The Energy Efficiency of IRAM Architectures

Richard Fromm, Stylianos Perissakis, Neal Cardwell, Christoforos Kozyrakis,
Bruce McGaughy, David Patterson, Tom Anderson, Katherine Yelick

Computer Science Division, University of California–Berkeley

Abstract

Portable systems demand energy efficiency in order to maximize battery life. IRAM architectures, which combine DRAM and a processor on the same chip in a DRAM process, are more energy efficient than conventional systems. The high density of DRAM permits a much larger amount of memory on-chip than a traditional SRAM cache design in a logic process. This allows most or all IRAM memory accesses to be satisfied on-chip. Thus there is much less need to drive high-capacitance off-chip buses, which contribute significantly to the energy consumption of a system. To quantify this advantage we apply models of energy consumption in DRAM and SRAM memories to results from cache simulations of applications reflective of personal productivity tasks on low power systems. We find that IRAM memory hierarchies consume as little as 22% of the energy consumed by a conventional memory hierarchy for memory-intensive applications, while delivering comparable performance. Furthermore, the energy consumed by a system consisting of an IRAM memory hierarchy combined with an energy efficient CPU core is as little as 40% of that of the same CPU core with a traditional memory hierarchy.

1 Introduction

Energy efficient computing is growing in importance. Sales of laptop and notebook computers have been steadily climbing, and applications for portable computing are likely to continue to grow in the near future, in areas such as PDAs (personal digital assistants), smart phones, GPS (global positioning system) receivers, and other "anywhere-anytime" consumer computing devices [17][10]. The increasing prevalence of portable computing has promoted energy efficiency from a concern primarily of circuit designers to an issue of general interest to the computer architecture community [30][28]. Hence this area has received significant research attention [3][16][4][5][21].

Due to recent advances in flat-panel displays and disk power management, the share of the energy in portable systems consumed by the processor and external memory is growing [20]. Within processors, often a large percentage of energy is devoted to on-chip memory [25][38][39][16].

Our goal is to reduce the energy consumed by the memory system. Integrating a microprocessor and DRAM memory on the same die, an idea that we call Intelligent RAM (IRAM)[31], offers the potential for dramatic improvements in the energy consumption of the memory system. DRAM is much denser than SRAM, which is traditionally used for on-chip memory. Therefore, an IRAM will have much fewer external memory accesses, which consume a great deal of energy to drive high-capacitance off-chip buses. Even on-chip accesses will be more energy efficient, since on-chip DRAM consumes less energy than either SRAM or off-chip DRAM.

Previous work has examined how IRAM can be utilized to improve performance [33][40]. This has been motivated by the exponentially growing performance gap between processors, which are increasing in performance at a rate of 60% per year, and memory access times, which are only getting approximately 7% faster per year [18][46]. An IRAM has both lower latency and higher bandwidth between the processor and the memory compared to a conventional architecture, offering the potential for high performance. A potential challenge to the performance of IRAM implementations is the speed of logic in a DRAM process.

The contribution of this paper is to evaluate to what extent an IRAM design reduces the energy consumption of the memory system. We compare an IRAM to a conventional approach for two different die sizes, using appropriate ranges to account for the potential differences in area between DRAM and SRAM and the speed of a processor in a DRAM process. Using cache simulations of several applications reflective of personal productivity tasks on low-power systems and models of the energy consumption in DRAM and SRAM memories, we calculate the energy consumption of the memory hierarchy and performance for the various architectures. We find that the IRAM memory hierarchy consumes as little as 22% of the energy compared to a conventional implementation for memory-intensive applications, while delivering comparable performance. Furthermore, the energy consumed by a system consisting of an IRAM memory hierarchy combined with an energy efficient CPU core is as little as 40% of that of the same CPU core with a traditional memory hierarchy.

The remainder of the paper is organized as follows: Section 2 discusses various system metrics, including power and energy. Section 3 discusses potential energy and performance benefits of IRAM. Section 4 presents several alternative architectures and our methodology for evaluating them. Section 5 presents the results of our evaluations. Section 6 summarizes related work, Section 7 suggests future work, and Section 8 presents our conclusions. A more detailed discussion of the energy consumption models used

This research was supported by DARPA (DABT63-C-0056), the California State MICRO Program, and by research grants from Intel and Sun Microsystems. The authors can be contacted at `http://iram.cs.berkeley.edu` or {`rfromm, sper, neal, kozyraki, brucemcg, patterson, tea, yelick`} `@cs.berkeley.edu`.

To appear in the 24th Annual International Symposium on Computer Architecture (ISCA '97), Denver, Colorado, June 2-4 1997.

for our evaluations is included as an Appendix.

2 Metrics

Various metrics provide different perspectives on the characteristics of a portable computer system.

2.1 Power

Traditionally, the unit of concern for portable devices has been power, measured in Watts. Power represents the amount of energy consumed per unit time: $Power = Energy/Time$.

Power is a useful metric for certain concerns. The amount of power required by a system determines, for a fixed voltage, how much current a battery or other power source must be able to deliver. Also, higher power leads to high temperature, which in turn leads to more expensive packaging and a cooling system that can be more expensive, larger, and heavier.

2.2 Energy

As anyone who has ever used a portable computer can attest, the overriding concern for a user of such a system is battery life. Battery life is measured in units of energy, not power. Energy is the product of power and time, i.e., $Energy = Power \cdot Time$. For a given amount of work, what matters most to the user is how much energy is required to do that work.

Thus energy efficiency, measured in energy consumed per instruction, or MIPS per Watt, is a better metric than power for measuring how a given machine best utilizes limited battery life [6][45][16]. Energy per instruction and MIPS per Watt are inversely proportional to each other, as can be seen from the following relationship:

$$\frac{Energy}{Instruction} = \frac{Energy/Sec}{Instruction/Sec} \propto \frac{Watts}{MIPS}$$

Power can be a deceiving metric, since it does not directly relate to battery life. For instance, if the clock rate of a processor is cut in half, then the processor will consume approximately half the amount of power, assuming that the voltage is held constant, since $Power = Frequency \cdot Capacitance \cdot Voltage^2$. However, if all else is kept equal, it will now take approximately twice as long to execute a sequence of instructions. Thus the energy consumed by the processor for some given amount of work will be roughly the same.[1] Even worse, since the task will take longer, the energy consumed by the display and other components of the system will be greater.

At the extreme, slowing the frequency down to zero and putting the processor in sleep mode wins if processor power consumption is the only metric, yet this allows no work to be accomplished.

2.3 Performance

Besides energy efficiency, users are also concerned about performance. Users of portables are willing to tolerate lower performance than a desktop system has, but performance still matters. Portable systems are being used increasingly for more demanding applications that require higher performance. Examples include preparing large, graphical presentations, audio and video playback, handwriting recognition, and speech recognition. The goal is to be able to deliver high performance while still maintaining energy efficiency.

[1] Reducing the clock rate may also make it possible to lower the voltage [45], which would reduce both energy and power consumption, at the cost of decreased performance.

