

Low Power Memory Hierarchies: An Argument for Second-Level Caches

J. Kelly Flanagan [†]
kelly@cs.byu.edu

James K Archibald [‡]
jka@ee.byu.edu

Jun Su [†]
junsu@cs.byu.edu

[†] Computer Science Department

[‡] Electrical and Computer Engineering Department
Brigham Young University
Provo, Utah 84602

March 4, 1997

Abstract

With the availability of high-performance, low-power microprocessors, portable computing is becoming commonplace. The prevalence of portable computers makes them the most obvious examples of systems in which power requirements are a significant design issue. This paper addresses the power tradeoffs of an important component of modern memory hierarchies: second-level caches. Thought by some to *increase* the total system power requirements, second-level caches can actually *reduce* the power consumed by the memory hierarchy – in addition to improving the overall performance. Power is saved by substituting second-level cache accesses for main memory accesses; given current memory technology, an active second-level cache and an idle main memory require less total power than an active memory by itself. Clearly the amount of power saved depends on the extent to which main memory accesses are reduced.

Keywords: Cache, Memory Hierarchy, Power Consumption, Low Power, Trace-Driven Simulation

1 Introduction

Computer performance has increased rapidly during the past few years as a result of improvements in processor architecture and implementation technology. Because of the increasing disparity between processor and main memory speeds, cache memories play an increasingly important role in reducing the memory bottleneck. The study in this paper assumes a memory organization in which the first-level (L1) caches are on the same silicon die as the CPU, and second-level (L2) cache is constructed of discrete SRAM components and a cache controller¹. Since the configuration of an on-chip L1 cache is fixed by the selection of a particular processor, the system designer is limited to specifying the organization of the L2 cache and main memory.

The performance tradeoffs of two level cache organizations have been the subject of many studies [2, 3, 4, 5, 6, 7], but little work has been done to evaluate the power tradeoffs in their design. Reducing power consumption is extremely important for portable computing devices [8, 9] and it is environmentally advantageous for desktop and server systems.

Table 1 illustrates the power consumed by a typical 12.1 inch TFT display, a 2.5 inch hard disk drive, a 133 MHz Pentium processor, a memory hierarchy consisting of a 256-KByte L2 cache and 40-MBytes of DRAM memory, and a system chipset and integrated modem. The power consumption for the TFT display was obtained by measuring the power consumed by a Winbook FX laptop computer running the Ziff-Davis Winstone benchmarks with and without the TFT display operating and then subtracting the two numbers. In contrast, power consumption figures for the hard disk, CPU, L2 cache, and main memory were obtained from product data sheets. The last step was to determine the power consumed by the chipset and modem components. This was

¹While recent technological improvements have permitted the introduction of on-chip L2 caches [1], larger, off-chip caches almost certainly will continue to be used. The power tradeoffs of such hierarchies are not explored here, but we believe they will be similar.

Component	Power Consumption	% of Total
12.1 inch TFT Display	5.9	32.3%
2.5 inch Hard Disk	3.9	21.3%
Memory Hierarchy	3.3	18.0%
133 MHz Pentium	3.3	18.0%
Chipset and Modem	1.9	10.4%
Total	18.3	100%

Table 1: Power consumption of various components of a Winbook FX laptop computer system. All units are in Watts.

calculated simply by subtracting the power consumption of all other components from the measured power consumption of the overall system.

It is clear from the table that the two most power consuming devices are the display and the hard disk drive. Previous work has focused on reducing the power consumption of hard disk drives [9] and, as a result, modern drives consume much less power than their predecessors. It is also clear from Table 1 that the memory hierarchy consumes a significant amount of power equal to that of the processor itself. This work investigates memory hierarchy configurations and technologies that decrease the power consumption of these components while increasing performance.

Until recently, portable computers have not contained L2 caches because it was assumed that they increase system power requirements [3]. One might suppose that their use in portable systems comes at the expense of shortened battery life, but we will demonstrate that this is not the case. The major contribution of this paper is in showing that an L2 cache can both decrease power consumption and increase system performance relative to the same system with no L2 cache.

To illustrate how this is possible, suppose we have two memory hierarchies: S_h , which includes both main memory and an L2 cache, and S_m , which is a memory only system with a main memory identical to that of S_h . We wish to compare the power consumption of S_h and S_m as they service the same reference stream. At any instant in time, each component in the hierarchy is either active

or inactive, requiring respectively the *active* or *standby power* specified by the manufacturer.

During intervals between requests when all components in the hierarchy are in their standby state, S_h consumes more power than S_m . S_h also consumes more power in all cases when main memory must be accessed in both hierarchies. The only case when S_h may consume less power is when it satisfies a request with an access to L2 and S_m must access memory. In this case power is saved only if the power consumed by an active L2 cache and an idle memory is less than the power consumed by an active memory alone; given current technology this condition holds, as will be shown in Section 4.2. The net power gain or loss depends on how often each of these situations occur. Clearly an accurate evaluation of power consumption requires an accurate characterization of the utilization of each component in the hierarchy.

In this paper, we present an analytical model used to determine the power consumption of various memory hierarchies. This model requires as input parameters both device power consumption figures and the fraction of time each component is active. These parameters are obtained from memory device data books and trace-driven simulation. We use this model to determine and compare the power consumption of various memory hierarchy configurations with and without L2 caches using several different workloads. We chose simulation using an analytical model over direct system measurement because the latter would severely constrain the range of cache sizes and memory and cache technologies we could examine.

As previously described, trace-driven simulation [6, 7, 10] was used to accomplish this work. Two items are needed to obtain accurate simulation results: an accurate simulation model, and input trace data which represents the memory references produced by the CPU. For this work, long and accurate traces were collected, using a hardware monitoring technique, from an Intel Pentium processor containing an L1 cache. Since we are investigating the external memory hierarchy, the

L1 cache remained enabled during the trace collection process. Each trace is at least 500 million references (either L1 misses or non-cacheable references) in length and contains the address and control information for each reference generated by the processor during execution.

The remainder of this paper is organized into four sections. Section 2 describes our analytical model. Section 3 describes our experimental setup, including the trace collection technique, chosen workloads, the system under test, our cache simulator, and collected trace data. Section 4 presents the experimental results and Section 5 contains conclusions and future work.

