Cache Performance for Multimedia Applications

Nathan T. Slingerland and Alan Jay Smith

Report No. UCB/CSD-00-1123

December 2000

Computer Science Division (EECS)

University of California

Berkeley, California 94720

Cache Performance for Multimedia Applications

Nathan T. Slingerland and Alan Jay Smith
{slingn, smith}@cs.berkeley.edu
Computer Science Division
EECS Department
University of California, Berkeley

December 21, 2000

Abstract

The caching behavior of multimedia applications has been described as having high instruction reference locality within small loops, very large working sets, and poor data cache performance due to non-locality of data references. Despite this, there is no published research deriving or measuring these qualities. Utilizing the previously developed Berkeley Multimedia Workload, we present the results of execution driven cache simulations with the goal of aiding future media processing architecture design. Our analysis examines the differences between multimedia and traditional applications in cache behavior. We find that multimedia applications actually exhibit lower instruction miss ratios and comparable data miss ratios when contrasted with other widely studied workloads. In addition, we find that longer data cache line sizes than are currently used would benefit multimedia processing.

1 Introduction

Multimedia is an amalgamation of various data types such as audio, 2D and 3D graphics, animation, images and video within a computing system or within a user application [Bhas97]. Put simply, a *multimedia application* is one which operates on data to be presented visually or aurally. The purpose of this work is to explore the cache behavior of real world multimedia applications. An important motivation is the widespread belief (seemingly without any actual basis in research) that data caches are not useful for multimedia applications because of the streaming nature of the data upon which they operate [Cont97], [Dief97], [Kuro98], [Lee96], [Rixn98]. The results presented in this paper strongly suggest that contemporary media processing applications perform no worse than traditional integer and floating point workloads.

Funding for this research has been provided by the State of California under the MICRO program, and by Cisco Corporation, Fujitsu Microelectronics, IBM, Intel Corporation, Maxtor Corporation, Microsoft Corporation, Sun Microsystems, Toshiba Corporation and Veritas Software Corporation.

2 Related Work

There have been a limited number of multimedia caching studies. [Sode97] studied the data cache behavior of MPEG-2 video decoding with the goal of optimizing playback performance through the cache sensitive handling of the data types used. It was found that although it has been suggested that caches are critically inefficient for video data (several media processor chips dispense with data caches entirely), there was sufficient reuse of values for caching to significantly reduce the raw required memory bandwidth. [Haku97], [Cox98], [Vart98] have studied the usefulness of caching the textures used in 3D rendering. A texture cache with a capacity as small as 16 KB has been found to reduce the required memory bandwidth three to fifteen times over a non-cached design and exhibit miss ratios around 1% [Haku97]. The addition of a larger second level of texture cache (2 MB) to a small first level cache (2 KB) can reduce the memory bandwidth from 475 MB/s to around 92 MB/s [Cox98].

There have been several studies of prefetching for multimedia. [Zuck96] examined several hardware data prefetching techniques for MPEG-1 (encoding and decoding) and MPEG-2 (decoding). Three hardware prefetching techniques were considered, with the most successful found to reduce the miss count by 70% to 90%. [Stru98] presents a combined hardware/software solution to prefetching for multimedia. Based on cycle accurate simulation of the Trimedia VLIW processor running a highly optimized video de-interlacing application, it was found that such a prefetching scheme was able to eliminate most data cache misses, with the effectiveness dependent on the timing parameters involved. [Cucc99] suggests a two-dimensional prefetching strategy for image data, due to the two separate degrees of spatial locality inherent in image processing (horizontal and vertical). When their 2D prefetching technique was applied to MPEG-2 decoding as well as two imaging applications (convolution and edge tracing), 2D prefetch was found to reduce the miss ratio more than one block look-ahead. Hardware implementation aspects of prefetching are discussed in [Tse98].

3 Multimedia Workload

3.1 Description

For our study of the cache behavior of multimedia applications, we employ the Berkeley Multimedia Workload, which we develop and characterize in [Sling00a]. A description of the component applications and data sets is given in Table 1. The main driving force behind application selection was to strive for completeness in covering as many types of media processing as possible. Open source software was used both for its portability (allowing for cross platform comparisons) as well as the fact that we could directly examine the source code.

The Berkeley workload represents the domains of *3D graphics* (Doom, Mesa, POVray), *document and image rendering* (Ghostscript, DjVu, JPEG), *broadband audio* (ADPCM, LAME, mpg123, Timidity), *speech* (Rsynth, GSM, Rasta) and *video* (MPEG-2). Three MPEG-2 data sets are included to cover Digital Video Disc (DVD) and High Definition Television or HDTV (720P, 1080I) resolutions. The parameters of the DVD, and HDTV data sets are listed in the table below:

Format	Aspect	Horizontal	Vertical	Frames
DVD	4:3	720	480	16
HDTV 720P	16:9	1280	720	16
HDTV 1080I	16:9	1920	1080	16

All of the applications in the Berkeley Multimedia Workload are written in C with the exception of DjVu which is coded in C++. Data sets were chosen to be on the order of real workloads, with long enough traces (instruction and data) to exercise very large caches. The trace lengths and other relevant simulation characteristics are listed in Table 2. Total simulation time for our work, not including false starts, machine down time and other simulation problems, was 24.4 days of CPU time for each multimedia run, once with and once without multiprogramming, and 147.2 days of CPU time for SPEC95 simulations, for a grand total of 196 days of CPU time. The machine used was a DEC AlphaStation 255 workstation with a 300 MHz Alpha 21064a).

As it is often necessary to reduce large volues of simulation results into a more easily digestible form, we will use averaging where necessary to compress results. Because the number of applications representing a particular application domain (audio, speech, document, video, 3D) is arbitrary, we will let each of the five application domains comprise a total of 20% of the averaged workload result, with the component applications of each domain being weighted equally.

3.2 Motivation

Motivating our detailed study of the cache behavior of multimedia applications is the large role memory latency plays in limiting performance. The memory hierarchy parameters of bandwidth and latency were measured for several current systems and are shown in Table 3 (the methodology used to obtain these parameters is detailed in the Appendix). Consider Table 4, which compares performance with caching against

the same system with all cache levels (L1 and L2) disabled. This was done by setting the appropriate BIOS parameters on our test system at boot time and then measuring the performance on real hardware. From this experiment we can see that modern microprocessor performance is highly dependent on an efficient memory hierarchy. The difference in latency between different levels of contemporary memory hierarchies is substantial, explaining the enormous slowdown we observed when the cache was disabled on our test system. Note that the system time (time spent in the operating system) slowdown is considerably less than that of the user time. This corroborates the generally held belief that the memory locality within operating system code is very poor, as it exhibits less of a performance degradation when caching is disabled.

Name	User Time Ratio	Sys Time Ratio
ADPCM Encode	25.0	3.6
ADPCM Decode	32.9	11.3
DJVVU Encode	56.7	3.6
DJVVU Decode	61.0	16.8
Doom	53.0	1.4
Ghostscript	63.7	34.7
GSM Encode	61.3	6.7
GSM Decode	77.8	16.5
JPEG Encode	103.4	1.3
JPEG Decode	103.0	10.5
LAME	80.3	1.4
Mesa Gears	44.2	11.0
Mesa Morph3D	35.9	35.0
Mesa Reflect	77.4	17.5
MPEG-2 Encode DVD	86.3	2.3
MPEG-2 Encode 720P	82.9	1.4
MPEG-2 Encode 1080I	86.5	1.5
MPEG-2 Decode DVD	94.1	9.3
MPEG-2 Decode 720P	95.1	5.9
MPEG-2 Decode 1080I	91.9	10.4
mpg123	83.7	7.5
POVray3	74.5	16.0
Rasta	83.8	7.0
Rsynth	86.5	27.0
Timidity	73.9	20.3
Arithmetic Mean	72.6	11.2
Geometric Mean	68.6	7.1

Table 4: **Uncached Performance Slowdown Factor** - ($t_{uncached}/t_{cached}$) when L1 and L2 caches were disabled on a 500 MHz AMD Athlon, 64 Kbyte L1 data cache, 64 Kbyte L1 instruction cache, 512 Kbyte L2 unified cache, 64-byte line size.

As another way to see the importance that caches play in modern computer performance, the actual measured total user time for each application was compared with the optimal user run time (a computed result based on the number and type of instructions executed) reported by the DEC ATOM *pixie* tool, which does not include memory latency and other effects in its measurements. Thus, the *pixie* results represent

Name	Description	Data Set
ADPCM	IMA ADPCM audio compression	Excerpt from Shchedrin's Carmen Suite, 28 sec., Mono, 16-bits, 44 kHz
DjVu	AT&T IW44 wavelet image compression	491x726 color digital photographic image
Doom	Classic first person shooter video game	25.8 sec. recorded game sequence (774 frames @ 30 fps)
Ghostscript	Postscript document viewing/rendering	First page of Rosenblum and Ousterhout's LFS paper (24.8 KB)
GSM	European GSM 06.10 speech compression	Speech by U.S. Vice President Gore, 24 sec., Mono, 16-bits, 8 kHz
JPEG	DCT based lossy image compression	491x726 color digital photographic image
LAME	MPEG-1 Layer III (MP3) audio encoder	Excerpt from Shchedrin's Carmen Suite, 28 sec., Stereo, 16-bits, 44 kHz
Mesa	OpenGL 3D rendering API clone	Animated gears, morph3d, reflect demos - 30 frames each at 1024x768
MPEG-2	MPEG-2 video encoding	16 frames (1 GOP) at DVD, HDTV 720P, HDTV 1080I resolutions
mpg123	MPEG-1 Layer III (MP3) audio decoder	Excerpt from Shchedrin's Carmen Suite, 28 sec., Stereo, 16-bits, 44 kHz
POVray	Persistence of Vision ray tracer	640x480 Ammonite scene by artist Robert A. Mickelson
Rasta	Speech recognition	2.128 sec. SPHERE audio file: "Laurie?...Yeah...Oh."
Rsynth	Klatt speech synthesizer	181 word excerpt of U.S. Declaration of Independence (90 sec., 1,062 bytes)
Timidity	MIDI music rendering with GUS instruments	X-files theme song, MIDI file (49 sec., 13,894 bytes), Goemon patch kit

Table 1: **Berkeley Multimedia Workload**

the best possible execution time for a given sequence of instructions. We assume that memory latency is the dominant factor for why measured performance does not match the theoretically optimal performance. The comparison is shown for DEC AlphaStation 255 workstations with a 300 MHz Alpha 21064a processor and 128 MB of RAM in Table 5.