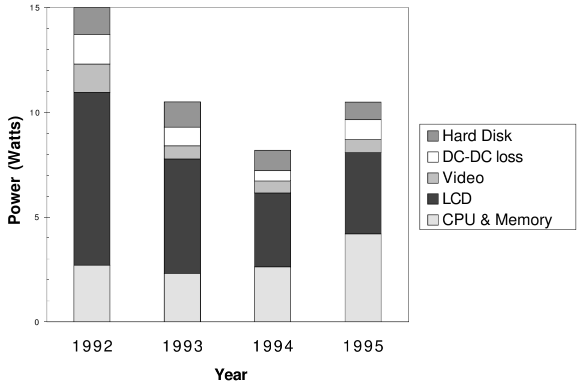

Figure 1: **Notebook Power Budget Trends**

In general, performance is measured by the execution time of a workload. However, we use MIPS to compare the performance of different systems on a given application since we limit our consideration to a single executable for a single instruction set architecture[18].

3 Benefits of Processor-Memory Integration

3.1 Energy Components of Current Systems

Given that portable devices, and hence energy efficient systems, are becoming more prevalent, it is useful to examine where the energy is consumed in such a device. Figure 1 shows the breakdown of the power consumption over time in IBM ThinkPad notebook computers [20]. Whereas the power used to be dominated by the screen, over time the CPU and memory are becoming an increasingly significant portion of the power budget. In smaller handheld portable devices, such as the Apple Newton and U.S. Robotics Pilot, there is no disk and the screen consumes much less power. The LCD on the original Newton consumed only 5 mW for static images, for example [6]. Hence, for these systems the power consumption of the CPU and memory is an even larger fraction of the total.

There are three parts to the portion labeled "CPU and Memory" in Figure 1: CPU core, on-chip caches, and external memory. The introduction of low power CPU cores places an even greater emphasis on the energy consumed by the memory system, both on-chip and off-chip. Considering only the on-chip costs, several studies show a roughly equal division of power or energy (different studies used different metrics) between CPU and memory.

StrongARM [25][38], a microprocessor by Digital that implements the ARM instruction set, delivers 183 Dhrystone MIPS at 160 MHz while dissipating less than 0.5 W. The 32 KB of on-chip caches consume 43% of the CPU power.

Sato, Nagamtsu, and Tago [39] used a power simulator of a generic RISC processor, examining both the current drawn by the various components and the percentage of time those components are used. Their results indicate that the instruction and data caches consume 60% of the total processor power.

Gonzalez and Horowitz [16] show in their simulations that 25-40% of the energy on a microprocessor is dissipated by the on-chip caches. They note that although each individual memory cell dissipates very little energy, the total represents a very significant sum. They are pessimistic that the combined goals of energy efficiency and performance can improve significantly since the energy is dissipated in so many small units. They feel that only a "radical new architecture" can significantly improve this situation.

die size	Small		Large	
implementation	Conventional	IRAM	Conventional	IRAM
CPU frequency	160 MHz	120 MHz *(0.75* **X***)* to 160 MHz *(1.0* **X***)*	160 MHz	120 MHz *(0.75* **X***)* to 160 MHz *(1.0* **X***)*
technology	0.35 μm logic	0.35 μm DRAM	0.35 μm logic	0.35 μm DRAM
L1 config	16 KB I + 16 KB D	8 KB I + 8 KB D	8 KB I + 8 KB D	8 KB I + 8 KB D
L1 associativity	32-way	32-way	32-way	32-way
L1 write policy	write-back	write-back	write-back	write-back
L1 block size	32 Bytes	32 Bytes	32 Bytes	32 Bytes
L1 type	SRAM on-chip	SRAM on-chip	SRAM on-chip	SRAM on-chip
L1 access time	1 cycle	1 cycle	1 cycle	1 cycle
L2 config	–	256 KB *(16:1)* to 512 KB *(32:1)* unified	256 KB *(32:1)* to 512 KB *(16:1)* unified	–
L2 associativity	–	direct-mapped	direct-mapped	–
L2 write policy	–	write-back	write-back	–
L2 block size	–	128 Bytes	128 Bytes	–
L2 type	–	DRAM on-chip	SRAM on-chip	–
L2 access time	–	30 ns	3 cycles (18.75 ns)	–
main memory	8 MB DRAM off-chip	8 MB DRAM off-chip	8 MB DRAM off-chip	8 MB DRAM on-chip
memory latency	180 ns	180 ns	180 ns	30 ns
bus width	narrow (32 bits)	narrow (32 bits)	narrow (32 bits)	wide (32 Bytes)

Table 1: **Architectural Models Used for Evaluation.** The SMALL and LARGE models roughly correspond to the die sizes given in Table 2. It is economically feasible to build large memory arrays by using redundancy. StrongARM's caches, in contrast, use no redundancy. Memory capacity and speed differentials between the CONVENTIONAL and IRAM models, including the variations in CPU frequency and L2 cache size, are based on the arguments presented in Sections 4.1 and 4.2 respectively. All caches are write-back to minimize energy consumption from unnecessarily switching internal and/or external buses. The 30 ns on-chip DRAM access time is based on [24], the 180 ns off-chip access time is based on [11], and the 18.75 ns on-chip L2 SRAM cache access time is chosen to be slightly larger than the on-chip L2 cache of the Alpha 21164A [8], which is slightly smaller (96 KB). The "narrow" bus width matches StrongARM, and the "wide" bus width takes advantage of the additional bandwidth available to main memory in an IRAM configuration. Note that it is only sensible to perform comparisons between SMALL-CONVENTIONAL and SMALL-IRAM and between LARGE-CONVENTIONAL and LARGE-IRAM. The SMALL and LARGE models correspond to different die sizes and are not meant to be compared to one another.

3.2 IRAM Energy Benefits

Although the concept of IRAM may represent a somewhat "radical" implementation, the integration of processor and memory has the potential to improve energy efficiency with a relatively simple system architecture.

There are numerous ways in which processor and memory can be integrated: 1) putting SRAM memory with a CPU in a logic process; 2) putting CPU logic with memory in a DRAM process; 3) putting DRAM memory with a CPU in a logic process. The first idea represents the conventional approach, using SRAM caches for on-chip memory. The second idea is what we are suggesting for IRAM. The third idea has a number of technological disadvantages, including a significant loss in memory density and a much higher refresh rate for the DRAM, due to DRAM process optimizations which are not present in a logic process [12]. Therefore, we only consider the first two approaches.

IRAM has a number of advantages over a conventional approach in achieving energy efficiency. First, DRAM is more energy efficient than SRAM per memory access, so accesses to the on-chip memory consume less energy. More importantly, since DRAM is much denser (see Section 4.1), more memory can reside on-chip, allowing some applications to become entirely chip-resident and significantly reducing the frequency of off-chip accesses for other applications. Driving high-capacitance off-chip buses requires a large amount of energy, so significantly reducing the number of off-chip accesses dramatically reduces the overall energy consumption.