2 Analytical Model

The model presented in this section quantifies the power consumption of a memory hierarchy given details about the workload, the chip technology, and memory and cache sizes. The model's output is limited to the power consumed by L2 and main memory – the only components of the memory hierarchy that a system designer can change after a specific processor chip is chosen. Since the model is restricted to L2 and main memory activity, relevant workload details include misses and write-backs from the L1 caches, as well as non-cacheable references that bypass the caches. To limit the space we must explore, this study is limited to direct-mapped L2 caches, but the approach is applicable to other cache organizations. As previously mentioned, we chose the combination of simulation and an analytical model over direct system measurement because of its greater flexibility in evaluating a larger range of design and technology parameters.

We first derive an analytical model that estimates the power consumed by systems during active and user-think-time periods. We then extend this model to estimate the power consumed by memory-only systems and those containing memory and a second level cache. Finally, we compare the power consumption of these two systems.

2.1 Active and Idle Systems

The first issue we will consider is that of user think time – important because it represents approximately 80% of the time a portable computer is on [11]. A simple, yet accurate modeling approach is to consider this “idle” time (from the users point of view) as a separate workload with its own operational characteristics. This leads to the following formulation, where P_h represents the average power consumed by a memory hierarchy:

$$P_h = f_{ui} * P_{ui} + (1 - f_{ui}) * P_{ua}, \quad (1)$$

where f_{ui} is the fraction of time the user sees the system as idle, and P_{ui} and P_{ua} represent the power consumed while the system is idle and active respectively.

One might assume that L2 and main memory are essentially idle during user think time, allowing their power consumption to be characterized as the standby power specified for those components. Surprisingly, our trace data (discussed in Section 4.4) shows high levels of memory traffic during user think time, resulting in power consumption by a memory-only system that is *higher than the power consumed during the execution of the applications we traced*. In light of this discovery, we elected to model periods of user think time as periods of active execution – with data from application execution. This corresponds to setting the term f_{ui} to zero in Equation 1. Although this approach actually *underestimates* power consumption by a slight amount, it has the advantage of simplicity and avoids placing too much emphasis on the behavior of the idle loop – code not typically optimized for performance or resource usage.

In the remainder of this section we will develop our analytical model ignoring user think time. The model does contain references to *idle* time, but this is the time between references produced by the CPU during which the external memory hierarchy is idle and not idle time as seen by the user.

2.2 No L2 Cache

For a system without an L2 cache, only the main memory must be considered. Let P_m represent the total power consumed by the memory hierarchy. Then

$$P_m = f_{ma} * P_{ma} + (1 - f_{ma}) * P_{mi} \quad (2)$$

where f_{ma} is the fraction of time the memory is active, and P_{ma} and P_{mi} are the power consumption of main memory when it is active and idle respectively.

2.3 L2 Cache

We can express the power consumed by a second-level cache, P_c , as follows:

$$P_c = f_{ca} * P_{ca} + (1 - f_{ca}) * P_{ci}, \quad (3)$$

where f_{ca} is the fraction of time the cache is active, and P_{ca} and P_{ci} are the power consumption of the cache when it is active and idle respectively.

Equations 2 and 3 can be combined to yield the power consumed by a memory hierarchy containing both L2 and main memory.

$$P_h = f_{ma} * P_{ma} + (1 - f_{ma}) * P_{mi} + f_{ca} * P_{ca} + (1 - f_{ca}) * P_{ci}. \quad (4)$$

2.4 The Complete Models

While servicing an identical sequence of requests, a hierarchy with a second level cache will have a smaller fraction of memory busy time than a memory-only system. In order to allow the comparison of the power requirements of both types of hierarchies without confusion, we introduce two new terms that differentiate between the two types of parameters. For a memory-only system we will use the term f_{ma}^m and for a hierarchy containing cache we use f_{ma}^h .

The final form of our equations representing power consumption are given below.

$$P_m = f_{ma}^m * P_{ma} + (1 - f_{ma}^m) * P_{mi} \quad (5)$$

$$P_h = f_{ma}^h * P_{ma} + (1 - f_{ma}^h) * P_{mi} + f_{ca} * P_{ca} + (1 - f_{ca}) * P_{ci} \quad (6)$$

The values of P_{ma} , P_{mi} , P_{ca} , and P_{ci} are dependent on system configuration and technology issues and can be obtained from data books or through circuit simulation. The remaining three terms f_{ma}^m , f_{ma}^h , and f_{ca} are dependent on both system configuration and workload characteristics. For this work we use trace-driven simulation to obtain these values.

3 Simulation Methodology

This section describes the collection of trace data that will be used to determine the fraction of time each component of a memory hierarchy is active. We collect accurate memory reference traces of realistic workloads for portable computers, then we use trace-driven simulation to determine the number of accesses to each level of the simulated memory hierarchy. The following subsections describe our trace collection technique, the workloads and system we traced, our cache simulation model, and the trace data that was collected.

3.1 Trace Collection Technique

Trace-driven simulation is very dependent on accurate trace data [12, 13]. Our technique [14, 15, 16] uses an off-the-shelf logic analyzer to capture complete and very long traces.

Figure 1 shows a typical setup that consists of the machine being traced (SUT), Logic Analyzer (TLA510), and a workstation for processing the acquired data. When enabled, the TLA510 monitors the pins of the SUT, storing desired signal values in an internal buffer. The collected sig-

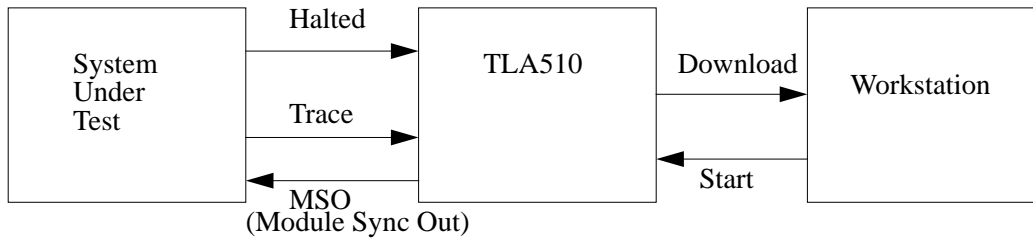


Figure 1: Communications between SUT, TLA510 and a workstation

nals include the processor’s address, data, and control pins. When the buffer fills, the workstation downloads the trace for storage or processing.

A technique that overcomes the trace length limitation of previous hardware monitors has been developed. When the TLA510’s internal buffer is full it sends an interrupt to the SUT. The low priority interrupt enables the SUT to finish other higher priority interrupts before responding. After all other devices have been serviced, the SUT enters an interrupt routine and spins in a tight loop. While the SUT spins in an interrupt loop with interrupts disabled, the internal buffer is emptied by the workstation via the network. The buffer contents may be stored to secondary storage media or processed while being extracted. When the buffer is emptied, the TLA510 signals the SUT to continue execution. This process may be repeated as many times as desired, producing a contiguous trace of arbitrary length.