4 Methodology

In order to measure cache miss ratios, we modified the LibCheetah v2.1 implementation [Aust] of the trace driven Cheetah cache simulator [Sugu93] to operate in an execution driven mode. It was also extended to allow for traces longer than 2^{31} references long. Cheetah simultaneously evaluates many alternative uniprocessor caches, but restricts the design options that can be varied. For each pass through an address trace, all of the caches evaluated must have the same block size, do no prefetching, and use the LRU or MIN replacement algorithms. Other cache simulators were also considered for this study (TychoII [WARTS], Dinero IV [Edler]), but were found to be considerably slower than Cheetah or otherwise unsuitable for use in execution driven simulation due to dynamic memory allocation issues. (See [Uhl97] and [Smit94] for overviews of trace driven simulation in general, and [Roth99b] for a comparison of the performance of a variety of execution and trace driven solutions.) DEC's ATOM [DEC] was used to instrument target applications with the modified Cheetah simulator, allowing for execution driven cache simulation.

4.1 Trace Length

Many cache studies utilize trace lengths that are a fraction of an application's total run time due the enormous simulation times required to account for every instruction and data cache reference. Unfortunately, short trace lengths are problematic because programs exhibit phase behavior; an effect which is easily seen in Figure 1.

Name	Measured (sec)	Optimal (sec)	Percent Difference
ADPCM Encode	0.324	0.260	19.85%
ADPCM Decode	0.225	0.162	27.82%
DJVVU Encode	2.482	1.592	35.86%
DJVVU Decode	1.848	1.164	37.01%
Doom	9.150	7.605	16.89%
Ghostscript	6.370	3.454	45.78%
GSM Encode	2.511	2.375	5.42%
GSM Decode	0.739	0.717	2.98%
JPEG Encode	1.046	0.873	16.56%
JPEG Decode	0.537	0.433	19.33%
LAME	71.300	40.680	42.95%
Mesa Gears	3.450	1.152	66.61%
Mesa Morph3D	2.950	0.998	66.18%
Mesa Reflect	17.660	13.970	20.89%
MPEG-2 Encode DVD	74.428	64.630	13.16%
MPEG-2 Encode 720P	201.663	172.500	14.46%
MPEG-2 Encode 1080I	460.429	397.700	13.62%
MPEG-2 Decode DVD	5.760	4.971	13.70%
MPEG-2 Decode 720P	17.980	15.280	15.02%
MPEG-2 Decode 1080I	36.230	30.360	16.20%
mpg123	3.830	2.620	31.59%
POVray3	66.442	32.780	50.66%
Rasta	0.183	0.112	39.02%
Rsynth	2.282	2.005	12.14%
Timidity	35.223	32.150	8.72%
Arithmetic Mean			26.10%
Geometric Mean			20.55%

Table 5: **Effect of Memory Latency on Execution Time** - DEC AlphaStation 255 with 300 MHz Alpha 21064a processor, 16 Kbyte L1 instruction cache, 16 Kbyte L1 data cache, 1 Mbyte L2 unified cache

The graphs depict the number of cache misses per 1,000,000 instructions executed for two sample applications. A second

Name	Instruction References	Load References	Store References	Cache Purge Interval	Data Time	User Time	System Time	Max Resident Set Size (kB)
ADPCM Enc.	64,020,339	4,302,782	616,116	708,037	27.818	0.102	0.036	1,472
ADPCM Dec.	49,687,192	4,302,782	1,229,491	708,037	27.818	0.054	0.067	1,472
DJVVU Enc.	394,242,073	68,204,647	27,458,767	4,754,521	-	0.700	0.033	41,664
DJVVU Dec.	328,761,829	59,700,283	31,845,270	4,754,521	-	0.484	0.037	20,992
Doom	1,889,897,116	500,225,773	109,222,846	4,284,671	25.800	2.216	0.939	26,432
Ghostscript*	970,395,449	188,116,952	96,837,718	1,227,194	-	1.190	0.164	32,192
GSM Enc.	375,971,389	55,009,077	14,010,892	297,641	24.341	0.468	0.016	1,024
GSM Dec.	126,489,950	10,711,683	3,812,483	297,641	24.341	0.209	0.014	1,024
JPEG Enc.	177,977,854	41,182,069	14,156,413	3,821,284	-	0.223	0.006	10,880
JPEG Dec.	80,176,365	16,419,065	4,585,079	3,821,284	-	0.093	0.024	10,880
LAME*	7,989,818,554	1,688,230,256	720,826,607	3,358,692	27.818	18.543	0.075	7,104
Mesa Gears*	296,287,705	36,839,087	38,449,257	2,173,610	1.000	0.484	0.039	50,240
Mesa Morph3D*	239,456,087	28,181,931	42,865,365	2,173,610	1.000	0.467	0.050	50,432
Mesa Reflect*	2,752,665,912	431,196,702	221,523,544	2,173,610	1.000	3.672	0.051	59,968
MPEG2 Enc. DVD	17,986,999,069	3,257,725,765	554,222,287	5,339,432	0.533	17.896	0.199	48,128
MPEG2 Enc. 720P	47,606,551,352	8,581,717,942	1,563,082,541	5,339,432	0.533	48.263	0.505	124,032
MPEG2 Enc. 1080I	111,041,463,652	20,148,301,625	3,349,482,784	5,339,432	0.533	113.050	0.521	277,952
MPEG2 Dec. DVD	1,307,000,398	219,595,775	76,688,056	1,055,372	0.533	1.911	0.051	16,512
MPEG2 Dec. 720P	3,992,213,571	673,343,544	243,881,680	1,055,372	0.533	5.796	0.141	41,472
MPEG2 Dec. 1080I	8,038,214,930	1,341,912,185	464,649,094	1,055,372	0.533	12.098	0.182	91,648
mpg123*	574,034,774	166,675,525	45,334,678	1,554,505	27.818	0.735	0.015	3,328
POVray3	6,017,197,975	1,562,189,592	683,690,648	5,928,433	0.033	11.296	0.121	16,000
Rasta*	25,120,492	5,925,648	1,989,604	2,560,537	2.128	0.039	0.014	5,632
Rsynth	402,500,964	102,351,142	39,223,906	594,438	99.680	0.780	0.004	7,808
Timidity	4,588,632,916	1,340,471,112	594,047,710	3,675,086	47.440	2.036	0.104	25,664
Total	217,315,777,907	40,532,832,944	8,943,732,836	-	-	242.805	3.406	-
Arithmetic Mean	8,692,631,116	1,621,313,318	357,749,313	-	-	9.712	0.136	-

Table 2: **Berkeley Multimedia Workload Simulation Characteristics** - *data time* (inherent time represented in data set - machine independent), *user time* (time spent processing in user space - machine dependent), and *system time* (time spent processing in system space on behalf of an application - machine dependent) in seconds. The *cache purge interval* is the number of instructions executed in each context interval before flushing the simulated cache. *Resident Set* is the maximum number of kilobytes in memory active at any one time, as determined by the `getrusage()` system call. All measurements were done on a DEC Alpha DS20 workstation with dual 500 MHz Alpha 21264 processors and 2048 MB of RAM running Compaq Tru64 Unix v5.0A (Rev. 1094). All applications were compiled with GCC v2.8.1 except (*) compiled with DEC C v5.6-075.

System	L1 $t_{latency}$	L1 r_{xfer}	L2 $t_{latency}$	L2 r_{xfer}	Main Mem $t_{latency}$	Main Mem r_{xfer}
Microstar* AMD Athlon (500 MHz)	4.0 ns	2657.18 MB/s	109.7 ns	1182.90 MB/s	242.5 ns	305.76 MB/s
DEC DS10 Alpha 21264 (466 MHz)	4.3 ns	1939.14 MB/s	30.4 ns	825.27 MB/s	197.2 ns	336.92 MB/s
BX** Intel Pentium III (450 MHz)	4.4 ns	1695.97 MB/s	46.6 ns	806.94 MB/s	149.8 ns	308.33 MB/s
HP N-Class PA-8500 (3 x 450 MHz)	4.6 ns	2190.42 MB/s	-	-	293.3 ns	338.50 MB/s

Table 3: **Memory Latency and Bandwidth** - where $t_{latency}$ is the time delay for any memory transaction, consisting primarily of memory latency and address transmission time and r_{xfer} is the bus transfer rate or bandwidth in bytes transferred per unit time. (*)Microstar - Microstar 6167 motherboard utilizing AMD's AMD-750 chipset, Mandrake Linux v7.0, 256 MB RAM (**)BX - unknown motherboard employing the Intel 440BX chipset, RedHat Linux v6.0, 128 MB RAM

difficulty with short trace lengths specific to cache simulations is the *cold start* problem. Cache simulation programs typically start with an empty cache which becomes filled as the simulation progresses. All initial memory accesses will miss the cache (*compulsory* misses), so cold start effects can potentially dominate if traces are too short to mitigate these effects.

Traces of a billion or more references may be needed to fully initialize multi-megabyte cache configurations [Kess91].

In order to be able to simulate the effects of a program's behavior, it is necessary to have a trace which captures all of its behavior. We found that although there are some applications (notably many of the SPEC92/95 benchmarks) that exhibit

Figure 1: **Example Cache Miss Profiles** - the number of data cache misses per million instructions executed (direct mapped, 8Kbyte cache with 32 byte line size)

uniform cache behavior over their entire run times, our multimedia workload applications did not share this property. The result of this is that full applications traces are the only way to completely characterize cache behavior. Our work traces application programs with realistic data sets for full execution runs. The trace lengths for the component Berkeley Multimedia Workload applications are given in Table 2.