3.3 IRAM Performance Benefits and Challenges

IRAM also has the potential for higher performance than a conventional approach. Even without considering architectural models specifically tailored to exploit the low-latency, high-bandwidth characteristics possible from an IRAM organization, the significant density improvement for DRAM over SRAM can result in higher performance for IRAM. As with the energy efficiency advantages, the big win for IRAM is that many more memory accesses can stay on-chip, which can significantly reduce the average memory access time. However, there are two limitations which might offset this benefit. First, access to a DRAM array is somewhat slower than an SRAM array. Second, logic in a DRAM process may initially be slower than logic in a state-of-the-art logic processes, leading to a slower CPU (see Section 4.2). The improvements due to the reduction in off-chip memory accesses will have to be greater than these slowdowns for there to be an overall increase in performance.

4 Methodology

To quantitatively investigate the energy efficiency and performance merits of IRAM, we compared the architectures listed in Table 1. The models assume commercially available logic and DRAM semiconductor processes. We use the models to estimate the behavior of a small, conventional CPU in a logic process (SMALL-CONVENTIONAL); that same CPU implemented in a DRAM process (SMALL-IRAM); a large DRAM die with a CPU added (LARGE-IRAM); and a similarly large die implemented in a logic process (LARGE-CONVENTIONAL).

In order to determine the characteristics of the models used to evaluate IRAM versus conventional architectures, two questions that need to be addressed are the area differences of DRAM and SRAM and the speed differences of logic in a DRAM versus logic process.

	StrongARM	64 Mb DRAM
process	0.35 μm CMOS	0.40 μm CMOS
memory cell size	26.41 μm^2	1.62 μm^2
number of memory bits	32 KB + tags = 287,744	64 Mb = 67,108,864
total chip area	49.9 mm^2	186.0 mm^2
total area of memory	27.9 mm^2	168.2 mm^2
Kbits per mm^2	10.07	389.6

Table 2: **Memory Cell Parameters For Typical Microprocessor [25][37] and DRAM Chips[24]**

4.1 Area Differences

As mentioned previously, the density of a DRAM array in a DRAM process is *much* higher than that of SRAM in a logic process. The ratio of the number of cells per unit area for DRAM versus SRAM is much greater than the 4:1 or 6:1 figure that one would assume if the only relevant factor was the number of transistors, four or six for an SRAM cell versus a single transistor for a DRAM cell [12]. DRAM storage cells use pseudo-3-dimensional trench or stacked capacitors to achieve very small cell sizes [35]. As Table 2 shows, the DRAM cell size for a 64 Mb DRAM implemented in a 0.4 μm CMOS process [24] is 16 times smaller than the SRAM cell size for StrongARM [37]. If the DRAM feature size is scaled down so that the comparison is for the same size process (0.35 μm), then the cell size is 21 times smaller.

What is more important, however, is to compare the total amount of memory that can fit in a given area when all circuits and interconnect that make up the entire memory array are taken into account. Examining the cell efficiency (bits of memory per unit area) shows that the 64 Mb DRAM is effectively 39 times more dense than the StrongARM. After again scaling the DRAM parameters to make an equal process comparison, the DRAM is 51 times more dense!

These numbers can be used to obtain an approximate value for the differences in memory capacity of a conventional and an IRAM architecture, given a fixed chip area. However, it is difficult to precisely quantify this for all cases, due to a number of extenuating circumstances. The comparisons above use chips with very different die areas, and the actual ratio obtained between two implementations is somewhat dependent on implementation-specific design decisions as well as the absolute size of the array – it is easier to make a memory array denser as it gets larger. Also, logic circuits in a DRAM process will be somewhat larger than logic circuits in a corresponding logic process. Therefore, to obtain equal total area, somewhat less area will be available for the memory array. This situation is likely to improve as future DRAM generations add additional metal layers, enabling more optimal layout of logic. Recognizing these factors, we are conservative when picking DRAM to SRAM capacity ratios used in the models, and instead of limiting our analysis to a single value, we use a range. The bounds of this range are obtained by rounding down the cell size and bits per unit area ratios to the nearest powers of 2, namely 16:1 and 32:1.

4.2 Speed Differences

Logic processes are optimized for fast transistors and interconnect, and DRAM processes are optimized for density and retention. Due to these different process characteristics, logic gates are currently slower in a standard DRAM process than a standard logic process, resulting in a CPU speed that may be slower in an IRAM than a conventional processor [19]. The precise slowdown varies by the manufacturer and the process generation. While logic circuits implemented in a DRAM currently have a speed disadvantage, ongoing trends in the DRAM industry will likely alleviate this difficulty. DRAM processes are beginning to incorporate faster transistors to support synchronous DRAM, and some DRAM manufacturers are already developing merged logic and DRAM processes in an attempt to get the best features of both on a single die. A panel of DRAM experts at the 1997 ISSCC [23] agreed that soon it will be possible to achieve the same performance from logic transistors in a DRAM process compared to a logic process, albeit at a modest (20-30%) increase in cost per wafer.

To address the question of transistor speeds in DRAM versus logic processes, we calculate our performance results for a range of CPU speeds for the architectures implemented in DRAM technologies. We vary the CPU speed from 0.75 as fast to equal in speed to the architectures implemented in logic processes. We expect that the low end of the range characterizes what an IRAM implementation might face today, while the high end of the range reflects DRAM processes likely to be available in the future.

4.3 Architectural Models

The SMALL-CONVENTIONAL design is architecturally similar to StrongARM [25][38], a low-power implementation of the ARM architecture by Digital that delivers 183 Dhrystone MIPS while dissipating under 0.5 W at 160 MHz.[2]

The SMALL-IRAM model addresses what the SMALL-CONVENTIONAL CPU would look like if implemented in a DRAM process given current technology, and if the overall chip area was kept constant. Since a single-cycle first level cache access is desirable for good performance, and access to an on-chip DRAM array is slower than that, we chose *not* to simply change the 32 KB of SRAM cache into a DRAM cache of similar area for the SMALL-IRAM model. Instead, we split the area originally allocated for the cache into two. Half of the area is allocated to a conventional L1 SRAM cache, and the remaining area is implemented as DRAM, organized as an L2 cache.

We consider L2 cache sizes of 256 KB and 512 KB, which corresponds to a density increase for DRAM compared to SRAM of 16:1 and 32:1 respectively (see Section 4.1). We consider CPU speeds between 120 and 160 MHz, which cover a range of speeds for logic in a DRAM process from 0.75 as fast to no slowdown (see Section 4.2).

The LARGE-IRAM model approaches the problem from a different angle: instead of starting with an existing CPU architecture and modifying it to be implemented in a DRAM process (with comparable total area), we start with a DRAM chip and add a CPU. We choose a 64 Mb DRAM chip, which is roughly comparable to 0.35 μm logic technology; both represent the state of the art available today in commercial implementations. This gives 8 MB of DRAM memory on-chip, which is large enough to be considered the main memory for many applications, especially in portable systems. We therefore assume that the on-chip DRAM array is main memory rather than an L2 cache. All memory references are assumed to be satisfied on-chip. Just as in the previous case (SMALL-IRAM), we desire for the CPU to be able to access most memory references in 1 cycle, so we again add a first level SRAM cache (again, 8 KB I + 8 KB D). Based on the same arguments as before (see Section 4.2), we consider CPU speeds of 120 and 160 MHz to address the potential speed differential between logic in DRAM and logic processes.