3.2 Workloads

The traced workloads include the Ziff-Davis Winstone 1995 benchmark suite and Descent, a popular 3D graphics game. The Winstone 1995 benchmark suite contains four major application categories: business graphics, wordprocessors, spreadsheets, and databases. Each category contains several prominent commercial applications. We believe these five workloads and the chosen operating

system, Microsoft Windows95, are representative of current and future workloads on a large number of portable computer systems.

3.3 System Under Test

The SUT is a Pentium-based system running at 66 MHz. The system contains write-back L1 caches, a 256-KByte L2 cache, 32-MBytes of main memory and a 540-MByte IDE hard disk drive. We believe this system is representative of many portable computer systems. Although future processors will certainly have higher clock rates than the machine being used here, the 66 MHz external bus is likely to remain the same. This will be the case with the use of 133 and 200 MHz clock-doubled and clock-tripled CPUs. As the internal CPU clock rates are increased, the amount of idle time between external references decreases, further increasing the power saving ability of L2 caches.

3.4 Cache Simulator

Our cache simulator consumes generated trace data on-the-fly and uses this data to simulate many cache configurations with various sizes, line sizes, associativities, and write policies. For this work we model direct-mapped caches ranging in size from 32-KBytes to 1024-KBytes. Each cache has 32-byte lines and performs allocate on writes and uses the write-back write policy. The simulator also evaluates cacheability information obtained from the system under test to model noncacheable references properly. The simulator returns the number of cycles the L2 cache and memory are active and the total number of cycles required for each configuration to satisfy all requests. With these values the fraction of time that the cache and memory are active can be computed.

The number of cycles that a cache and memory are active and the power consumed for a given request is technology dependent. For this work we consider several different types of SRAM

Benchmark	Total Refs *10 ⁶	Total Components *10 ⁶			Noncacheable *10 ⁶		
		IFetches	Reads	Writes	IFetches	Reads	Writes
Database	970	399	272	299	0	131	266
Graphics	540	168	135	237	0	62	220
Spreadsheet	675	250	200	225	0	90	199
Wordprocessor	590	218	180	192	0	86	170
Descent	625	94	124	407	0	26	386
Total	3,400	1,129	911	1,360	0	395	1,241

Table 2: Statistics of traces collected from our Intel Pentium based machine running Microsoft Windows95, the Ziff-Davis Winstone 95 Benchmark Suite, and a popular 3D graphics game. The figures illustrated in the *total components* section of the table represent the number of cacheable and noncacheable references of each type.

and DRAM technologies which are used extensively in current computer systems. The chosen technologies include synchronous pipelined SRAM, synchronous non-pipelined SRAM, fast page mode DRAM, and EDO page mode DRAM. Each of these yield different bus transfer rates and power consumption figures.

3.5 Collected Trace Data

For this work, we traced the four workload categories from the Winstone benchmark suite from beginning to end. The Descent trace represents the execution of a short, but complete session replayed from a recorded game. The resulting traces range between 540 million and 970 million references in length, as shown in Table 2. Each record in the trace represents an L1 cache miss, an L1 write-back request, or a noncacheable reference; the TLA510 logic analyzer collected traces between the internal L1 caches and the external L2 cache. Since the traces do not include any CPU references serviced by the L1 caches – the majority of total references – each represents several minutes of execution. Recording references after the L1 caches is ideal for our study since we are interested only in those that are serviced by the off-chip memory hierarchy; the technique also has the advantage of perturbing the system less by having the L1 cache enabled.

4 Experimental Results

In this section we discuss the technique used to obtain the memory activity parameters from the collected trace data. We also present timing and power details about the SRAM and DRAM memory technologies considered in this study. We then present and discuss power consumption results from our model, comparing and contrasting different memory technologies and hierarchy configurations. Our results show that the addition of an L2 cache can increase the time when main memory is idle, leading to a reduction in the total power required by the memory hierarchy. Results are also presented showing that power consumption during periods of user think time is at least as high as during the execution of the benchmarks we traced. Finally, to allow an evaluation of power and performance tradeoffs, we quantify the improvement in system performance using a second-level cache.

4.1 Determining Cache and Memory Active Time

The fraction of time a device is active is simply the time the device is active divided by the total time of execution:

$$f_a = \frac{T_a}{T_t}, \quad (7)$$

where f_a is the fraction of active time for a given device, T_a is the time the device is active, and T_t is the total execution time. Using trace-driven simulation it is easy to determine the time each component is active (T_a). However, the total execution time is more difficult to obtain. If we simply measure run time on a particular system our results will be affected by implementation specific details such as the size of the L2 cache or the speed of memory in the measured system. Since the total execution time should depend only on the details of the modeled hierarchy, an alternative approach must be taken.

The technique we employed is to separate the execution time on the measured system into two parts: one dependent on the specific configuration and technology used in the hierarchy, and the other independent of the specific L2 cache and memory. The dependent part is simply the time spent by the hierarchy servicing memory requests; the latter is the remaining time during which the L2 and main memory are idle – either no requests were generated or they were serviced by the internal L1 caches. If T_t represents the total execution time, then

$$T_t \approx T_s + T_i, \tag{8}$$

where T_s is the time spent by the specific system in question servicing requests and T_i is the time the hierarchy was idle. Since T_i is constant for a given CPU, L1 cache, and workload, we can obtain it from analysis of our trace data and then add the appropriate value of T_s obtained from simulation of a particular memory hierarchy. Summing the values produces a reasonably accurate estimate of the total execution time for each specific configuration.²

4.2 Technology and Timing Issues

In this study we consider several SRAM and DRAM technologies used extensively in memory hierarchies. Caches and main memories based on these technologies have various timing requirements for reading and writing data. For instance, a cache based on synchronous pipelined SRAM has a read transfer rate of 2-1-1-1, which means the first transfer between L2 and L1 requires two CPU cycles while the remaining data requires a single cycle per word. The transfer rates for each configuration are shown in Table 3.

In the future, custom L2 caches similar to those found in the Intel Pentium Pro and the DEC Alpha will be more common. In this case the CPU architect and designer can optimize the cache

²The estimate is precise only if the CPU is always stalled waiting for L2 and memory requests to complete.