4.2 Operating System Behavior

In general, studies including operating system behavior are rare because of the difficulty involved in obtaining this information. User space is freely manipulated, but tracing system space usually requires that modifications be made to the operating system. Table 2 lists the amount of time the CPU spends either in user space (*user time*) doing actual work for the application, or in system space (*system time*) serving I/O requests and dealing with other overhead on behalf of the application. Both user time and system time are machine dependent, and vary based on the instruction set, clock cycle length and other architectural parameters. *Data time* is machine independent, and is the implicit time length of the data set. For example, 24 frames of DVD movie represent one second of data time, even though it may take only 0.5 seconds of actual computation time (the sum of system and user time) to process the data. Although our traces only include user state references, this represents almost all of the memory behavior of the programs under study; less than 1% of our multimedia workload was system time. To some degree this may be an artifact of the nature of the Berkeley Multimedia Workload, which involves only file I/O. In an actual multimedia application where data must be transferred to and from I/O devices such as network, disk, or sound and video controller cards, a larger amount of OS activity could be present.

4.3 Multiprogramming

Despite the fact that the Berkeley Multimedia Workload is dominated by user time computation, because of multiprogramming we cannot ignore operating system behavior. Although a quantum length that depends on clock time or external events remains constant with architectural change (10 to 100 ms), the number of cycles Q in each quantum has increased over time for various reasons, including less efficient software and a speedup of the processor relative to the speed of real time events. The number of cycles in a quantum affects cache miss ratio. When a context switch occurs, the instructions and data of the newly scheduled process may no longer be in the cache from the last time it was run due to the memory use of programs scheduled in the interim.

The effect of multiprogramming can be roughly approximated by periodically flushing (clearing) a simulated cache. In order to correctly simulate the effect of multiprogramming for our multimedia workload, the average context switching interval for commercial (closed source) Microsoft Windows applications was measured on real hardware. The applications were chosen to correspond as closely as possible to those comprising the Berkeley Multimedia Workload, such that, for example, the context interval measured for actual DVD video playback was used in our simulations of MPEG-2 video decoding at DVD resolutions. The context switch intervals of the actual applications from the Berkeley Multimedia Workload were not measured because they are primarily file based applications, typically converting between compressed and uncompressed format without presenting the resulting data to the user. So, although the algorithms they employ (and therefore their memory access patterns) should for the most part be realistically similar to their commercial counterparts, their scheduling behavior is vastly different. Microsoft Windows NT and Windows 2000 both maintain a large amount of performance information for a large number of system objects

including context switch count, user time and system time per thread. By dividing the sum of system time and user time by the measured context switch count it was possible to compute the average context switch interval for each type of application. The details of these measurements are given in the Appendix, with the simulation quanta (cache flush intervals) applied to each application listed in Table 2.

For comparison purposes, the SPEC95 benchmark suite was also simulated. SPEC95 was simulated without multiprogramming (cache flushing) for several reasons. First, it is normally run in a uniprogrammed mode in order to extract the highest benchmark performance [Gee93]. More importantly, when we measured the actual context switch intervals for SPEC95 on a modern DEC Alpha workstation (DS20 with dual 500 MHz Alpha 21264 processors), the context interval, Q , was measured to be sufficiently large such (2.2 million instructions, on average) that multiprogramming has very little effect on miss ratios (unlike multimedia applications which typically have several tightly cooperating threads, the SPEC applications are single threaded and entirely compute bound).

Many UNIX-type operating systems maintain context switch counts on a per process basis which is accessible through the `getrusage()` system call. The average context switch interval was computed in the same manner as for the Windows multimedia applications. The measured context switch intervals for SPEC95 are presented in the Appendix.

The component applications for both the multimedia workload and SPEC95 were compiled for the Alpha AXP architecture running Digital UNIX v4.0E with the default optimization levels in the case of the multimedia workload, and the base optimization level for SPEC95 (the same compiler optimization flags on all applications: `-fast -O5 -non_shared`). The resulting binaries were then instrumented with the Cheetah cache simulator using ATOM and run on 300 MHz DEC Alpha AXP machines with 128 MB of RAM.

5 Simulation Results

The two major determinants of cache performance are *access time* (the latency from the beginning of an access until the time the requested data is retrieved) and *miss ratio* (the fraction of cache references which are not found in the cache) [Smit82]. Based on the latencies of a particular cache memory candidate design, in combination with the simulated or measured miss ratio, it is possible to select the design with the highest overall performance (lowest average memory access time) at some level of implementation cost.

Complete tables of the results from all of our simulations are available on the world wide web at <http://www.cs.berkeley.edu/~slingn/research/>, from which the cache performance of any application set of interest can be computed. For the sake of brevity, only overall average miss ratios are included in the Appendix.

5.1 Capacity

Capacity, or total cache size, has the greatest effect on miss ratio, and so it is one of the most important cache design parameters. Capacity, especially for L1 caches which are typically on the same die as the CPU, is limited by physical die size and implementation cost. In addition, the larger the capacity of a cache, the slower it is due to increased loading of critical address and data lines, thus requiring additional buffering [Przy88]. In order to study the effect of cache capacity on miss ratio, caches were simulated ranging in size from 1K to 2M bytes.

5.1.1 Other Workloads

The results of other studies, with a variety of workloads, on the effect of cache size on the miss ratio are presented alongside our simulation results for the Berkeley Multimedia Workload. All of the miss ratios presented in Figures 2, 3, and 4 are for caches with a line size of 32 bytes and two-way associativity, which represent common values for these parameters in current cache systems. Because the results shown have been gathered from a motley assortment of studies of disparate ages and architectures, many of which did not analyze configurations precisely identical to ours in terms of line size and associativity, we use adjusted results from [Gee93]. These adjustments modify the original results of the studies according to the ratios of miss ratios found in [Hill89] for differences in associativity, and [Smit87] for variations in line size. Extensions to larger cache sizes were made for the DTMR results using the $\sqrt{2}$ rule from [Smit82]. Each of the workloads is described in detail in the Appendix. It is important to note that many of the other studies included for comparison purposes also measured or simulated multiprogramming behavior, but because they are based on older machine architectures, their Q (quantum) lengths and therefore their context switch intervals are significantly shorter than those used in our simulations.

Figure 2: Unified Cache Miss Ratio - 32 byte line size

Figure 3: Instruction Cache Miss Ratio - 32 byte line size

Figure 4: Data Cache Miss Ratio - 32 byte line size

The most significant result of Figures 2, 3, and 4 is that far from multimedia applications exhibiting degenerate cache behavior in comparison to more traditional workloads, our results demonstrate that they actually perform better for nearly all examined cache configurations. We believe that this is attributable to several factors. First, most of the comparison workloads are for timeshared machines on which task switching between users occurred very frequently. Further, the comparison studies are of architectures with much lower clock speeds than modern processors, and so exhibit higher miss ratios due to shorter context switch intervals based on real time periods. Even so, the uniprogrammed SPEC92 and SPEC95 benchmarks still demonstrate higher miss ratios than our multimedia workload. We believe that this is because many multimedia algorithm building blocks (such as the discrete cosine transform and fast Fourier transform) internally reference the same data locations repeatedly. In the case of

streaming multimedia applications, data is typically copied into a fixed region of memory (buffer) from the source file of network interface device. Even algorithms which simply traverse enormous arrays of data without re-referencing (color space conversion, subsampling) typically do so in linear memory order, and so benefit greatly from the “prefetching” effect of long cache lines. In addition, multimedia data types are typically small (8-bits for video and speech, 16-bits for audio, single precision (32-bit) floating point for 3D geometry calculations). This means that in comparison to the other workloads which utilize full 32-bit integers or 64-bit (double precision) floating point, more multimedia data elements fit in a single cache line, thus improving the relative hit ratio. Finally, it is also possible that the memory behavior of the Berkeley Multimedia Workload component applications have had their memory behavior tuned in order to avoid cache problems.

5.1.2 Multimedia Domains

When broken down into the five application domains (audio, speech, document, video and 3D graphics), some important trends become apparent (Figure 5). Instruction cache miss ratios are quite similar across the various application domains, with a 16 KB or 32 KB cache being sufficient. This supports the idea that multimedia applications are dominated by small kernel loops, rather than large code sizes. Data cache miss ratios show significant variation between domains. Speech, video, and audio domains exhibit similar (low miss ratio) cache performance, while the document and 3D applications have higher miss ratios. This is attributable to the non-linear way in which data sets are traversed during processing for these applications.

5.1.3 SIMD Effects

The motivation behind SIMD within a register approach taken by multimedia extensions such as Intel’s MMX or Motorola’s AltiVec is the fact that on general purpose microprocessors data paths are typically 32 or 64-bits wide, while multimedia applications typically deal with narrower width data. By packing multiple narrow operations into the wider native processor data path, it is possible to improve multimedia performance.

Although it might be expected that current scalar compilers would place multiple short values into a register and then extract them with register to register operations in order to minimize memory access overhead, we found that this was not the case for the compilers available on our DEC Alpha test platform. Instead, multiple independent short loads are issued. Because of this, the use of SIMD instruction set extensions for multimedia will result in higher cache miss ratios, although the total number of memory references would decrease, due to the folding of several scalar load operations into a single parallel operation for sub-word data types which are adjacent in memory. Note that programs employing multimedia (SIMD) instruction sets are likely to be handcoded, as no currently available commercial compilers are able to gen-

Figure 5: **Multimedia Domains** - Line size is 32 bytes. The domains consist of *3D graphics* (Doom, Mesa, POVray), *document* (Ghostscript, DjVu, JPEG), *audio* (ADPCM, LAME, mpg123, Timidity), *speech* (Rsynth, GSM, Rasta) and *video* (MPEG-2).

erate them automatically; this will also effect their memory behavior.