The LARGE-CONVENTIONAL model rounds out the four architectures studied by assuming roughly the same die area as the LARGE-IRAM model, but using a logic process instead. The large, on-chip memory array is therefore composed of SRAM cells. Based on the

[2]While 32-way set associativity may seem somewhat excessive, the StrongARM designers note that only 4-way associativity was desired for performance goals. Additional design requirements of single cycle access and bank selection led to the highly associative cache as a result. See [38] for details. We choose to keep the same 32-way set associativity for the L1 cache for all of our models to enable fair comparisons.

benchmark	instructions	16K L1 I miss	16K L1 D miss	% mem ref	description
hsfsys	1.8 billion	0.01%	5.2%	27%	Form-based handwriting recognition system; 1 page (55 MB)
noway	83 billion	0.02%	5.7%	31%	Continuous speech recognition system; 500 words (20.6 MB)
nowsort	48 million	0.0031%	6.9%	34%	Quicksorts 100-byte records with 10-byte keys (6 MB)
gs	3.1 billion	0.70%	3.0%	22%	Postscript interpreter; 9-chapter text book (7 MB)
ispell	26 billion	0.02%	2.0%	13%	Spelling checker; histories and tragedies of Shakespeare (2.9 MB)
compress	49 billion	0.000003%	9.3%	30%	Compresses and decompresses files; 16 MB
go	102 billion	1.3%	3.0%	31%	Plays the game of Go against itself three times
perl	47 billion	0.33%	0.63%	38%	Manipulates 200,000 anagrams and factors 250 numbers in Perl

Table 3: **Benchmarks and Data Sets Used For Evaluation.** Hsfsys [14] is from the National Institute of Standards and Technology (NIST), and noway [36] was written at the University of Sheffield. Nowsort was developed at the University of California-Berkeley. Ghostscript (gs) and ispell are well-known utilities. The final three benchmarks are from the SPECint95 benchmark suite [42]. Cache miss rates are for the SMALL-CONVENTIONAL model only and are meant, along with the percentage of instructions which are memory references (loads/stores), to give an overview of the memory behavior of each program.

	DRAM	SRAM	
		L1	L2
Internal power supply	2.2V	1.5V	1.5V
Bank width	256 bits	128 bits	128 bits
Bank height	512 bits	64 bits	512 bits
Bit line swing (read)	1.1V	0.5V	0.5V
Bit line swing (write)	1.1V	1.5V	1.5V
Sense amplifier current	–	150μA	150μA
Bit line capacitance	250fF	160fF	1280fF

Table 4: **Major Technology Parameters Used in Memory Hierarchy Models**

same arguments as before (see Section 4.1), we model both a 16:1 and 32:1 ratio of DRAM to SRAM array areas. This gives SRAM array sizes of 512 KB and 256 KB respectively. This is unlikely to be adequate for main memory, even for portable applications. Consequently, we assume that this is treated as an L2 cache, with conventional (off-chip) DRAM main memory as the next level in the memory hierarchy.

4.4 Simulations

Table 3 shows the applications we used for our evaluations. Each of these benchmarks is representative of current or potential future applications for a portable computing device. For each of these benchmarks and each of the architectural models in Table 1 we calculated the performance of the system as well as the energy consumed by the memory hierarchy, including caches, memory buses, and main memory.

The benchmarks were simulated using the multilevel cache simulator `cachesim5` included with `shade` [43], a tool from Sun which integrates instruction set simulation and trace generation with custom trace analysis. Activity at each of the levels of the memory hierarchy was recorded. Additionally, the base cycles per instruction (CPI), as if there were no stalls due to memory references, was determined using `spixcounts` and `ifreq`, dynamic instruction frequency profiling utilities also included with `shade`. Final performance numbers were computed by combining the base CPI with the miss rates and latencies at the various levels of the memory hierarchy.

The models for memory system energy consumption capture the dominant effects in DRAM and SRAM memories. Technology parameters were taken from typical DRAM circuits of the 64 Mb generation [24][47][44][27] and contemporary microprocessor SRAM cache generations [11][26][9] (see Table 4), which represent the state of the art commercially available in 1997. See the Appendix for a more detailed explanation of how energy consumption was calculated.

Our CPU model is similar to StrongARM, a single-issue, in-order CPU that achieves remarkable energy efficiency. The off-chip latency is the time to return the critical word. The CPU initially stalls on cache read misses, then continues execution while the rest of the cache block is fetched. We assume a write buffer big enough so that the CPU does not have to stall on write misses.

It should be noted that our energy models only address the energy consumed by the memory hierarchy, and not by the CPU core. Previous work [3][10] has already addressed techniques for reducing the CPU energy consumption; it is presumed that any system employing IRAM for reasons of energy efficiency will employ many of these techniques. Section 5.1 compares our results for the energy consumption of the memory hierarchy to the energy consumption of a low power CPU.

5 Evaluation

5.1 Energy

Figure 2 shows the energy consumption of the memory hierarchy for each of the benchmarks for each of the models. The energy consumption is divided into its various components, including L1 instruction and data caches, L2 cache, main memory, and the energy to drive the buses to access the various levels. Only a single value is given for each configuration, including the DRAM configurations where the CPU speed is varied, since the energy consumed by the memory system, for a given voltage, does *not* depend on CPU frequency.

As can be seen from the figure, the various IRAM configurations can significantly reduce the amount of energy consumed per instruction compared to the corresponding conventional cases. For the small chips, the memory hierarchy of the IRAM architectures consumes as little as 29% of the energy of the corresponding conventional architectures; for the large chips IRAM consumes as little as 22% of the conventional cases. In the worst case, the energy consumption of the IRAM memory hierarchy is comparable to that of the conventional models – either 116% or 76% for the small and large chips respectively. Note that the valid comparisons here are between the SMALL-CONVENTIONAL and SMALL-IRAM models, and between the LARGE-CONVENTIONAL and LARGE-IRAM models. The SMALL and LARGE models correspond to different die sizes and are not meant to be compared to one another.

The results can be understood by considering the following equation that relates the energy consumption at each level of the memory hierarchy with the frequency of accesses to the various