	Pipelined SRAM	Non-pipelined SRAM	EDO Page DRAM	Fast Page DRAM
Read	2-1-1-1	3-1-1-1	7-2-2-2	7-3-3-3
Write	3-1-1-1	4-1-1-1	4-2-2-2	4-3-3-3

Table 3: Bus transfer rates for cache and memory systems based on several SRAM and DRAM technologies. The SRAM based cache systems require one additional cycle for a write requests to determine if the request generates a hit or a miss. In DRAM based main memory systems a read request results in an internal read followed by a write requiring additional initial cycles.

	32K	64K	128K	256K	512K	1024K
Pipelined-active	1.557	1.946	2.433	3.042	3.802	4.752
Pipelined-standby	0.260	0.324	0.406	0.507	0.634	0.792
Non-pipelined-active	2.163	2.703	3.379	4.224	5.280	6.600
Non-pipelined-standby	0.433	0.541	0.676	0.845	1.056	1.320

Table 4: Power consumption of different 3.3-volt synchronous SRAM caches. All units are in Watts.

for power consumption and performance. We assume that a doubling in cache size results in a 25% increase in power consumption for a given technology [17].

For this work we consider 3.3 Volt SRAM technologies for L2 caches and 3.3 Volt DRAM technologies for main memory. The power consumed by each component was obtained through manufacturer data books [18, 19]. We first calculated the power consumption of a 1024-KByte cache and divided this value by 1.25 as we decreased cache size by a factor of two. The power consumed by each cache system is presented in Table 4 while the power consumed by 32-MByte main memories is presented in Table 5.

4.3 Power Consumption

Figures 2 and 3 illustrate the power consumed by various memory hierarchies for the spreadsheet portion of the Winstone 95 benchmark suite. Notice in Figure 2 that for all but the largest cache a memory hierarchy containing both pipelined and nonpipelined L2 caches consumes less power than

	Active	Standby
EDO DRAM	6.534	0.106
FPM DRAM	5.346	0.106

Table 5: Power consumption of two types of 3.3-volt 32-MByte main memories. All units are in Watts. These values are significant since the active power consumption of these devices exceed the active power consumption of our 2.5 inch hard disk drive unit.

the memory only system. While this is true for the EDO DRAM based main memory, it can be seen in Figure 3 that a nonpipelined L2 cache does not save power when used with fast page mode DRAM. This is due to the fact that the FPM DRAM consumes less power than the EDO DRAM while the hierarchy containing a nonpipelined L2 cache consumes approximately the same power in either case. The very lowest power consumption is obtained by using a FPM main memory and a 256-KByte pipelined cache. While slightly more power is consumed by using the same cache configuration with EDO DRAM, improved performance results.

To conserve space we will not present the figures for the database, graphics, and word processing portions of the benchmark, but these are almost identical to those of the spreadsheet portion. The trends are identical and the lowest power consuming configuration is the same in each. The only significant difference is the amount of power consumed by the memory only systems in each case. The FPM memories vary in power consumption from a low of 2.8 Watts for the graphics workload to a high of 3.25 Watts for the database benchmark. The EDO main memory power consumption ranges from 3.25 Watts to 3.7 Watts for the same workloads.

Figures 4 and 5 illustrate the power consumed by various memory hierarchies for the Descent benchmark. These figures have similar trends to those presented above, but several differences appear. First, the total power consumed by the memory hierarchy is quite low. This is due to the efficiency of the internal L1 cache; few references reach the external memory hierarchy. As

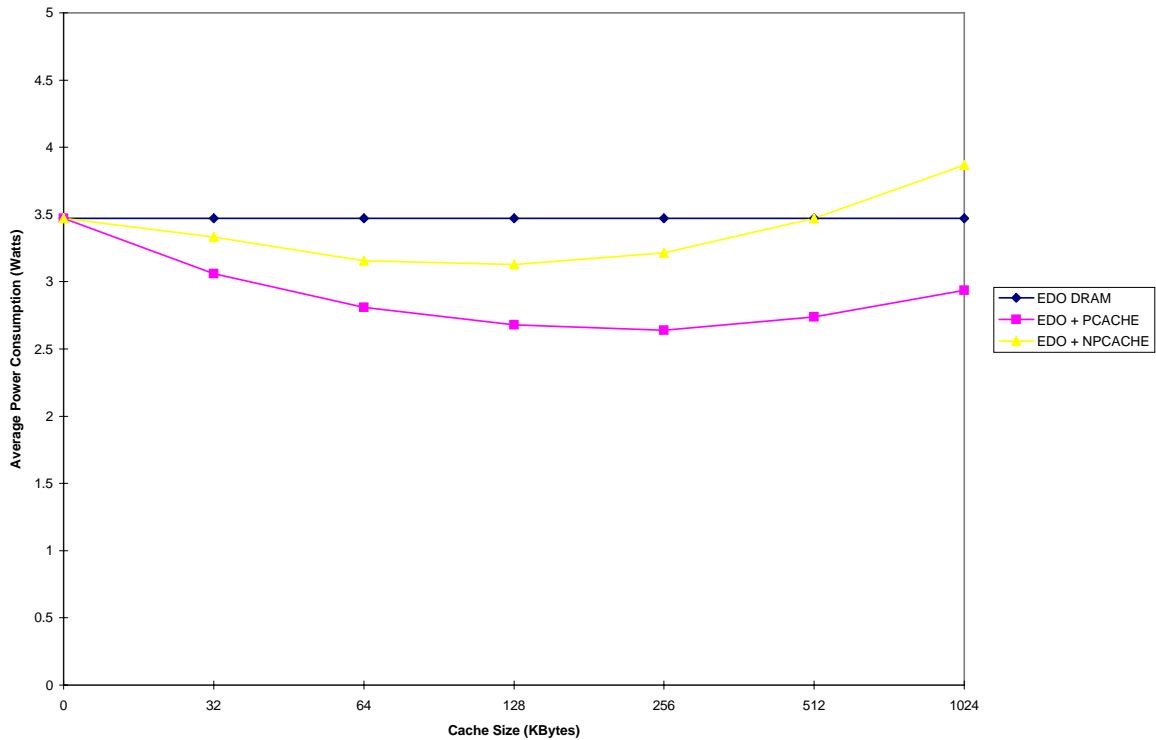


Figure 2: Power consumption of various memory hierarchies using EDO DRAM technology while running the spreadsheet portion of the Winstone 95 benchmark suite. The flat line is the power consumed by the EDO DRAM in a memory only system.

evidence to support this claim we looked at the number of cycles between references serviced by the external memory hierarchy. The four Winstone 95 benchmarks presented above average 10 CPU cycles between external memory requests while descent averages 30 cycles between requests. This efficient use of the L1 cache decreases the total power consumed by the memory hierarchy by decreasing the fraction of time the hierarchy is active. Second, the configuration requiring the least power is again the 256-KByte pipelined cache in conjunction with the FPM main memory shown in Figure 5. Unlike the Winstone 95 benchmarks, all memory hierarchies containing EDO DRAM consume more power than the FPM memory only systems. Finally, the hierarchies containing nonpipelined caches never decrease power consumption below that obtainable with a memory only system.