5.2 Line Size

The block or line size of a cache memory is another cache design parameter that strongly affects cache performance [Smit87]. Generally, increasing the line size decreases the miss ratio, since each fetch from memory retrieves more data, thus fewer accesses outside the cache are required. When the line size is made too large, *memory pollution* can adversely affect cache performance, causing material to be loaded that is either never referenced or evicting information that would have been referenced before being replaced. Large lines also decrease the likelihood of “line crossers” - multibyte memory accesses across the boundary between two cache lines, such as occur with many CISC architectures. This type of access incurs a performance penalty since it usually requires two cache accesses; string operations can induce multiple cache data misses. Additionally, small line sizes require a greater number of bits be dedicated to tag space than for larger lines, although a *sector* or *sub-block* cache is one way to avoid this problem. (See [Roth99a] for an investigation into sub-sector cache design issues.)

In addition to affecting the performance metric of miss ratio, large line sizes can have long transfer times and create excessively high levels of memory traffic [Smit87]. It is possible to model the time to fetch a cache line, t_{line} , assuming no prefetching and that all loads load a full cache line:

$$t_{line} = t_{latency} + \frac{\left(\frac{L}{d}\right)}{r_{xfer}} \quad (1)$$

where,

L - line size (bytes)

d - data path width to memory (bytes)

$t_{latency}$ - delay for any memory transaction, consisting primarily of memory latency and address transmission time (seconds)

r_{xfer} - bus transfer rate or bandwidth (bytes per second)

In order to select an optimal line size, it is necessary to minimize $t_{line} \cdot m(L)$, where $m(L)$ is the miss ratio as a function of line size. To investigate the effect of line size choice on miss ratio, instruction and data caches were simulated with line sizes ranging from 16 bytes to 256 bytes and total capacities ranging from 1 KB to 2 MB. As an example, we used the parameters measured for the memory hierarchy on a 500 MHz AMD Athlon system, as listed in Table 3. Because we are only considering one level caches in this work, we use the measured L2 parameters for the memory miss latency and bandwidth. For every cache capacity there is an optimal line size that minimizes the average memory reference delay. In the case of the largest caches simulated (1M and 2M capacity), the largest line size of 256 bytes produced minimal average delay for instruction caches. Table 6 summarize the mean memory reference delay for the multimedia workload for SPEC92, and SPEC95, in addition to the Berkeley Multimedia Workload. The best values are highlighted in bold text. Some of the instruction cache results exhibit anomalies for extremely small miss ratios due to the limited precision of our results in these instances (only a few misses for many millions of instruction references).

Our results indicate that for the Berkeley Multimedia Workload (as well as SPEC95), instruction cache line sizes should be as large as possible, due to the extremely low miss ratios exhibited for even moderate capacities. Instructions are likely to be accessed sequentially, so the fetching of large line sizes pays off. Data caches, on the other hand, have clearly optimal line sizes, depending on the total cache capacity. In the selection of an optimal line size, it should be kept in mind that large line sizes can be problematic in multiprocessor systems where system bus bandwidth must be shared. Very long line sizes may also cause real-time problems, as when I/O operations cause buffer overruns due to an inability to get on the memory bus. With many desktop computer manufacturers already offering 2 and even 4-way multiprocessor support,

this may have a limiting effect on the usefulness of long cache lines.

5.3 Associativity

Determining optimal associativity is important because changing associativity has a significant impact on cache performance (latency) and cost. Increasing set associativity may require additional multiplexing in the data path as well as increasing the complexity of timing and control [Przy88]. [Hill89] developed a rule of thumb for how associativity affects miss ratio: reducing associativity from eight-way to four-way, from four-way to two-way, and from two-way to direct mapped was found to cause relative miss ratio increases of approximately 5, 10, and 30 percent, respectively. In order to see how associativity affects miss ratios for our multimedia workload, *miss ratio spreads* were calculated for unified, data and instruction caches for our suite of multimedia applications. Miss ratio spread computes the benefit of increasing associativity, and is defined in [Hill89]:

$$\text{Miss Ratio Spread} = \frac{m(A = n) - m(A = 2n)}{m(A = 2n)} \quad (2)$$

Where $m(A = n)$ is the miss ratio for n -way set associativity, A . As in [Hill89], a block size of 32 bytes was chosen, with all simulated caches utilizing the LRU replacement algorithm. The miss ratio spreads of the Berkeley Multimedia Workload as well as SPEC92 and SPEC95 are shown in Figure 6. Please note that in order to preserve visual detail across the wide range of workload behaviors observed, the subfigures use different vertical scales. Unlike the original [Hill89] study, our curves are not smoothed or averaged.

From the miss ratio spread results in Figure 6, we can see that instruction caches for multimedia applications (and generally for SPEC92 and SPEC95) benefit from 2- or 4-way associativity for moderate size caches (16 KB to 256 KB). (Note that the scale is different for each portion of the figure.) For the multimedia workload, most of the benefit from associativity seems to be obtained with two-way set associativity; additional associativity doesn't seem to improve performance significantly, except for small cache sizes. Note that increasing associativity can also be a useful way to increase overall cache capacity when limited by virtual memory constraints (a limited number of page offset bits to index the cache). This was the approach taken both by the designers of Motorola's G4 processor (which includes 8-way associative L1 caches) as well as the IBM 3033 which has a 16-way associative 64k cache.

6 Future Directions

The final determination we would like to make is what cache designers should plan for in order to support future multimedia applications. We look at this in terms of the potential for data set expansion within each multimedia application domain. We expect that audio and speech application data sets will not change significantly in size, as current data sets are already at the limit of human audio fidelity. Document

processing should also not change as current documents are sufficient for either printing or previewing at laser printer resolutions.

Video resolutions are not yet close to the limits of the human eye. This can be seen in the upcoming high resolution digital formats currently in the pipeline for consumer level products: DVD (720x480), HDTV 720P (1280x720), and HDTV 1080I (1920x1080). In order to determine if the working set size of video applications is increasing, and therefore larger cache capacities are necessary to support these new resolutions, we compared the effect of cache capacity on miss ratios for them in Figures 7 and 8 utilizing the ratio of miss ratios for increasing resolution. Our results were obtained by running the same MPEG-2 decoding and encoding applications with three 30 frame data sets at DVD, HDTV 720P and HDTV 1080I resolutions. As an example of how to interpret the figures, DVD $\Rightarrow$ 720P refers to the ratio of miss ratios of 720P/DVD resolutions. This metric shows the relative change in miss ratio for the higher resolution compared to the preceding lower resolution.

From Figure 7, we can see that instruction miss ratios are hardly affected at all by changes in resolution and although there are some minor fluctuations, the ratios are generally quite close to 1.0. Data miss ratios (Figure 8) show a stronger influence for small caches (capacities less than 32K), but level off for larger caches. This shows that the type of data locality being exploited by data caches for digital video is at the MPEG video block or macroblock level (which are the same size in all formats) rather than the frame level since caches are equally effective on all resolutions above a certain minimum working set size.

Figure 7: **Digital Video Trends: Instruction Cache** - ratio of miss ratios for increasing resolution

Previous research ([Haku97], [Cox98], [Vart98]) has found that even a small texture cache located on a 3D accelerator board reduces the required bandwidth to main memory significantly. Past architectural trends suggest that all 3D rendering functionality will eventually be folded into the main

Multimedia					
Instruction Cache					
Block Size (bytes)					
Size	16	32	64	128	256
1K	6.22630	3.79536	2.68253	1.96086	1.74648
2K	3.13298	1.94226	1.40646	1.01556	0.93899
4K	1.67616	1.08187	0.81106	0.60246	0.57495
8K	0.95800	0.64909	0.46229	0.35912	0.33620
16K	0.43464	0.28182	0.19453	0.15618	0.15525
32K	0.16759	0.10355	0.07412	0.05721	0.04810
64K	0.09868	0.05709	0.03657	0.02492	0.01902
128K	0.07714	0.04281	0.02534	0.01516	0.01016
256K	0.07514	0.04126	0.02407	0.01393	0.00897
512K	0.07496	0.04110	0.02392	0.01379	0.00883
1M	0.07496	0.04110	0.02392	0.01379	0.00882
2M	0.07496	0.04110	0.02392	0.01379	0.00882
Data Cache					
Size	16	32	64	128	256
1K	10.70698	9.09775	10.17477	14.66781	24.87838
2K	8.04009	6.20969	6.31052	8.56975	14.26147
4K	6.15572	4.35004	3.83097	4.40862	6.87190
8K	4.64852	3.06934	2.40616	2.44560	3.27708
16K	3.48517	2.24199	1.61893	1.51314	1.86139
32K	2.81276	1.77501	1.24353	1.05964	1.14475
64K	2.41197	1.49988	1.03259	0.82706	0.78846
128K	2.30867	1.38347	0.91999	0.71591	0.65241
256K	2.23803	1.31936	0.85758	0.64236	0.57225
512K	2.18862	1.27134	0.80799	0.58706	0.50072
1M	1.94165	1.02288	0.55434	0.31764	0.19900
2M	1.93021	1.01452	0.54805	0.31211	0.19121
Unified Cache					
Size	16	32	64	128	256
1K	10.28672	7.50803	6.78065	7.94751	11.89479
2K	6.56454	4.76822	4.25393	4.83131	6.63532
4K	3.94739	2.86512	2.46797	2.81398	3.51678
8K	2.47522	1.74480	1.39840	1.38439	1.70539
16K	1.54548	1.03360	0.78006	0.77335	0.94230
32K	1.06232	0.69658	0.49541	0.44901	0.48632
64K	0.75353	0.48619	0.34184	0.28650	0.27540
128K	0.61073	0.38293	0.26271	0.21472	0.19302
256K	0.55575	0.32711	0.21159	0.15799	0.13967
512K	0.53903	0.31085	0.19540	0.13926	0.11636
1M	0.48135	0.25490	0.13870	0.07990	0.05011
2M	0.47826	0.25256	0.13686	0.07821	0.04800