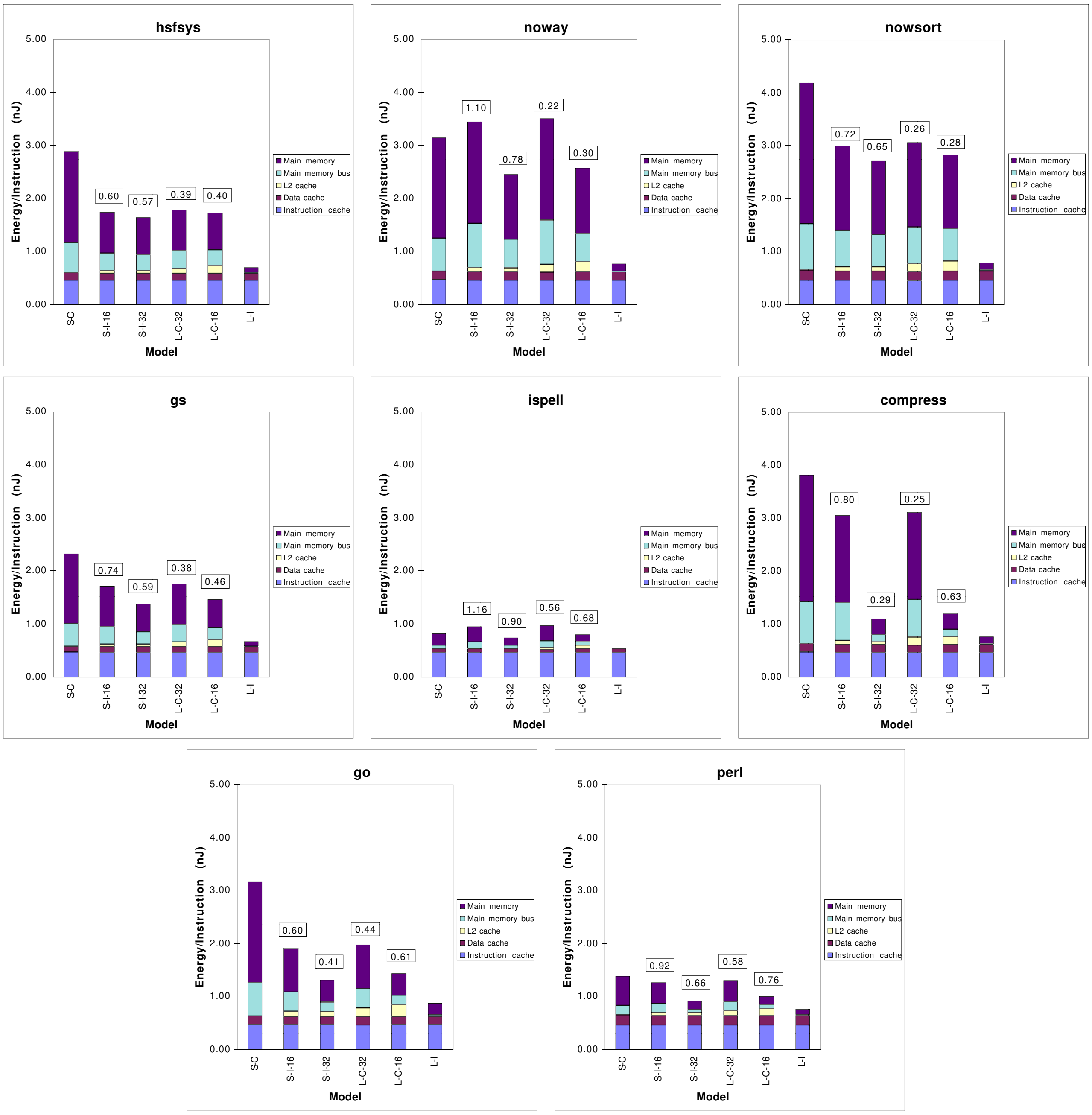

Figure 2: **Energy Consumption of Memory Hierarchy.** The labels for the models are as follows: S-C = SMALL-CONVENTIONAL; S-I-16 = SMALL-IRAM with 16:1 DRAM to SRAM-cache area density ratio (i.e. 256 KB L2 cache); S-I-32 = SMALL-IRAM with 32:1 ratio (512 KB L2); L-C-32 = LARGE-CONVENTIONAL with 32:1 ratio (256 KB L2); L-C-16 = LARGE-CONVENTIONAL with 16:1 ratio (512 KB L2); and L-I = LARGE-IRAM. The values atop the IRAM bars show the ratios of energy consumption compared to the CONVENTIONAL implementations. Ratios less than 1.0 indicate that IRAM consumes less energy per instruction.

	Small		Large	
	Conventional	IRAM	Conventional	IRAM
L1 access	0.447	0.447	0.447	0.447
L2 access	–	1.56	2.38	–
MM access (L1 line)	98.5	–	–	4.55
MM access (L2 line)	–	316	318	–
L1 to L2 Wbacks	–	1.89	2.71	–
L1 to MM Wbacks	98.6	–	–	4.65
L2 to MM Wbacks	–	321	323	–

Table 5: **Energy (in nanoJoules) Per Access to Levels of Memory Hierarchy.** Note that this table is somewhat of an approximation. For instance, the L2 cache access values vary somewhat depending on whether the access is a read or a write, as well as on the size of the cache. The average is shown.

levels (see Table 5). It is closely modeled after the familiar equation for average memory access time [18]:

$$\begin{aligned} &\textit{Energy per instruction} = \\ &\quad AE_{L1} + (MR_{L1} \times (1 + DP_{L1}) \\ &\qquad \times (AE_{L2} + (MR_{L2} \times (1 + DP_{L2}) \\ &\qquad\qquad \times AE_{\textit{off-chip}}))) \end{aligned}$$

where AE = access energy,
MR = miss rate,
and DP = dirty probability.

It is clear from this equation that there are two ways to reduce the energy consumption per instruction: 1) reduce the energy to access various levels of the memory hierarchy; 2) reduce the frequency of accesses to lower levels (i.e. reduce the miss rate). IRAM has the ability to deliver both.

There are two major sets of reductions of energy per access for a given level of an IRAM memory hierarchy – the differences between DRAM and SRAM and the differences between on-chip and off-chip accesses. For accesses that hit in the second level cache, accessing a DRAM array is more energy efficient than accessing a much larger SRAM array of the same capacity (see Table 5), mostly because the interconnect lines are shorter and the related parasitic capacitances are smaller. More striking is the comparison between on-chip and off-chip main memory, which is DRAM in both cases. Having the DRAM on-chip saves energy in three ways. First, accesses to the high-capacitance, off-chip bus are avoided. Second, with the multiplexed address scheme of conventional DRAMs, the short row address will select a larger number of DRAM arrays than needed to deliver the desired number of bits. On an IRAM, the entire address is available at the same time, which allows the minimum required number of arrays to be selected.[3] Finally, an external DRAM with a narrow pin interface will need to go through a number of column cycles to deliver an entire cache block, using additional energy to decode the column address and drive the long column select lines and multiplexers in every cycle. This energy is saved with an on-chip DRAM, which can deliver the entire cache line in one cycle.

The big win for IRAM comes from reducing the frequency of accesses to lower levels of the memory hierarchy. By having a DRAM array on-chip instead of an SRAM array, the on-chip memory can be *much* larger. Consequently, the IRAM configurations will have much lower off-chip miss rates and will not have to pay the significant energy penalty for going off-chip as frequently. For example, the off-chip (L1) miss rate for the go benchmark is 1.70% on the SMALL-CONVENTIONAL resulting in an off-chip energy cost of 2.53 nanoJoules/instruction and a total memory system energy consumption of 3.17 nJ/I. For the SMALL-IRAM case with a 32:1 DRAM to SRAM density ratio, although the local L1 miss rate rises to 3.95% (the L1 caches are only 8 KB each instead of 16 KB), the large L2 cache reduces the global off-chip (L2) miss rate to 0.10%. This contributes to the result of an off-chip energy cost of 0.59 nJ/I and a total memory system energy consumption of 1.31 nJ/I. These are respectively 23% and 41% of the conventional values. In the LARGE-IRAM case, where the main memory array is on-chip, all memory accesses can be satisfied without paying this high energy cost, offering the potential to even further reduce the energy consumption. The degree of improvement for IRAM depends on the nature of the application. Memory-intensive applications are much more likely to benefit by having access to much more memory than compute-intensive applications. If an application already fits within the available on-chip memory in a conventional approach, having still more memory will not provide a significant benefit.

