimizations on individual circuits can be done to reduce power. Besides the power, area, and delay impact, the use of such ideas depends on factors such as design complexity, schedule impact, noise and reliability issues, etc. A detailed discussion of this class of ideas is beyond the scope of this paper. However, an illustrative example for a domino multiplexer is shown in Figure 14 and is discussed below.

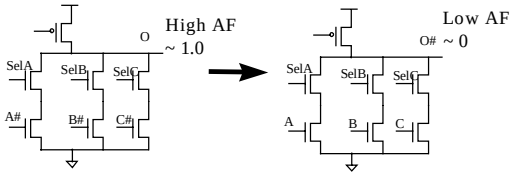

Figure 14: Domino AF reduction in muxes

In this circuit, at least one of *SelA, SelB, SelC* is high every cycle. The data inputs A#, B#, C# are mostly high (A,B,C have very low signal probability). In this situation, all the capacitance associated with the data nodes toggles almost every cycle. This total capacitance is almost 3 times that associated with the clocked nodes. Consequently, when the polarity of the data inputs is changed, as shown on the right, large power savings are seen. The use of a related idea on a large logic block in a recent CPU resulted in an estimated *3%* full-chip power savings.

4.7 System Power Management

The interaction of the CPU with the rest of the system also provides avenues for reducing average power. Often the CPU is waiting for inputs from peripherals and its power is being wasted. To reduce this waste, CPUs are now provided with a hierarchy of power states. Each state defines a certain level of activity on the CPUs and a certain time penalty for it to get back into a fully active state. Memory and I/O devices often also have similar power states. It is the *system power management* mechanism that monitors the system activity and enforces the movement of the system components between different power states [4].

System power management has its roots in mobile systems. However, EPA requirements under the Energy Star program motivated the migration of these techniques to desktop systems. A recent development in this area is a cross-company initiative called ACPI (Advanced Configuration & Power Interface) [1]. The recognition of the need to eliminate wasted power ensures that system power management will continue to be an area of high interest and active development.

4.8 Software Based Power Reduction

Traditionally the focus on low power design has been purely hardware based. This tends to ignore the fact that it is the software that executes on a CPU that determines its power consumption. The software perspective on power consumption has been the subject of recent work [9]. Here a detailed instruction-level power model of the Intel486DX2 was built. The impact of software on the CPU's power and energy consumption, and software optimizations to reduce these were studied. An important conclusion from this work was that in complex CPUs like the 486DX2, software energy and performance track each other, i.e., for a given task, a faster program implementation will also have lower energy. This is because the CPU power consumption is dominated by a large cost factor (clocks, caches, etc.) that for the most part, does not vary much from one cycle to the other.

There are some issues when this work is extended to recent CPUs. First, multiple-issue and out-of-order execution mechanisms make it hard to model power on a "per instruction" basis, and more complex power models are required. Also, increased use of clock gating implies that there is greater variation in power consumption from cycle to cycle. However, it is expected that the relationship between software energy and power that was observed before will continue to hold. In any case, it is important to realize that software directly impacts energy/power consumption, and thus it should be designed to be efficient with respect to these metrics.

A classic example of inefficient software is "busy wait loops". Consider an application such as a spreadsheet that requires frequent user input. During the times when the spreadsheet is recalculating values, high CPU activity is desired in order to complete the recalculation in a short time. In contrast, when the application is waiting for the user to type in values, the CPU should be inactive and in a low-power state. However, a busy wait loop will prevent this from happening, and will keep the CPU in a high-power state. The power wastage is significant. For example, a 166MHz Pentium® Processor with MMX™ technology draws over 7 Watts in normal operation but only 1 Watt when halted. The Intel Power Monitor (IPM) is a publicly available [6] software analysis tool that monitors system activity to provide information about software that may be wasting power in this and other cases.

5. CONCLUSIONS

High-performance CPU design presents unique challenges for research in power related issues. Certain directions here need increased research and development focus in the future.

The highest priority is to continue pushing the voltage scaling treadmill. However, the technological and design hurdles in the path of using sub-1V supply voltages in large, high-performance CPUs have to be removed. Circuit styles and methodologies suited for low voltage are also needed.

Microarchitectures in today's high-performance CPUs are aimed at exploiting ever-increasing amounts of instruction-level parallelism. Organizational choices and tradeoffs are not made with power in mind. This needs to change and power consumption has to become a primary consideration here, since higher levels of design have the greatest leverage on the overall power consumption. Investigation of the hardware-software interface in CPUs will yield additional avenues for power reduction.

As frequencies increase, the number of pipeline stages will most likely increase. Combined with the trend for wider machines, this means that the number of clocked elements will keep increasing. Thus more efficient clock distribution schemes and latches are needed. It also implies that a more aggressive use of clock gating would be required. CAD tools that help in identifying and implementing clock gating opportunities will be useful here. Clocks, high-performance datapaths, and memories are not well-served by existing CAD tools. To have a significant impact on power, CAD tools have to target these domains.

Lower voltages, higher power consumption, more devices, and more clock gating - all these imply that the inductive noise problem will get worse. Increased innovation in packaging and power supplies would be needed to make sure that power delivery does not become the limiting factor for high-performance CPUs.

ACKNOWLEDGMENTS

The authors would like to thank Shekhar Borkar, Ricardo Suarez, Guy Therenin, Michael Walz, Steve Gunther, Shishpal Rawat and Jeff Parkhurst for assistance in writing this paper.

REFERENCES

[1] ACPI home page. *http://www.teleport.com/~acpi*.

[2] S. Borkar, Intel Corp. Personal communication.

[3] A. Chandrakasan, R. Brodersen. Minimizing power consumption in digital CMOS circuits. *Proceedings of the IEEE* 83(4), April 1995.

[4] S. Ellis. Power management in notebook computers. *Proc. Silicon Valley Personal Computing Design Conference*, July 1991.

[5] L. Gwennap. Power issues may limit future CPUs. *Microprocessor Report*, 10(10), August 1996.

[6] Intel Power Monitor home page. *http://developer.intel.com/ial/ipm*.

[7] J. Schutz. A 3.3V 0.6um BiCMOS SuperScalar Microprocessor. *ISSCC Digest of Tech. Papers*, Feb 1994.

[8] V. Tiwari, P. Ashar, S. Malik. Technology mapping for low power. *Proc. Design Automation Conference*, June 1993.

[9] V. Tiwari, S. Malik, A. Wolfe, T.C. Lee. Instruction level power analysis and optimization of software. *Journal of VLSI Signal Processing*, 13(2), August 1996.

[10] C.Y. Tsui, M. Pedram, A. Despain. Technology decomposition and mapping targeting low power dissipation *Proc. Design Automation Conference*, June 1993.