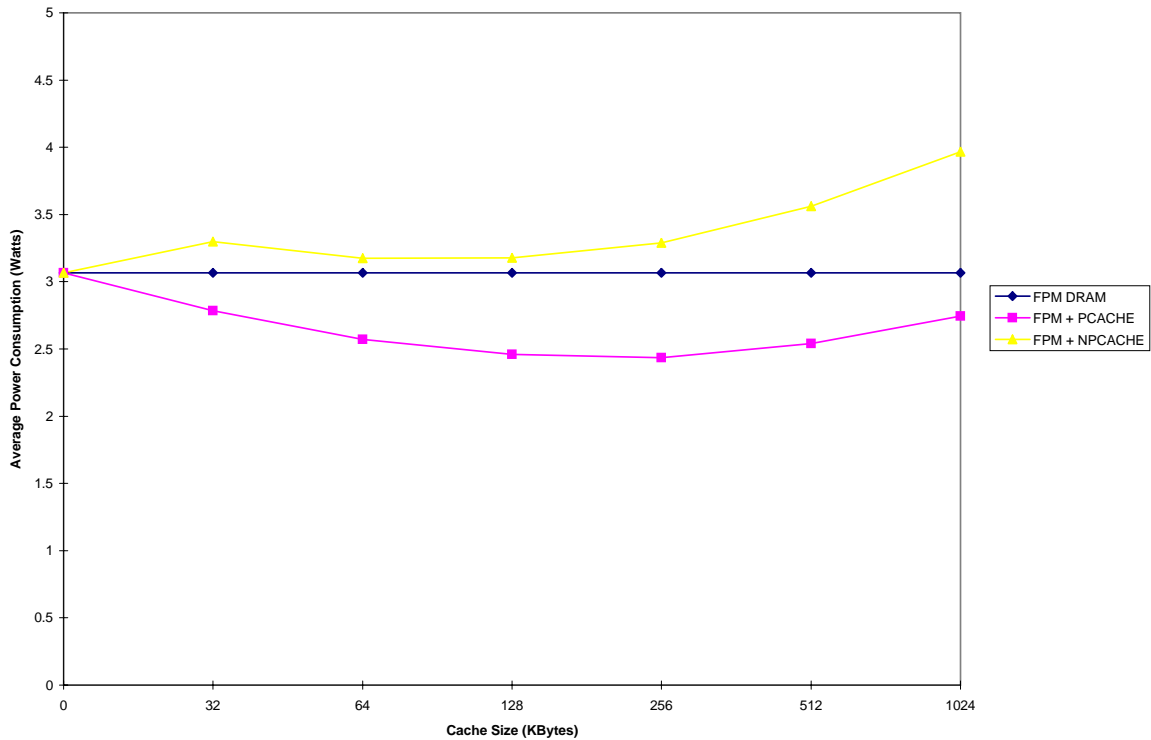


Figure 3: Power consumption of various memory hierarchies using FPM DRAM technology while running the spreadsheet portion of the Winstone 95 benchmark suite. The flat line is the power consumed by the FPM DRAM in a memory only system.

Consistent with Table 4, the results demonstrate that faster pipelined caches consume less power than nonpipelined caches. This is due to new fabrication technologies and not architectural differences. Note also that the EDO DRAMs consume more power than the slower FPM devices.

The unfortunate conclusion that can be drawn from the previous figures is that the system designer must choose between minimizing power consumption and maximizing performance. For example, the systems with the largest L2 caches will have the best performance, but they do not have the lowest power consumption. The tradeoffs can be a bit tricky: in systems containing 256-KByte L2 caches, pipelined L2 cache components have approximately a 20% performance advantage over non-pipelined components and consume less power. On the other hand, EDO DRAM results in a small performance improvement, but consumes approximately 12% more power. In systems

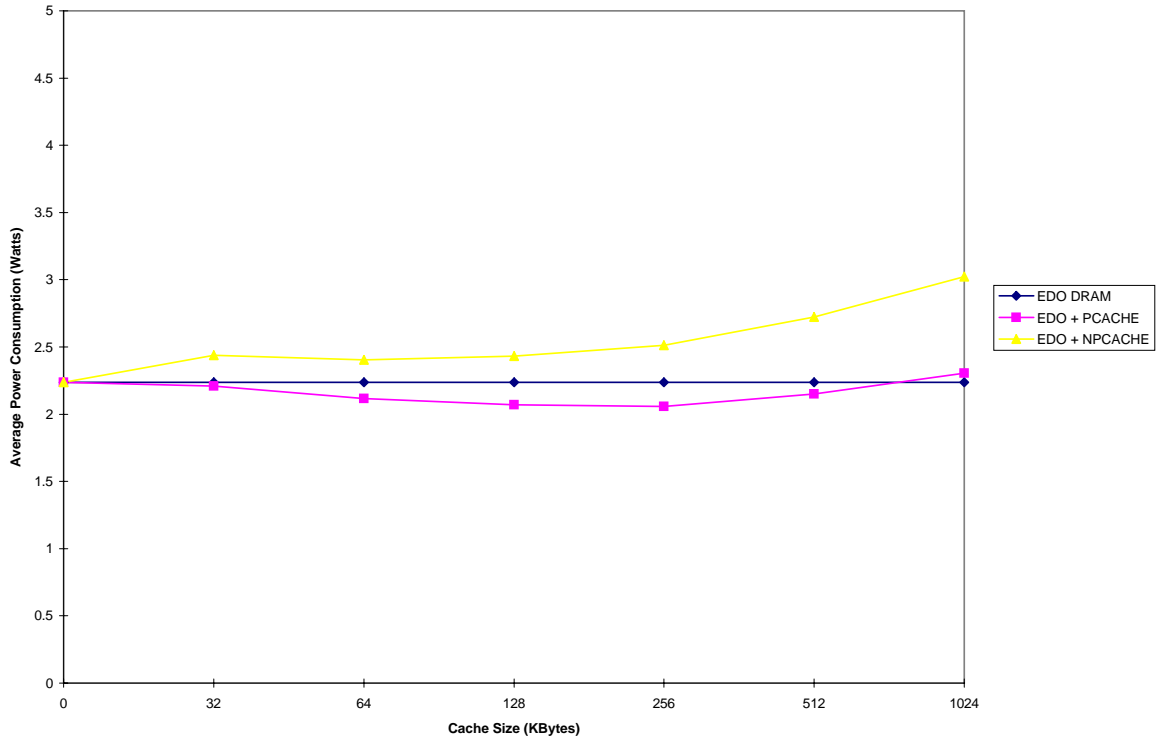


Figure 4: Power consumption of various memory hierarchies using EDO DRAM technology while running the Descent benchmark. The flat line is the power consumed by the EDO DRAM in a memory only system.

where power consumption is the main concern, the combination of a 256-KByte pipelined L2 cache and FPM main memory appears to be the best choice. If performance is the main concern, but low power is advantageous, the same L2 cache could be used in conjunction with EDO DRAM main memory.