SPEC92					
Instruction Cache					
Block Size (bytes)					
Size	16	32	64	128	256
1K	3.71840	2.17213	1.33401	0.99493	0.77374
2K	2.70159	1.54599	0.94898	0.63635	0.47634
4K	1.90405	1.05308	0.62611	0.40832	0.29344
8K	1.15215	0.62794	0.36595	0.23302	0.16946
16K	0.88056	0.46364	0.25446	0.14906	0.09699
32K	0.33792	0.17852	0.09883	0.05834	0.03805
64K	0.02737	0.01647	0.01171	0.00807	0.00689
128K	0.00734	0.00519	0.00359	0.00318	0.00216
256K	0.00178	0.00124	0.00069	0.00012	0.00014
512K	0.00122	0.00067	0.00012	0.00012	0.00014
1M					
2M					
Data Cache					
Size	16	32	64	128	256
1K	20.96124	19.39993	19.84360	23.28471	30.41951
2K	17.25565	16.02179	16.25980	18.30165	23.07412
4K	11.54032	10.47322	10.94480	12.22563	15.48550
8K	8.27219	6.05842	5.26856	5.28466	6.37997
16K	6.55353	4.29003	3.35500	3.00883	3.30687
32K	5.35748	3.70255	2.49839	2.12650	2.21407
64K	4.33509	2.67295	1.90952	1.53742	1.55691
128K	4.03750	1.84523	1.23878	0.92322	0.84286
256K	1.95191	1.08356	0.65050	0.43468	0.34085
512K	1.47494	0.77360	0.42169	0.24833	0.16336
1M					
2M					
Unified Cache					
Size	16	32	64	128	256
1K	10.24520	8.61222	8.09564	9.04907	11.75792
2K	7.78352	6.44349	6.00378	6.40153	7.89891
4K	5.28845	4.25106	3.97639	4.21803	5.13135
8K	3.50912	2.45442	2.00881	1.95240	2.27774
16K	2.52124	1.57757	1.17196	1.02740	1.11110
32K	1.77878	1.08804	0.77880	0.67819	0.65592
64K	1.21944	0.74588	0.52501	0.42360	0.42752
128K	0.86048	0.51497	0.34442	0.25689	0.23311
256K	0.54715	0.30834	0.18589	0.12634	0.10034
512K	0.40357	0.21293	0.11622	0.06996	0.04615
1M					
2M					

SPEC95					
Instruction Cache					
Block Size (bytes)					
Size	16	32	64	128	256
1K	14.27069	7.91963	4.97274	4.35598	2.87418
2K	9.98204	5.53019	3.48003	2.85900	1.89096
4K	5.59066	3.17658	2.10667	1.75073	1.23805
8K	3.20456	1.77827	1.18120	0.98336	0.72259
16K	1.81675	0.98199	0.64224	0.51674	0.38979
32K	0.58192	0.31805	0.22947	0.22521	0.16785
64K	0.14942	0.06065	0.04621	0.08140	0.05070
128K	0.09541	0.01656	0.01220	0.03635	0.00956
256K	0.07924	0.00570	0.00406	0.02055	0.00286
512K	0.07324	0.00188	0.00137	0.01320	0.00096
1M	0.07106	0.00045	0.00035	0.00874	0.00030
2M	0.07071	0.00019	0.00016	0.00561	0.00014
Data Cache					
Size	16	32	64	128	256
1K	28.39914	26.87923	29.13518	35.47889	49.55017
2K	21.39439	19.36667	20.11207	24.15869	33.54895
4K	16.71953	14.11966	14.10880	16.75037	22.38782
8K	13.18879	10.41741	10.04470	11.51744	15.18622
16K	10.39420	6.84679	6.48190	7.54309	10.22041
32K	8.88017	5.03129	3.99639	4.41458	6.26585
64K	7.80708	3.92517	2.53108	2.49027	3.74590
128K	7.21604	3.49942	2.14027	1.49853	1.43653
256K	6.57932	3.05452	1.78869	1.17317	0.90454
512K	5.87008	2.53829	1.39700	0.82849	0.55867
1M	5.29881	2.18049	1.14680	0.62642	0.36857
2M	4.57430	1.78668	0.93162	0.49926	0.28320
Unified Cache					
Size	16	32	64	128	256
1K	23.79340	18.25885	16.44461	17.68433	23.34513
2K	17.96932	13.42188	11.77179	12.32043	15.71593
4K	12.51167	9.38807	8.12927	8.38950	10.45317
8K	8.50513	6.38475	5.44560	5.51777	6.80582
16K	5.35151	3.75193	3.22112	3.39591	4.25854
32K	3.56756	2.31962	1.82203	1.89118	2.48627
64K	2.53677	1.48147	0.99837	0.97665	1.39368
128K	2.08627	1.17987	0.73081	0.52139	0.51386
256K	1.81360	0.99893	0.58455	0.38362	0.29702
512K	1.56373	0.82786	0.45573	0.27048	0.18301
1M	1.37100	0.70968	0.37403	0.20508	0.12160
2M	1.13350	0.58111	0.30338	0.16290	0.09276

Table 6: Average Delay per Memory Reference (ns)

Figure 8: Digital Video Trends: Data Cache - ratio of miss ratios for increasing resolution

processor, at such time as there is adequate silicon to devote to it. We found that 3D applications exhibited the poorest locality of all the multimedia application types examined. Moving 3D functionality entirely into the CPU (and therefore sharing the cache with other applications) may require the reorganization of program structures to render vertices in an order amenable to LRU caching ([Haku97] examines several approaches for doing this) or larger caches to hold the substantial working sets of such applications. Texture size is dependent more upon the quality of rendered output rather than on display resolution, and is therefore subject to great

pressure for growth [Intel].

7 Summary

In this paper we have provided a thorough analysis of three important cache parameters in order to support multimedia applications: cache capacity, line size and set associativity. Using execution driven simulation, a large design space was simulated incorporating multiprogramming effects.

7.1 Capacity

A moderate instruction cache capacity of 16 KB or 32 KB was found to be sufficient for all of the applications in our multimedia workload. Despite the widespread misconception that multimedia applications exhibit poor data cache performance, the Berkeley Multimedia Workload was found to exhibit quite low miss ratios. Optimal data cache size depends on the type of multimedia applications that are of interest. For the most common audio, speech and video multimedia applications, a data cache of 32 KB in capacity is large enough to exhibit extremely low (<1%) miss ratios. Document and 3D processing exhibit less locality, and in fact even the largest cache sizes simulated (2 MB) still suffered significant misses for 3D graphics. As mentioned, this is due in large part to the fact that 3D graphics primitives (vertices) are processed in object order rather than memory order, leading to poor memory referencing behavior.

7.2 Line Size

We found benefit in instruction cache block sizes as large as the largest in our study (256 bytes) for the memory techni-

Figure 6: **Berkeley Multimedia, SPEC92 and SPEC95 Miss Ratio Spreads** - Each line, labeled N to M , indicates the fraction increase in miss ratio when reducing associativity from N -way to M -way. To preserve detail across the wide range of workload behaviours observed, different vertical scales are used in each graph.

ogy examined at any cache capacity. Data cache block size selection is more dependent on the capacity of the cache. It is important to note that our block size choices considered only average memory access time, and did not consider issues such as total memory traffic or bus busy periods, which are important considerations for multiprocessor machines.

7.3 Associativity

Based on the results of the miss ratio spread analysis, instruction caches can optimally benefit from 2- or 4-way associativity for most moderate cache sizes (16 KB to 256 KB). Data cache benefits from varying degrees of associativity are more difficult to generalize and appear to be highly dependent on the specific workload, but in general, 2-4 way associativity is also a good choice.

References

- [Aust] Todd M. Austin, Doug Burger, Manoj Franklin, Scott Breach, Kevin Skadron, "The SimpleScalar Architectural Research Tool Set," <http://www.cs.wisc.edu/~mscalar/simplescalar.html>, retrieved April 24, 2000
- [Bhas97] Vasudev Bhaskaran, Konstantinos Konstantinides, and Balas Natarajan, "Multimedia architectures: from desktop systems to portable appliances," *Proc. Multimedia Hardware Architectures*, San Jose, California, February 2-14, 1997, *SPIE Vol. 3021*, pp. 14-25
- [Cont97] Thomas M. Conte, Pradeep K. Dubey, Matthew D. Jennings, Ruby B. Lee, Alex Peleg, Salliah Rathnam, Mike Schlansker, Peter Song, Andrew Wolfe, "Challenges to Combining General-Purpose and Multimedia Processors," *IEEE Computer*, Vol. 30, No. 12, December 1997, pp. 33-37
- [Cox98] Michael Cox, Narendra Bhandari, Michael Shantz, "Multi-Level Texture Caching for 3D Graphics Hardware," *Proc. 25th Int'l Symp. on Computer Architecture*, Barcelona, Spain, June 27-July 1, 1998, pp. 86-97
- [Cucc99] Rita Cucchiara, Massimo Piccardi, Andrea Prati, "Exploiting Cache in Multimedia," *Proc. of IEEE Multimedia Systems '99 Vol. 1*, Florence, Italy, July 7-11 1999, pp. 345-350