There are some minor offsetting factors. For example, the SMALL-IRAM configuration has an L1 cache that is half of the size of the SMALL-CONVENTIONAL configuration, giving it a higher L1 miss rate and forcing it to access its next level (L2 DRAM cache) on some occasions in which the SMALL-CONVENTIONAL case hits in its L1 cache. This factor is small enough compared to the savings from going off-chip less often that, in most cases, there is a significant reduction in the energy consumption of the memory hierarchy by integrating the processor with DRAM.

Another offsetting factor arises from the particulars of the architectural models that we chose for our simulations. The L1 cache block sizes are 32 Bytes, while the L2 block sizes are 128 Bytes. As a result of this, main memory accesses that have to perform a cache fill consume more energy on the SMALL-IRAM model than they do for SMALL-CONVENTIONAL(see Table 5). This causes some anomalous cases (See noway and ispell in Figure 2) in which the energy consumption of the memory hierarchy for an IRAM implementation is actually greater than for a corresponding conventional model. This illustrates that the choice of block size is important for energy efficiency. While there has been a trend over time towards larger block sizes, fetching potentially unneeded words from memory may not be the best choice, depending on the memory access patterns of a given application, when energy consumption is taken into account.

By comparing our energy results to some known values from StrongARM, we can perform a quick validation of a portion of our energy consumption models. StrongARM dissipates 336 mW while delivering 183 Dhrystone MIPS. Of this, 27% of the power consumption comes from the ICache [25]. This translates into 0.50 nanoJoules per instruction. The energy consumption of the ICache in our simulations is fairly consistent across all of our benchmarks, at 0.46 nJ/I.

Our results presented so far only include the energy consumption of the memory hierarchy. Using an analysis similar to that above, we can place our results in the context of the energy consumption of a CPU combined with memory. As stated earlier, the on-chip caches on StrongARM consume 43% of the power, leaving 57% for the CPU core. Based on the same 336 mW and 183 MIPS figures noted above, this translates into 1.05 nanoJoules per instruction. For a memory-intensive application, this is a small portion of the energy consumed by the CPU and external memory. Thus, improving the energy of the memory hierarchy leads to a noticeable reduction in the energy dissipated by the CPU and memory. For example, for noway, comparing the energy consumption of LARGE-CONVENTIONAL (with 32:1 DRAM to SRAM density ratio)

[3] This might mean a corresponding increase in the number of cycles needed to refresh the entire memory, but with a minor increase in complexity an on-chip DRAM could separate the refresh operation from the read and write accesses and make it as wide as needed to keep the number of cycles low.

to LARGE-IRAM, and adding 1.05 nJ/I for the CPU core, shows that IRAM (1.82 nJ/I) uses only 40% of the energy of the conventional model (4.56 nJ/I). For more compute-intensive applications, the energy consumed by the CPU dominates that of the memory hierarchy. However, even for compute-intensive applications, the memory hierarchy is still relevant and represents a significant portion of the total energy consumption, especially when on-chip accesses are considered. Even if an application is entirely cache-resident, some energy will be consumed to access the caches.

5.2 Performance

While the primary motivation for this study is energy efficiency, it is also important to ensure that energy efficiency can be obtained while demonstrating comparable performance.

Table 6 shows the performance of each of the models, assuming a 32:1 DRAM to SRAM-cache area density ratio, on each of the benchmarks. This is shown for the range of frequencies for the CPU core implemented in a DRAM process compared to a logic process discussed earlier, from 0.75 times as slow for the CPU in a logic process to equal in speed. For the small chips, the IRAM architectures range from 0.78 to 1.50 times the performance of the corresponding conventional cases. For the large chips, IRAM ranges from 0.76 to 1.09 times the performance.

There are two opposing factors that determine the speed of the IRAM architectures relative to the conventional architectures. Which of these factors will dominate is a function of both the physical organization and the application. For an application that is compute-intensive and is not heavily dependent on the memory hierarchy, it is possible that a naive IRAM architecture will have lower performance than a conventional processor. The processor's operating frequency may initially be limited by the DRAM process in which it is implemented, and a compute-intensive application will likely be adversely impacted by such a slowdown.

For an application that is heavily dependent on the memory hierarchy and is already not fully utilizing the CPU, however, IRAM has the potential for a large performance gain. Many more of the memory accesses can be satisfied in the low-latency, high-bandwidth on-chip memory, due to the much higher density of DRAM than SRAM. For instance, Cvetanovic and Bhandarkar [7] found that a 300 MHz Alpha 21164 microprocessor spends about 75% of its time in the memory hierarchy for database and matrix computations. As the performance gap between processors and memory continues to increase, as applications have more demanding memory requirements, and as DRAM capacities continue to increase beyond the 64 Mb used in this study, the performance advantages of IRAM will grow.

The performance results of this study are, unfortunately, not impressive. One important conclusion from this is that, for these benchmarks, an IRAM implementation of a conventional architecture is not likely to lead to dramatic performance gains. This is consistent with other preliminary IRAM investigations [33][32][22]. This illustrates the importance of investigating new performance-oriented architectural ideas and organizations that can take better advantage of the lower latency and dramatically higher bandwidth between the processor and memory that IRAM offers.

6 Related Work

Other researchers have investigated a close integration of processors and memory. However, previous discussions have concentrated on the potential performance benefits. This paper is the first to quantify the energy efficiency advantages of IRAM.

Commercial products integrating DRAM and logic include graphics accelerators from NeoMagic [29] and Accelerix [1], and a chip from NEC that combines 16 Mb of DRAM with 128 8-bit processors for image-processing applications [2]. In addition, Mitsubishi has announced the M32R/D, which integrates a processor, 2KB of SRAM cache, and 2 MB of DRAM and is targeted at personal digital assistants [41][22]. They state that integrating a processor with memory significantly reduces power dissipation.

Researchers at Sun [40] evaluated the performance of a RISC processor on a 256 Mb DRAM, using the sense amps as caches. They arbitrarily limited their additions to be about 10% of the DRAM size, and found that they could achieve comparable integer performance and about half the floating point performance compared to a 1995 DEC Alpha. Other research projects investigating processor-memory integration include the Galileo project at the University of Wisconsin [13], the PPRAM project at Kyushu University in Japan [34], and the Processor-in-Memory Technology Infrastructure Development project at the University of Notre Dame.

7 Future Work

There is much more work to be done in this area, concerning both low level circuit issues and high level architectural issues. The physical implications (including temperature and noise) of closely integrating logic and memory need to be studied. For instance, as a rule of thumb, for every increase of 10 degrees Celsius, the minimum refresh rate of a DRAM is roughly doubled [15]. Research in process development would be useful in this and other areas. Perhaps the best realization of processor-memory integration can be achieved in a hybrid CMOS process that incorporates the best features of both logic and DRAM processes.

Also, as we suggested in Section 5.1, it would be useful to quantify the energy dissipation impact of cache design choices, including block size and associativity.