4.4 Power Consumption During User Think Time

In Section 2.1 we stated that memory traffic and power consumption do not drop during periods of user think time. Data supporting this claim is presented in Figures 6 and 7. These two figures represent the power consumed by memory hierarchies while the Windows95 background screen and six icons are visible. To confirm the surprising results, we also traced two screen savers – the first simply blanks the screen and the second is the flying windows icon screen saver. Results for all idle

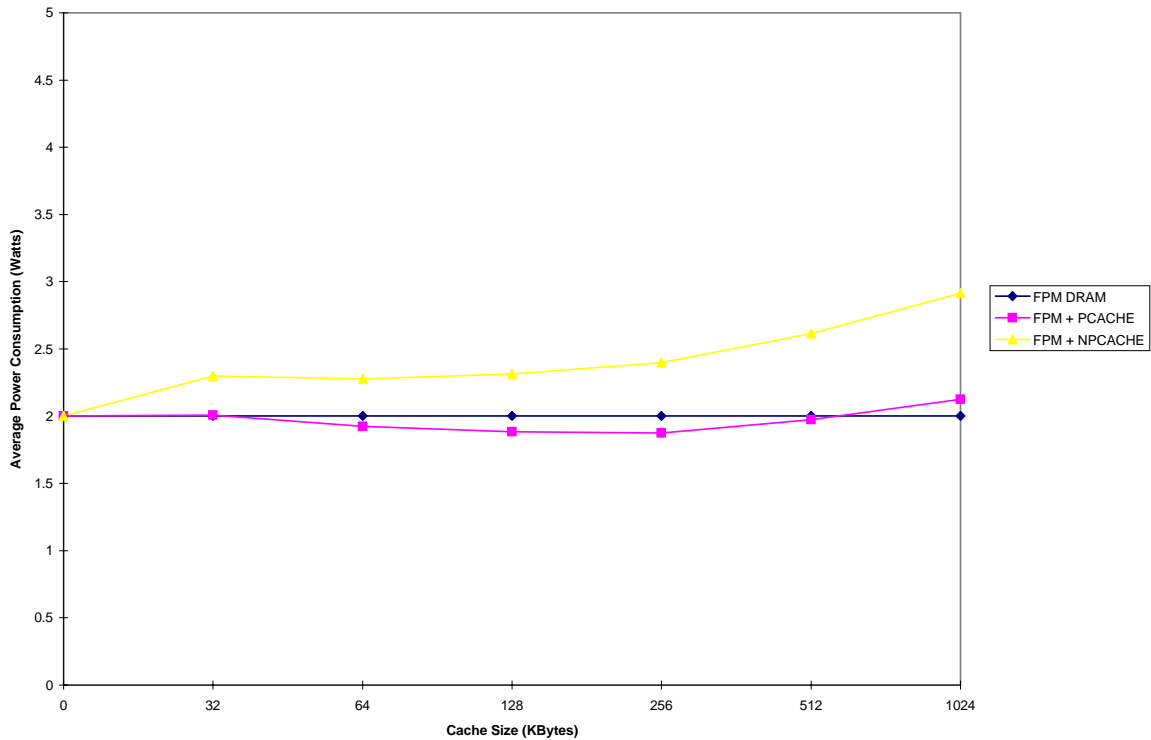


Figure 5: Power consumption of various memory hierarchies using FPM DRAM technology while running the Descent benchmark. The flat line is the power consumed by the FPM DRAM in a memory only system.

workloads traced are virtually identical.

Note that the power consumed by memory-only systems during user think time is significantly higher than the power consumed by the memory-only systems during the execution of any of the traced benchmarks. Because of the direct correlation between power consumption and component utilization, this clearly demonstrates an increase in memory traffic during user think time. Particularly noteworthy is that this workload causes twice the power consumption as the Descent benchmark. To summarize, the surprising conclusion that we draw from this data is the following: *not only is the external memory hierarchy not idle during user think time, but it is actually more active than during the execution of the other workloads!*

The data also clearly demonstrates that caches can substantially reduce power consumption over