- [DEC] Digital Equipment Corporation, *ATOM Reference Manual*, <http://www.partner.digital.com/www-swdev/files/DECOSF1/Docs/Other/ATOM/ref.ps> retrieved April 24, 2000
- [Dief97] Keith Diefendorff, Pradeep K. Dubey, "How Multimedia Workloads Will Change Processor Design," *IEEE Computer*, Vol. 30, No. 9, September 1997, pp. 43-45
- [Edler] Jan Edler, Mark D. Hill, "Dinero IV Trace-Driven Uniprocessor Cache Simulator," <http://www.neci.nj.nec.com/homepages/edler/d4/>, retrieved April 24, 2000
- [Gee93] Jeffrey D. Gee, Mark D. Hill, Dionisios N. Pnevmatikatos, Alan Jay Smith, "Cache Performance of the SPEC92 Benchmark Suite," *IEEE Micro*, Vol. 13, No. 4, August 1993, pp. 17-27
- [Haku97] Ziyad S. Hakura, Annap Gupta, "The Design and Analysis of a Cache Architecture for Texture Mapping," *Proc. 24th Int'l Symp. on Computer Architecture*, Denver, Colorado, June 2-4, 1997, pp. 108-120
- [Hill89] Mark D. Hill, Alan Jay Smith, "Evaluating Associativity in CPU Caches," *IEEE Trans. on Computers*, Vol. 38, No. 12, December 1989, pp. 1612-1630
- [Intel] Intel Corporation, "Accelerated Graphics Port Interface Specification v2.0," May 4, 1998, <http://developer.intel.com/technology/agp/>, retrieved April 24, 2000
- [Kess91] Richard Eugene Kessler, "Analysis of Multi-Megabyte Secondary CPU Cache Memories," *University of Wisconsin-Madison Computer Sciences Technical Report #1032*, July 1991
- [Kuro98] Ichiro Kuroda, Takao Nishitani, "Multimedia Processors," *Proc. IEEE*, Vol. 86, No. 6, June 1998, pp. 1203-1221
- [Lee96] Ruby B. Lee, Michael D. Smith, "Media Processing: A New Design Target," *IEEE Micro*, Vol. 16, No. 4, August 1996, pp. 6-9
- [Przy88] Steven Przybylski, Mark Horowitz, John Hennessy, "Performance Tradeoffs in Cache Design," *Proc. 15th Annual Int'l Symp. on Computer Architecture*, Honolulu Hawaii, May 30-June 2, 1988, pp. 290-298
- [Rixn98] Scott Rixner, William J. Dally, Ujval J. Kapasi, Bruce Khailany, Abelardo Lopez-Lagunas, Peter R. Mattson, John D. Owens, "A Bandwidth-Efficient Architecture for Media Processing," *Proc. 31st Int'l Symp. on Microarchitecture*, Dallas, Texas, November 30-December 2, 1998, pp. 3-13
- [Roth99a] Jeffrey B. Rothman, Alan Jay Smith, "The Pool of Subsectors Cache Design," *Proc. Int'l Conference on Supercomputing*, Rhodes, Greece, June 20-25 1999, pp. 31-42
- [Roth99b] Jeffrey B. Rothman, Alan Jay Smith, "Multiprocessor Memory Reference Generation Using Cerberus," *Proc. 7th Int'l Symp. on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'99)*, College Park, Maryland, October 24-28, 1999, pp. 278-287
- [Sling00a] Nathan T. Slingerland, Alan Jay Smith, "Design and Characterization of the Berkeley Multimedia Workload," *University of California at Berkeley Technical Report CSD-00-1122*, December 2000
- [Smit82] Alan Jay Smith, "Cache Memories," *ACM Computing Surveys*, Vol. 14, No. 3, September 1982, pp. 473-530
- [Smit87] Alan Jay Smith, "Line (Block) Size Choice for CPU Cache Memories," *IEEE Trans. on Computers*, Vol. C-36, No. 9, September 1987, pp. 1063-1075
- [Smit94] Alan Jay Smith, "Trace Driven Simulation in Research on Computer Architecture and Operating Systems," *Proc. Conference on New Directions in Simulation for Manufacturing and Communications*, Tokyo, Japan, August 1994, pp. 43-49
- [Sode97] Peter Soderquist, Miriam Leeser, "Optimizing the Data Cache Performance of a Software MPEG-2 Video Decoder," *Proc. ACM Multimedia '97*, Seattle, Washington, November 9-13, 1997, pp. 291-301
- [Stru98] Pieter Struik, Pieter van der Wolf, Andy D. Pimentel, "A Combined Hardware/Software Solution for Stream Prefetching in Multimedia Applications," *Proc. 10th Annual Symp. on Electronic Imaging*, San Jose, California, January 1998, pp. 120-130
- [Sugu93] Rabin A. Sugumar, Santosh G. Abraham, "Efficient Simulation of Caches under Optimal Replacement with Applications to Miss Characterization," *Proc. 1993 ACM Sigmetrics Conference on Measurements and Modeling of Computer Systems*, May 1993, pp. 24-35
- [Tse98] J. Tse, A. J. Smith, "CPU cache prefetching: Timing evaluation of hardware implementations," *IEEE Trans. on Computers*, Vol. 47, No. 5, May 1998, pp. 509-26
- [Uhli97] Richard Uhlig, Trevor Mudge, "Trace-Driven Memory Simulation: A Survey," *ACM Computing Surveys*, Vol. 29, No. 2, June 1997, pp. 128-170
- [Vart98] Alexis Vartanian, Jean-Luc Bechennec, Natalie Drach-Temam, "Evaluation of High Performance Multicache Parallel Texture Mapping," *Proc. 1998 Int'l Conference on Supercomputing*, Melbourne, Australia, July 13-17, 1998, pp. 289-296
- [WARTS] Glenn Ammons, Tom Ball, Mark Hill, Babak Falsafi, Steve Huss-Lederman, James Larus, Alvin Lebeck, Mike Litzkow, Shubendu Mukherjee, Steven Reinhardt, Madhusudhan Talluri, and David Wood, "WARTS: Wisconsin Architectural Research Tool Set," <http://www.cs.wisc.edu/~larus/warts.html>, retrieved April 24, 2000
- [Zuck96] Daniel F. Zucker, Michael J. Flynn, and Ruby B. Lee, "A Comparison of Hardware Prefetching Techniques for Multimedia Benchmarks," *Proc. 3rd IEEE Int'l Conference on Multimedia Computing and Systems*, Hiroshima, Japan, June 17-23, 1996, pp. 236-244

1 Appendix

1.1 Context Switch Intervals

[Agar88] found that actual multiprogramming traces showed a wide variation in time slice duration and execution order of processes. Although the level of multiprogramming on a desktop workstation is typically low, multimedia applications are often multi-threaded, such that in the example of DVD playback, one thread handles bit stream parsing, another video decoding, and still another audio decoding. Multimedia applications are time dependent; their output or input of processed multimedia data must be synchronized with production or consumption of that data. For this reason, the length of the *context interval* (utilized portion of the quantum) depends on I/O device interaction, number of component threads, and amount of computation time necessary to operate on a given unit of data time.

1.1.1 Windows 2000 Multimedia

In order to estimate typical context switch intervals for real multimedia applications, a set of commercial multimedia applications corresponding to those in the Berkeley Multimedia Workload were measured. All runs were with a single active process running. The context switch intervals of the actual applications from the Berkeley Multimedia Workload were not measured because they are primarily file based applications, typically converting between compressed and uncompressed format without presenting the resulting data to the user. So, although the algorithms they employ (and therefore their memory access patterns) should for the most part be realistically similar to their commercial counterparts, their scheduling behavior is vastly different. For example, in the case of on screen DVD movie play back, there are typically several concurrent threads of execution, each dealing with a particular aspect of MPEG-2 decoding (e.g. audio, video, bitstream parsing/demuxing). Acceptable playback requires that for each unit of data set time each decoding component must be computed:

1. fast enough to leave time for computing the other components in that unit of time (otherwise video frames may need to be dropped)
2. with sufficient precision to not introduce noticable artifacts
3. within a given time frame to prevent latency effects from disrupting the perceived synchronization between audio and video.

These requirement affect scheduling, and are not taken into account in an application which operates in a batch or offline mode.

Context switch intervals were measured on a 500 MHz AMD Athlon with 256 MB of PC100 DRAM and an MSI MS-6167 motherboard running Windows 2000 Professional v5.00.2195. Both a sound card (Sound Blaster Live Value)

and 3D accelerator card (AGP Nvidia Riva TNT) were installed. Table 2 lists these intervals as measured by the Windows 2000 performance counters, which return results in terms of time (CPU cycles).

CINT95	Context Interval
099.go	21,134,208
124.m88ksim	5,122,455
126.gcc	3,845,678
129.compress	22,719,364
130.li	21,754,551
132.jpeg	16,093,926
134.perl	16,308,625
147.vortex	13,193,123
CFP95	Context Interval
101.tomcatv	10,185,364
102.swim	13,753,700
103.su2cor	9,595,431
104.hydro2d	18,624,108
107.mgrid	17,791,106
110.applu	4,644,660
125.turb3d	22,366,853
141.apsi	11,743,787
145.fpppp	19,004,011
146.wave5	19,015,575
Arithmetic Mean	2,243,433
Geometric Mean	2,238,412

Table 1: **Average SPEC95 Context Switch Intervals** - measurements are given in 500 MHz clock cycles

In our cache simulations, we simulate normal task switching by flushing the cache every $Q_{context}$ instructions. Because our cache simulation is instruction, rather than cycle based, we require cache purge intervals measured in terms of instructions executed between cache flushes. In order to convert our context switch interval data from cycles to instructions we need to know the corresponding cycles per instruction (CPI) ratio. However, we can not simply treat x86 CISC instructions as being equivalent to RISC Alpha instructions, due to the inherently different amounts of work done by each class of instructions. Interestingly, although the x86 instruction set architecture that the AMD Athlon executes is CISC in nature, the full CISC instructions are internally decoded by the Athlon processor into RISC-like instructions referred to as *micro-ops* or simply *ops* [AMD]. Thus, in order to approximate the equivalent number of Alpha RISC-like instructions in each context switch interval, we divide the number of cycles by the number of cycles per micro-op ($CP\mu Op$).