Finally, an IRAM organization gives us the opportunity for both lower latency and dramatically higher bandwidth between the processor and memory. Investigating new ideas and organizations that can turn these opportunities into significantly increased performance is an exciting and open area of research in architecture as well as compilers and operating systems. While this study concentrated on the benefits of IRAM for energy-conscious systems, this is certainly not the only realm in which IRAM may play an important role in redefining our notion of processor and memory system design.

8 Conclusion

We have quantified the energy efficiency advantages of IRAM memory hierarchies, relative to traditional memory hierarchies, by applying models of energy consumption in DRAM and SRAM memories to results of cache simulations of several applications reflective of personal productivity tasks on low power systems.

We found that IRAM memory hierarchies consume as little as 22% of the energy consumed by a conventional memory hierarchy (with on-chip L1 and L2 caches) for memory-intensive applications. They do so while delivering comparable system performance. When placed in the context of the energy dissipated by a high performance, low power CPU, we found that the energy consumed by an IRAM memory hierarchy combined with an energy efficient CPU core is as little as 40% of that consumed by that same CPU core combined with a traditional memory hierarchy.

Acknowledgments

We are grateful to Dan Dobberpuhl, Dan Murray, and Sribalan Santhanam of Digital Equipment Corporation for their helpful an-

benchmark	Small			Large		
	Conventional	IRAM (.75 **X**)	IRAM (1.0 **X**)	Conventional	IRAM (.75 **X**)	IRAM (1.0 **X**)
hsfsys	138	112 *(0.81)*	150 *(1.08)*	149	114 *(0.77)*	152 *(1.02)*
noway	111	99 *(0.89)*	132 *(1.19)*	127	104 *(0.82)*	139 *(1.09)*
nowsort	109	104 *(0.95)*	138 *(1.27)*	136	110 *(0.81)*	147 *(1.08)*
gs	119	107 *(0.90)*	142 *(1.20)*	141	109 *(0.78)*	146 *(1.04)*
ispell	145	113 *(0.78)*	151 *(1.04)*	149	115 *(0.77)*	153 *(1.03)*
compress	91	102 *(1.13)*	137 *(1.50)*	127	104 *(0.82)*	139 *(1.09)*
go	97	96 *(0.99)*	128 *(1.31)*	128	98 *(0.76)*	130 *(1.02)*
perl	136	106 *(0.78)*	141 *(1.04)*	140	107 *(0.76)*	142 *(1.01)*

Table 6: **Performance (in MIPS) of IRAM versus conventional processors, as a function of processor slowdown in a DRAM process.** Only the models with the 32:1 DRAM to SRAM-cache area density ratio are shown. The values in parentheses are the ratios of performances of the IRAM models compared to the CONVENTIONAL implementations. Ratios greater than 1.0 indicate that IRAM has higher performance.

swers to our questions about StrongARM. Thanks to Krste Asanovic for his role as an indispensable technical reference and to Noah Treuhaft, Randi Thomas, and Scott McPeak for useful feedback on earlier drafts of this paper.

Appendix

Energy dissipation modeling

The dominant factors of energy consumption in SRAM caches, DRAM caches, and external memory were captured in a spreadsheet. Typical values of circuit parameters, such as bit and word line capacitances and memory bank dimensions, were used [24][47][21][27][44][11][26][9] (see Table 4).

The dominant factor in DRAM energy dissipation is the capacitance of the bit lines being driven to the power supply rails. SRAM power dissipation is dominated by the sense amplifiers when reading, because the swing of the bit lines is low. However, to write the SRAM, the bit lines are driven to the rails, so their capacitance becomes the dominant factor when writing. For large arrays of SRAM and DRAM, driving the data into or out of the array and distributing the address to the row decoders also plays a significant role. Current-mode signaling is used for the data I/O, which is more energy efficient than voltage-mode [44]. Finally, there is some "background" power consumption, which is mostly cell leakage for SRAM and refresh power in the case of DRAM. This is normally very small, but can become non negligible when a memory is accessed rarely.

For all architectural models, the first-level instruction and data caches were closely modeled after the StrongARM caches, which are 32-way set-associative and are implemented as 16 banks. The tag arrays are implemented as Content-Addressable Memories (CAMs). This was done mainly to reduce power, since the conventional way of accessing a set-associative cache, reading all the lines in a set and then discarding all but one, is clearly wasteful. The second level unified cache is assumed to consist of the appropriate number of 512-by-256 DRAM banks, or 512-by-128 SRAM banks. This is organized in the conventional way, since it is direct mapped.

The IRAM model consists of 512 128Kbit sub-arrays, like some high-density DRAMs [27]. On-chip L2 caches, as well as the on-chip main memory, have 256-bit wide interfaces to the first level caches. In the case of IRAM, this is a significant departure from the common 4- to 16-bit wide memories and one of the main performance and energy advantages of IRAM-based architectures.

For external memory, for a fair comparison we used a single 64Mbit chip. This of course assumes that such chips with 32-bit wide interfaces will be available. This choice clearly minimizes the external memory power, both in the DRAM chips themselves (just one in our case) and in the bus, which is as small as possible. If such chips are not available, external power consumption will be higher and the IRAM advantage more pronounced.

Having calculated the energy dissipated in the various parts of the memory system each time they are accessed, the energy required for each memory operation is easily computed. For example, a primary cache read miss that hits in the secondary cache consists of (unsuccessfully) searching the L1 tag array, reading the L2 tag and data arrays, filling the line into the L1 data array, updating the L1 tag and returning the word to the processor. In addition, a writeback may be needed. Individual energy components are summed to yield the total energy for this operation. Such results are combined with the miss rates, dirty probabilities and read/write frequencies reported by shade to calculate the average energy per instruction.

References

[1] Accelerix Incorporated. http://www.accelerix.com/.

[2] AIMOTO, Y., ET AL. A 7.68 GIPS, 3.84 GB/s 1W parallel image-processing RAM integrating a 16 Mb DRAM and 128 processors. In *Digest of Technical Papers, 1996 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb 1996), vol. 39, pp. 372–373, 476.

[3] BURD, B., AND BRODERSEN, R. Energy efficient CMOS microprocessor design. In *Proroceedings of the Twenty-Eighth Hawaii International Conference on System Sciences* (Los Alamitos, CA, Jan. 1995), vol. 1, pp. 288–297.

[4] CHANDRAKASAN, A. *Low Power Digital CMOS Design*. PhD thesis, University of California at Berkeley, 1994.

[5] CHANDRAKASAN, A., ET AL. System design of a multimedia I/O terminal. In *Proceedings of IEEE Workshop on VLSI Signal Processing* (Veldhoven, Netherlands, October 1993), pp. 57–65.

[6] CULBERT, M. Low power hardware for a high performance PDA. In *Digest of Technical Papers, 1994 IEEE Symposium on Low Power Electronics* (San Diego, CA, Oct. 1994), pp. 28–31.

[7] CVETANOVIC, Z., AND BHANDARKAR, D. Performance characterization of the Alpha 21164 microprocessor using TP and SPEC workloads. In *Proceedings, Second International Symposium on High-Performance Computer Architecture* (San Jose, CA, Feb 1996), pp. 270–280.