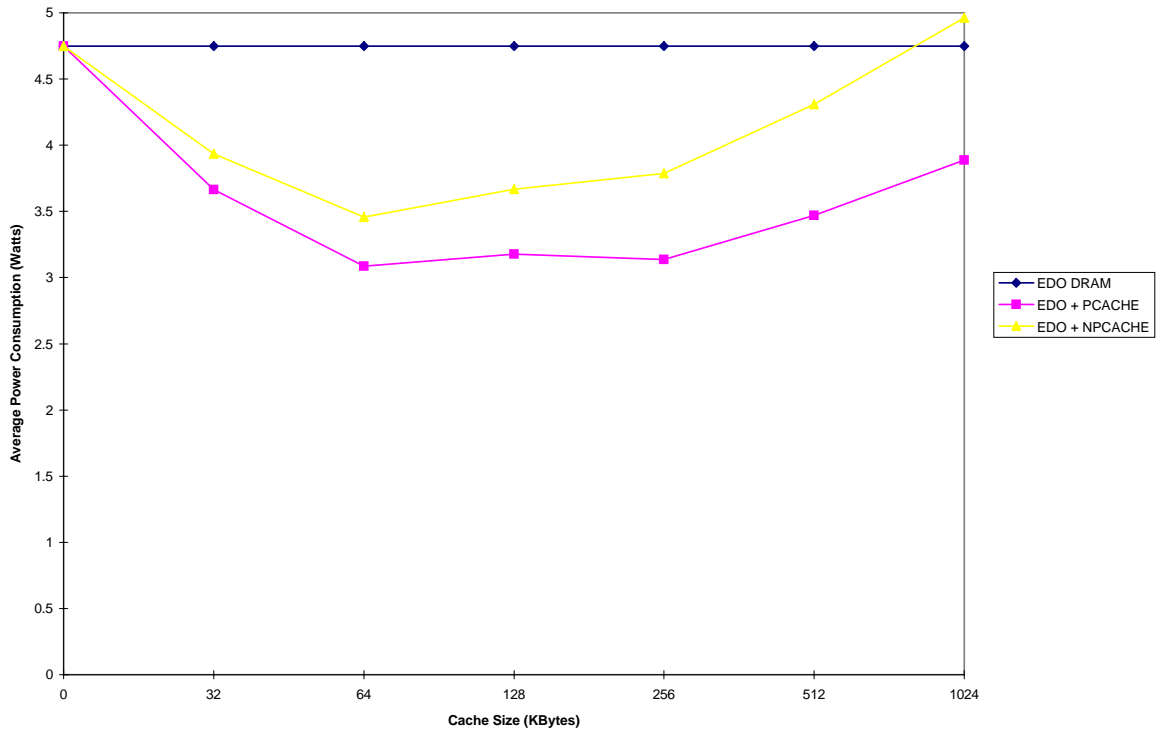


Figure 6: Power consumption of various memory hierarchies using EDO DRAM technology while the system is displaying the Windows95 background and six icons (user think time). The flat line is the power consumed by the FPM DRAM in a memory only system.

a memory-only system. In fact, they can reduce it by a greater percentage than when running any of the other benchmarks. We conclude, therefore, that if user think time with these characteristics were included in our analytical model, the benefit of using an L2 cache would be even greater.

4.5 Performance Enhancement

To quantify the improvement in system performance using an L2 cache, we present Figure 8 that illustrates the execution time in seconds of the spreadsheet benchmark with and without L2 caches.

It is clear from the figure that using EDO DRAM decreases execution time by approximately 10% compared to FPM DRAM in systems containing no L2 cache. Both EDO and FPM DRAM based systems benefit from the addition of an L2 cache. With the addition of a 256-KByte L2

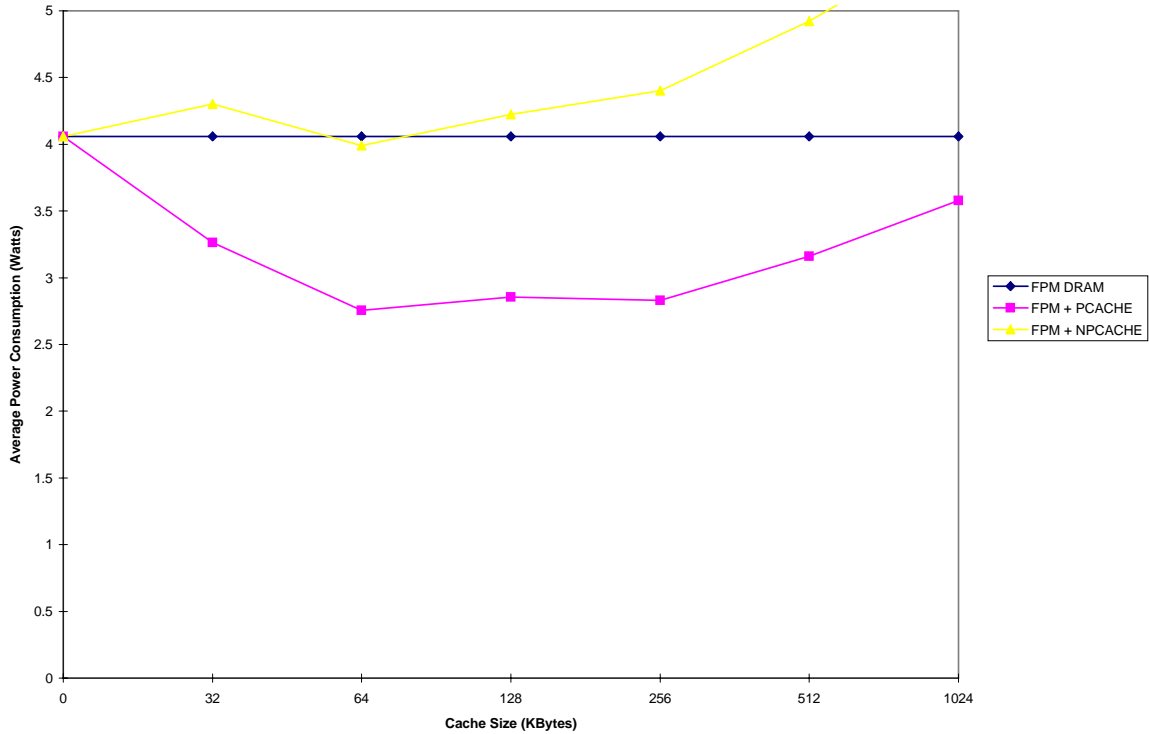


Figure 7: Power consumption of various memory hierarchies using FPM DRAM technology while the system is displaying the Windows95 background and six icons (user think time). The flat line is the power consumed by the FPM DRAM in a memory only system.

cache a system containing FPM DRAM achieves a decrease in execution time of nearly 25% while a similar system containing EDO DRAM decreases execution time by approximately 16%.

5 Conclusions

In this paper we used trace-driven simulation to investigate the power consumption of memory hierarchies in portable computer systems. Long and accurate traces of prominent commercial programs contained in the Ziff-Davis Winstone 1995 benchmark suite and a popular game were collected using a hardware monitoring device. Each trace is hundreds of millions of references long and contains the address and control information for each reference.

Until recently, portable computers have not contained L2 caches because it was assumed that