In order to measure a representative value of cycles per micro-op, the same AMD Athlon system as was used to run the Windows workload was measured running the Berkeley Multimedia Workload on Mandrake Linux v7.0. This system was then modified to use kernel version 2.2.15 with Mikael Petterson's perfect counters patch v1.2 [Pett]. The AMD Athlon (as well as many other processors) implement hardware performance counters which can keep track of statistics

Application Name	Data Set	Context Interval
3D Flowerbox OpenGL Screen Saver (Windows 2000)	1280x1024x32bpp, (1:00)	23,653
RealPlayer v7.0 RealAudio Player	KAMU 64Kbps, stereo, G2 stream, (5:00)	40,396
Real Jukebox v1.0.0.488 MP3 Player	Santana - <i>Smooth</i> , 160 Kbps, stereo (4:54)	58,399
MediaPlayer GSM 06.10 (Windows 2000)	Speech by Al Gore, 8 kHz Mono, 16-bits (0:24)	297,641
K-Jofol 2000 MP3 Player v1.0	Santana - <i>Smooth</i> , 160 Kbps, stereo (4:54)	360,336
3D Pipes OpenGL Screen Saver (Windows 2000)	1280x1024x32bpp, (1:00)	567,080
Narrator Text to Speech (Windows 2000)	U.S. Declaration of Independence	594,438
MediaPlayer IMA ADPCM (Windows 2000)	Santana - <i>Smooth</i> , 160 Kbps, stereo (4:54)	708,037
WinDVD v2.0 DVD Player	(5:00) clip from <i>Amadeus</i>	921,510
PowerDVD v2.55 DVD Player	(5:00) clip from <i>Amadeus</i>	1,189,234
Ghostscript/Ghostview PostScript Previewer	Rosenblum and Ousterhaut's LFS paper (15 pages)	1,227,194
Dragon Naturally Speaking Preferred Speech Recognition	U.S. Declaration of Independence	2,560,537
Audio Catalyst v2.1 MP3 Encoder	Santana - <i>Smooth</i> , 44 kHz, stereo (4:54)	3,358,692
Audio Compositor MIDI Renderer	X-files theme song, Personal Copy v4.2 Sound Fonts	3,675,086
Irfanview v3.15 Image Viewer	Kodak's Iowa Corn jpeg image (2048x3072x24bpp)	3,821,284
Quake III Arena (Demo)	Internal demo #1, demo #2 (640x480)	4,284,671
DjVushop Document Compression	Scanned cover of March 2000 <i>IEEE Computer</i> journal	4,754,521
Avi2Mpg2 MPEG-2 Encoder	160 frames, 720x480 from <i>Amadeus</i>	5,339,432
POVray v3.1g Raytracer	Torus (internal demo scene), 800x600 Anti-Aliased	5,928,433
3D Maze OpenGL Screen Saver (Windows 2000)	1280x1024x32bpp, (1:00)	5,930,096
Arithmetic Mean		2,247,389
Geometric Mean		1,015,426

Table 2: **Average Multimedia Context Switch Intervals** - Measurements are given in 500 MHz clock cycles. Applications annotated with “(Windows 2000)” are packaged with the Windows 2000 operating system.

such as instructions executed, micro-ops executed, as well as many others [AMD]. (These hardware performance counters are not readily accessible under Windows.) The results of these measurements are shown in Table 3, which lists data both in terms of instructions as well as micro-ops.

In order to determine $Q_{context}$ for the RISC Alpha architecture, we divide the context switch intervals in Table 2 by a CPI equal to the average 1.06 CPμOp measurement from Table 3. As stated earlier, we use this parameter rather than x86 CPI because micro-ops are much more similar to RISC Alpha instructions than CISC x86 instructions. Note also that in a real system the interval between task switches is variable, not fixed; since we don't have the distribution of inter-interrupt times, we chose to use a constant interval. Alternately, we could have chosen some other distribution, such as exponential, normal or uniform.

1.1.2 Compaq Unix SPEC95

Table 1 lists context switch intervals for SPEC95 measured for Compaq Tru64 Unix v5.0 running on a DEC DS20 workstation (dual 21264 processors, each running at 500 MHz), with 2 GB of RAM, again running in a system with a single active task. For comparison, we review below the multiprogramming parameters of a variety of other workloads and systems previously studied in the literature. [Agar88] found that task switch times varied between 10,000 and 50,000 instruction references for both DEC Ultrix and VMS operating systems on a VAX 8200 processor (5 MHz). [Borg90] lists parameters

for Titan, an experimental RISC workstation with a clock speed of 22.2 MHz, for which $Q = 25,000$ clock cycles, with CPI typically being greater than one. [Clark83] measured context switch intervals between 11,000 and 23,000 instruction references on the DEC VAX 11/780 (5 MHz). Later measurements by the same author in [Clark88] on a VAX 8800 processor (22.2 MHz) found an average of 19,353 instructions between context switches. The traces used in [Smit87] were from a variety of architectures (Zilog Z8000, DEC VAX 11/780, CDC 6400, IBM 370, and Motorola 68000), and were simulated with a task switch interval of 15,000–20,000 memory references. [Koba86] examines the task switching behavior of IBM's MVS operating system (OS/VS2 MVS) on an Amdahl 470 V/8 system. Measurements obtained by a hardware monitor for a production time sharing option (TSO) workload had task intervals of approximately 7,600 instructions and 3,600 instructions for supervisor (system) state and problem (user) state respectively. A batch workload exhibited 8,700 and 14,000 for supervisor and problem state. Note that all of these other workloads were heavily multiprogrammed (and timeshared), and thus are not fully comparable. Of course, much computation is currently done on single user PCs and workstations, so a uniprogrammed system is probably the most representative of current technology.

1.2 Simulation Methodologies

Even though our study utilizes execution driven simulation, tracing full applications is still costly in terms of simulation

Name	User Time (cycles)	Instructions	Micro-Ops	Cycles/Instruction	Cycles/Micro-Op
ADPCM Encode	76,904,958	61,876,844	62,158,059	1.24	1.24
ADPCM Decode	64,344,819	53,313,197	53,594,784	1.21	1.20
DJVVU Encode	445,708,199	287,174,703	322,253,494	1.55	1.38
DJVVU Decode	356,305,000	244,918,278	266,984,846	1.45	1.33
Doom	1,472,798,700	793,162,073	1,181,637,518	1.86	1.25
Ghostscript	670,630,530	748,440,530	827,890,835	0.90	1.81
GSM Encode	188,190,001	225,158,845	248,092,796	0.84	0.76
GSM Decode	60,417,117	83,563,455	87,869,424	0.72	0.69
JPEG Encode	153,366,654	155,941,646	172,197,756	0.98	0.89
JPEG Decode	78,532,637	69,994,886	71,219,757	1.12	1.10
LAME	10,695,164,749	8,758,744,400	9,792,965,536	1.22	1.09
Mesa Gears	607,508,141	238,630,536	246,478,641	2.55	2.46
Mesa Morph3D	430,137,133	200,480,866	247,112,920	2.15	1.74
Mesa Reflect	1,940,806,991	1,692,447,264	2,090,552,059	1.15	0.93
MPEG-2 Encode DVD	14,037,454,521	14,005,949,265	16,702,192,536	1.00	0.84
MPEG-2 Encode 720P	37,759,708,362	37,445,342,666	44,698,434,032	1.01	0.84
MPEG-2 Encode 1080I	85,720,663,990	87,173,068,231	103,429,470,721	0.98	0.83
MPEG-2 Decode DVD	778,728,056	930,444,099	974,763,847	0.84	0.80
MPEG-2 Decode 720P	2,462,187,286	2,937,962,420	3,110,807,374	0.84	0.79
MPEG-2 Decode 1080I	5,043,716,605	5,759,111,626	6,014,852,989	0.88	0.84
mpg123	446,758,849	409,577,365	486,650,398	1.09	0.92
POVray3	72,427,881,123	5,137,007,093	6,809,522,382	1.45	1.09
Rasta	27,180,004	21,569,326	26,527,625	1.26	1.02
Rsynth	526,040,697	421,248,076	534,184,372	1.25	0.98
Timidity	1,132,954,552	1,363,791,599	1,522,840,735	0.83	0.74
Average				1.21	1.06

Table 3: **Measured CPI and CP μ Op for Berkeley Multimedia Workload** - 500 MHz AMD Athlon system, 128 MB RAM, running Mandrake Linux v7.0 with a modified kernel v2.2.15

time. Our cache simulations were run on a lab of twenty Alpha AXP 300 workstations at Berkeley, each with a single 300 MHz Alpha 21064a processor and 128 MB of RAM. Simulation time per memory reference is dependent on the type of cache being simulated, requiring 0.418 μ s (126 cycles), 1.770 μ s (531 cycles), and 1.262 μ s (379 cycles) on average per reference for instruction, data and unified cache simulations respectively. Instruction cache simulation is especially fast because references can be simulated at the basic block level of granularity rather than per instruction as is necessary for data references. Average simulation time per reference was between 0.114 μ s (35 cycles) and 2.112 μ s (634 cycles) (for comparison, [Gee93]'s simulation of SPEC92 on 40 MHz DECstation 5000/240 machines required 200 μ s (8000 cycles) to 400 μ s (16,000 cycles) per reference simulated for simulations). Total simulation time for our work, not including false starts, machine down time and other simulation problems was 24.4 days of CPU time for each multimedia run (once with and once without multiprogramming), and 147.2 days of CPU time for SPEC95 simulations, for a grand total of 196 days of CPU time.

Figure 1: **Average ATOM Cheetah Simulation Speeds**

1.3 Current Cache Configurations

As can be seen from Table 4, currently available processors are very similar in their cache design choices.