[8] DIGITAL EQUIPMENT CORPORATION. *Alpha 21164 Microprocessor Hardware Reference Manual*, 1995.

[9] DIMARCO, D., ET AL. A 200MHz 256KB second-level cache with 1.6GB/s data bandwidth. In *Digest of Technical Papers, 1996 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb 1996), vol. 39, pp. 158–159.

[10] DOBBERPUHL, D. The design of a high performance low power microprocessor. In *Digest of Technical Papers, 1996 International Symposium on Low Power Electronics and Design* (Montery, CA, Aug. 1996), pp. 11–16.

[11] DOBBERPUHL, D. Personal communication, Nov. 1996.

[12] FOSS, R. Implementing application specific memory. In *Digest of Technical Papers, 1996 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb. 1996), pp. 260–261, 456.

[13] Galileo project. http://www.cs.wisc.edu/galileo/. Computer Sciences Department, University of Wisconin - Madison.

[14] GARRIS, M., ET AL. *Public Domain OCR: NIST Form-Based Handprint Recognition System*. National Institute of Standards and Technology, http://www.nist.gov/itl/div894/894.03/databases/defs/nist_ocr.html.

[15] GIACALONE, G., ET AL. A 1 MB, 100 MHz integrated L2 cache memory with 128b interface and ECC protection. In *Digest of Technical Papers, 1996 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb. 1996), pp. 370–371.

[16] GONZALEZ, R., AND HOROWITZ, M. Energy dissipation in general purpose microprocessors. *IEEE Journal of Solid-State Circuits 31*, 9 (Sept. 1996), 1277–1284.

[17] HARRIS, E., ET AL. Technology directions for portable computers. In *Proceedings of the IEEE* (Apr. 1995), vol. 83, pp. 636–658.

[18] HENNESSY, J., AND PATTERSON, D. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann Publishers, Inc, San Francisco, CA, 1996.

[19] HOROWITZ, M. Merged DRAM/Logic. IEDM short course on "DRAMs in the 21st Century", San Francisco, CA, December 1996.

[20] IKEDA, T. Thinkpad low-power evolution. In *Digest of Technical Papers, 1995 IEEE Symposium on Low Power Electronics* (San Jose, CA, Oct. 1995), pp. 6–7.

[21] ITOH, K., SASAKI, K., AND NAKAGOME, Y. Trends in low-power RAM circuit technologies. In *Proceedings of the IEEE* (Apr. 1995), vol. 83, pp. 524–543.

[22] IWATA, S., SHIMIZU, T., ET AL. Performance evaluation of a microprocessor with on-chip DRAM and high bandwidth internal bus. In *Custom Integrated Circuits Conference* (1996), pp. 269–272.

[23] KIMURA, O., ET AL. Panel: DRAM + logic integration: Which architecture and fabrication process. In *IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb. 1997).

[24] KOIKE, H., ET AL. A 30ns 64Mb DRAM with built-in self-test and repair function. In *Digest of Technical Papers, 1996 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb. 1996), pp. 150–151, 270.

[25] MONTANARO, J., ET AL. A 160MHz 32b 0.5W CMOS RISC microprocessor. In *Digest of Technical Papers, 1996 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb. 1996), pp. 214–215, 447.

[26] MURRAY, D. Personal communication, Nov. 1996.

[27] NAKAMURA, M. TAKAHASHI, T., ET AL. A 29ns 64Mb DRAM with hierarchical array architecture. In *Digest of Technical Papers, 1995 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb 1995), vol. 38, pp. 246–247.

[28] NSF workshop on critical issues in computer architecture research. http://www.cise.nsf.gov/mips/MSAWorkshop96/, May 1996.

[29] NeoMagic Corporation. http://www.neomagic.com/.

[30] PATTERSON, D. Panel: Research opportunities and critiques: An industrial perspective. In *1996 International Symposium on Computer Architecture* (Philadelphia, PA, May 1996).

[31] PATTERSON, D., ANDERSON, T., AND YELICK, K. A case for Intelligent DRAM: IRAM. In *HotChips VIII* (Stanford, CA, 1996), pp. 75–93.

[32] PATTERSON, D., ET AL. A case for Intelligent RAM: IRAM. In *IEEE Micro* (Apr. 1997).

[33] PATTERSON, D., ET AL. Intelligent RAM (IRAM): Chips that remember and compute. In *Digest of Technical Papers, 1997 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb. 1997), pp. 224–225.

[34] PPRAM project. http://kasuga.csce.kyushu-u.ac.jp/˜ppram/. Computer Science Division, Kyushu Univeristy, Japan.

[35] PRZYBYLSKI, S. A. *New DRAM Technologies: A Comprehensive Analysis of the New Architctures*. MicroDesign Resources, Sebastopol, California, 1994.

[36] RENALS, S., AND HOCHBERK, M. Decoder technology for connectionist large vocabulary speech recognition. Research Memorandum CS-95-17, Department of Computer Science, University of Sheffield, 1995.

[37] SANTHANAM, S. Personal communication, Oct. 1996.

[38] SANTHANAM, S. StrongARM SA110: A 160MHz 32b 0.5W CMOS ARM processor. In *HotChips VIII* (Stanford, CA, 1996), pp. 119–130.

[39] SATO, T., NAGAMATSU, M., AND TAGO, H. Power and performance simulator: ESP and its application for 100MIPS/W class RISC design. In *Digest of Technical Papers, 1994 IEEE Symposium on Low Power Electronics* (San Diego, CA, Oct. 1994), pp. 46–47.

[40] SAULSBURY, A., PONG, F., AND NOWATZYK, A. Missing the memory wall: The case for processor/memory integration. In *Proceedings of the 23rd Annual International Symposium on Computer Architecture* (Philadelphia, PA, May 1996), pp. 90–101.

[41] SHIMIZU, T., ET AL. A multimedia 32b RISC microprocessor with 16Mb DRAM. In *Digest of Technical Papers, 1996 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb. 1996), pp. 216–217, 448.

[42] SPEC CPU95 benchmarks. http://www.specbench.org/osg/cpu95/.

[43] SUN MICROSYSTEMS LABORATORIES, INC. *Introduction to Shade*. Mountain View, CA, Apr 1993.

[44] TAGUCHI, M. TOMITA, H., ET AL. A 40ns 64Mb DRAM with current-sensing data-bus amplifier. In *Digest of Technical Papers, 1991 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb 1991), vol. 34, pp. 112–113.

[45] WEISER, M., ET AL. Scheduling for reduced CPU energy. In *Proceedings of the First USENIX Symposium on Operating Systems Design and Implementation OSDI* (Monterey, CA, November 1994), pp. 13–23.

[46] WULF, W., AND MCKEE, S. Hitting the memory wall: Implications of the obvious. *Computer Architecture News 23*, 1 (Mar. 1995), 20–24.

[47] YAMADA, T., ET AL. A 64Mb DRAM with meshed power line and distributed sense-amplifier driver. In *Digest of Technical Papers, 1991 IEEE International Solid-State Circuits Conference* (San Francisco, CA, Feb 1991), vol. 34, pp. 108–109.