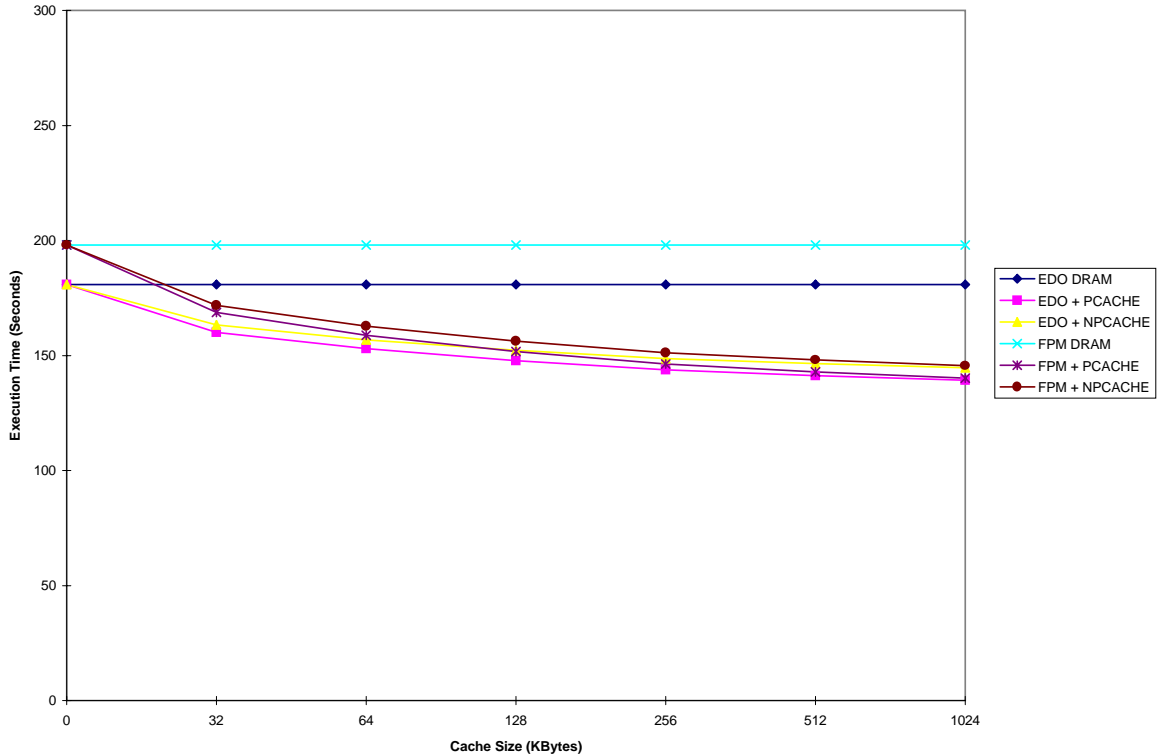


Figure 8: Performance of the spreadsheet benchmark with various L2 caches. The top two lines indicate that EDO DRAM is approximately ten percent faster than FPM DRAM for this workload. In addition, a 16 to 25 percent increase in performance can be achieved by using a 256-KByte L2 cache.

their addition would increase system power requirements. We have found that L2 caches reduce the memory hierarchy power requirements while increasing system performance. Our measurements and simulation study have revealed several interesting results:

1. Memory hierarchies in portable computer systems consume as much power as typical processors and should receive more attention in ongoing efforts to reduce system power consumption.
2. Second-level direct-mapped caches can reduce system power requirements in both desk-top and portable computer systems while increasing performance.
3. Of all memory hierarchies evaluated, the lowest power consumption was obtained by using a FPM main memory and a 256-KByte pipelined L2 cache. This configuration also yields

excellent performance.

4. Pipelined L2 caches consume less power and offer higher performance than nonpipelined implementations, greatly simplifying design decisions. In contrast, the choice of main memory is more complicated. EDO DRAM consumes more power than FPM DRAM, but yields higher performance.
5. Surprisingly, more power was consumed by the memory hierarchies during periods of user think time than during execution of the chosen workloads. The power savings of L2 caches are even greater when user think time is considered.

This work is an important step towards decreasing the power consumption of portable and desktop systems. Decreasing the power consumption of these systems will increase productivity and decrease the impact on the environment.

References

- [1] Microprocessor, The Pentium Pro processor at 150 MHz, Technical report, Intel Corporation, 1995.
- [2] A. J. Smith, Cache memories, *Computing Surveys*, 14(3):473–530, September 1982.
- [3] Jim Handy, *The cache memory book*, Academic Press, 1993.
- [4] S. Przybylski, M. Horowitz, and J. Hennessy, Characteristics of performance-optimal multi-level cache hierarchies, In *Proc. of 16th Annual Int. Symp. on Computer Architecture*. IEEE, 1989.
- [5] Steven A. Przybylski, *Cache and Memory Hierarchy Design, A performance-directed approach*, Morgan Kaufmann Publishers, Inc, 1990.
- [6] R. T. Short and H. M. Levy, A Simulation Study of Two-Level Caches, In *Proc. of 15th Int. Symp. on Computer Architecture*, pages 81–88. IEEE, 1988.
- [7] Hakon Bugge, E. Kristiansen, and B. Bakka, Trace driven simulation for a two level cache design in open bus systems, In *Proc. of 17th Int. Symp. on Computer Architecture*, pages 250–259. IEEE, 1990.
- [8] Kester Li, Toward a low power file system, Master’s thesis, University of California at Berkeley, May 1994.

- [9] Kester Li, Roger Kumpf, Paul Horton, and Thomas Anderson, A quantitative analysis of disk drive power management in portable computers, In *The proc. of the winter 1994 USENIX conference*, 1994.
- [10] A. Borg, R. E. Kessler, and D. W. Wall, Generation and analysis of very long address traces, In *Proc. of 17th Int. Symp. on Computer Architecture*, pages 270–279. ACM, 1990.
- [11] J. Thomas Pawlowski, Secondary cache increases performance and reduces power use in portable PCs, *EDN, Electrical Design News*, **23**:135–140, November 1994.
- [12] J. Kelly Flanagan, Brent E. Nelson, James K Archibald, and Greg Thompson, The inaccuracy of trace-driven simulation using incomplete multiprogramming trace data, In *Proc. of the IEEE International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems MASCOTS*. IEEE, 1996.
- [13] K. Grimsrud, J. Archibald, R. Frost, B. Nelson, and K. Flanagan, Estimation of simulation error due to trace inaccuracies, In *Proc. of 26th Asilomar Conference on Signals, Systems and Computers*, October 1992.
- [14] J. Kelly Flanagan, Brent E. Nelson, James K Archibald, and Knut Grimsrud, BACH: BYU Address Collection Hardware, the collection of complete traces, In *Proc. of the 6th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation*, pages 128–137, 1992.
- [15] J. Kelly Flanagan, *A New Methodology for Accurate Trace Collection and its Application to Memory Hierarchy Performance Modeling*, PhD thesis, Brigham Young University, December 1993.
- [16] K. Grimsrud, J. Archibald, M. Ripley, K. Flanagan, and B. Nelson, BACH: A hardware monitor for tracing microprocessor-based systems, *Microprocessors and Microsystems*, **17**(6), October 1993.
- [17] December 1995, A conversation with J. Thomas Pawlowski of Micron Semiconductor, SRAM Division.
- [18] Micron Semiconductor, *SRAM data book*, Micron Semiconductor, Inc., 1995.
- [19] Micron Semiconductor, *DRAM data book*, Micron Semiconductor, Inc., 1995.