	AMD Athlon	DEC 21264A	HP PA-8500	Intel Pentium III	MIPS R12000	Motorola 7400 (G4)	Sun UltraSPARC III
Transistors ($\times 10^6$)	22	15.2	140	9.5	6.9	6.5	5.4
Process (μm)	0.25	0.25	0.25	0.18	0.25	0.20	0.25
Die Size (mm^2)	184	205	468.6	104.6	204	83	126
Clock (MHz)	800	750	440	733	300	500	480
L1 \$I\$ Size (KB)	64	64	512	16	32	32	16
L1 \$I\$ Associativity	2	2	4	4	2	8	2
L1 \$I\$ Line Size (bytes)	64	16	32/64	32	32	32	32
L1 \$D\$ Size (KB)	64	64	1024	16	32	32	16
L1 \$D\$ Associativity	2	2	4	4	2	8	1
L1 \$D\$ Line Size (bytes)	64	64	32/64	32	32	32	32

Table 4: Current Microprocessor Cache Configurations [Burd][Noer]

1.4 Measuring Cache Parameters

In order to choose the optimal line size for a given cache size, it is first necessary to know the parameters of the memory system to be used. In order to illustrate this, we employed HBench-OS v1.0 [Brow97], an extended version of the `lmbench` micro-benchmarking suite detailed in [McVoy96] (this work was apparently developed independently from the TLB and cache measurement techniques presented in [Saav95]). Each micro-benchmark measures the average time per iteration required to read or write a subset of elements belonging to an array of known size. The number of cache misses (and therefore the latency) is a function of the size of the array and the stride between consecutive addresses referenced. This effect can be seen in Figure 2 which graphs latency for various stride sizes for the 500 MHz AMD Athlon system utilized in our study. Each data set curve represents a stride size with the array size varying from 512 bytes up to 8 Mbytes, with a latency of 1 clock cycle taken to be “zero” - the time reported is only memory latency time, and does not include the instruction execution time. This is why all of the curves in Figure 2 are zero left of the L1 data cache capacity (64 Kbytes). Each horizontal plateau represents another level of the memory hierarchy (L1 at 64 Kbytes, L2 at 512 Kbytes for the AMD Athlon). Different strides are measured by the benchmark in order to determine the cache line size. The smallest stride that has a latency equal to that of main memory is the cache line size because the strides that are faster than memory must necessarily hit more than once per cache line.

1.5 Comparison with [Smit87]

In order to compare our line size analysis results with other workloads, we examined the metric of *ratio of miss ratios* [Smit87], which is the ratio of the miss ratio at a given cache size and line size to that for the same cache size but half the line size. The analysis of line size in [Smit87] employs a set of 27 traces from five different machine architectures. Table 5 compares the ratio of miss ratios metric for that workload against those of our multimedia workload. Graphs of the ratios of miss ratios for the Berkeley Multimedia Workload

Figure 2: 500 MHz Athlon System Memory Latency Profile - memory latency for various stride length memory accesses

are shown in Figures 3, 4, and 5. Cache behavior was quite similar between the two workloads except in the case of data caches. The multimedia workload benefitted more than the one studied in [Smit87] when going from 16B to 32B data cache line lengths, but the opposite held true for the other data cache line length comparisons (64B:32B and 128B:64B).

1.6 CCC Model

It is useful to examine the miss ratio spreads for the multimedia, SPEC92 and SPEC95 workloads in light of the three C’s model of cache miss behavior introduced in [Hill87] which categorizes misses into three classes: *conflict misses*, which are due to two different elements of the working set mapping to the same physical cache blocks (none for a fully associative cache), *capacity misses* which result from the cache capacity being too small to hold the working set (none for an infinitely large cache) and *compulsory misses* which are necessary when initially loading the cache (present for any cache configuration). It is only where conflict misses exist that associativity makes a difference, and there is a great variation among different applications in the contribution conflict misses make

32B:16B						
	Unified		Instruction		Data	
Size	Media	[Smit87]	Media	[Smit87]	Media	[Smit87]
1K	0.676	0.831	0.597	0.963	0.789	0.836
2K	0.679	0.731	0.607	0.737	0.670	0.787
4K	0.727	0.719	0.673	0.651	0.657	0.727
8K	0.654	0.645	0.622	0.598	0.627	0.667
16K	0.627	0.616	0.609	0.581	0.633	0.646
32K	0.632	0.601	0.559	0.581	0.615	0.646

64B:32B						
	Unified		Instruction		Data	
Size	Media	[Smit87]	Media	[Smit87]	Media	[Smit87]
2K	0.819	0.787	0.698	0.701	0.889	0.853
4K	0.785	0.746	0.831	0.660	0.740	0.770
8K	0.756	0.661	0.717	0.667	0.706	0.723
16K	0.676	0.633	0.633	0.624	0.662	0.672
32K	0.656	0.601	0.696	0.624	0.686	0.662

128B:64B						
	Unified		Instruction		Data	
Size	Media	[Smit87]	Media	[Smit87]	Media	[Smit87]
4K	0.938	1.809	0.733	0.831	0.925	0.835
8K	0.937	0.753	0.776	0.690	0.865	0.817
16K	0.868	0.685	0.655	0.657	0.841	0.747
32K	0.739	0.660	0.854	0.634	0.750	0.697

Table 5: **Comparison to [Smit87]** - Ratio of miss ratio to that of line half as long. Results are shown only for those cache configurations that are common to both [Smit87] and this study.

Figure 3: **Ratio of Miss Ratios: Unified Cache** - Berkeley Multimedia Workload

to overall miss ratio. Figure 6 graphs the variation of the three types of misses for MPEG decoding at DVD resolution, and the SPECint95 147.vortex application. What is clear is that these two applications have widely different proportions of conflict, capacity and compulsory miss ratios for any given cache capacity. Although individual traces exhibit widely differing contributions from the three miss ratio factors, traces

(a) Instruction Cache

Figure 4: **Ratio of Miss Ratios: Instruction Cache** - Berkeley Multimedia Workload

(a) Data Cache

Figure 5: **Ratio of Miss Ratios: Data Cache** - Berkeley Multimedia Workload

on average follow the heuristics of [Hill89] (see Figure 7).

1.7 Workloads

1.7.1 SPEC92 [Gee93]

In the past decade, SPEC (Standard Performance Evaluation Corporation) benchmarks have played an extremely important role in both hardware and compiler design. SPEC CPU benchmarks are taken to be generally representative of traditional workstation applications, with the integer component reflecting system or commercial applications, and the floating point component representing numeric and scientific applications. The SPEC92 benchmark consists of six integer-

Figure 6: Miss Ratio Factors

Figure 7: Average Miss Ratio Factors

intensive, and fourteen floating-point-intensive programs (see Table 6). All of the component benchmarks are written in either C (integer) or Fortran (floating point).

It is important to note that although the SPEC component applications are based on real world counterparts, many of them were modified by SPEC from their original form to reduce the influence of I/O and the operating system on performance. In some instances, this means that test data is read from and written to memory, rather than disk (as would be the case in the original form of the application). Clearly this

could potentially affect memory behavior compared to real world applications.

[Gee93] analyzed the cache behavior of the SPEC92 benchmark suite running on DECstations with MIPS R2000 or R3000 processors and version 4.1 of the DEC Ultrix operating system. Version 2.0 of the C compiler and version 2.1 of the FORTRAN compiler were used with the optimization level according to the SPEC Makefiles. Because the SPEC benchmarks are typically run in a uniprogrammed environment, no cache flushing or other method was used to simulate

multiprogramming. [Gee93] also found that for the SPEC92 benchmark suite, system time is insignificant compared to user time, and so operating system memory behavior is largely unimportant for this suite of applications.

Integer	Description
008.espresso	Circuit theory
022.li	Lisp interpreter
023.eqntott	Logic design
026.compress	Lempel-Ziv coding
072.sc	Spread sheet
085.gcc	GNU C Compiler
Floating Point	Description
013.spice2g6	Circuit design
015.doduc	Monte-Carlo simulation
034.mdljdp2	Quantum chemistry
039.wave5	Maxwell's equations
047.tomcatv	Coordinate translation
048.ora	Optics ray-tracing
052.alvinn	Robotics
056.ear	Model of the human ear
077.mdljsp2	Quantum chemistry
078.swm256	Shallow water model
089.su2cor	Quantum physics
090.hydr2d	Astrophysics
093.nasa7	NASA kernels
094.fpppp	Quantum chemistry

Table 6: SPEC92 Benchmark Suite

1.7.2 SPEC95

SPEC95 is an upgraded version of the SPEC92 benchmark suite. It consists of eight integer intensive and ten floating-point intensive applications. Several of the programs from SPEC92 are found in SPEC95. 126.gcc is the same as 085.gcc except that a newer version of gcc is used. 129.compress is the same as 026.compress. 130.li is the same as 022.li. 101.tomcatv is the same as 047.tomcatv. 102.swim is the same as 078.swm256 except that it uses a 1024x1024 grid. 103.su2cor is the same as 089.su2cor. 104.hydro2d is the same as 090.hydro2d. 145.fpppp is the same as 094.fpppp. 146.wave5 is the same as 039.wave5. In general, the applications were designed to have larger code size and greater memory activity than those of SPEC92.

1.7.3 Multiprogramming Workload (Mult) [Borg90]

[Borg90] generated miss ratios for very long address traces (up to 12 billion memory references in length) on the Titan RISC architecture in order to evaluate the performance of a variety of cache designs. Three individual traces were used in addition to another which was a multiprogrammed workload consisting of several jobs. Our comparison includes their miss ratio results for their 7.6 billion reference (68.5% instruction, 30.6% load, 15.4% store) multiprogramming workload

(referred to as “Mult” by the authors of [Borg90]). It consists of:

- **make** run compiling (from C) portions of the **magic** source code
- **grr** routing the DECstation 3100 printed circuit board
- **magic** (VLSI editor) design rule checking the MultiTitan CPU chip
- **tree** (compiled Scheme) builds tree data structure and searches for largest element
- **make** run that calls **xld** (linker/loader) on the **magic** code
- **tcsh** script of infinite loop of interactive commands (**cp**, **cat**, **ex**, **rm**, **ps**, **-aux**, **ls -l /***)

1.7.4 Design Target Miss Ratios (DTMR) [Smit85]

[Smit85] introduced the concept of *design target miss ratios* (DTMRs), intended to represent typical levels of performance across a wide class of workloads and machines, to be used for hardware design. The DTMRs were synthesized from real (hardware monitor) measurements that existed in the literature and from trace driven simulations using a large number of traces taken from several architectures, originally coded in several different languages.

1.7.5 VAX 11/780 [Clark83], VAX 8800 [Clark88]

Two studies done at Digital Equipment Corporation (DEC) supply miss ratios for a time-shared engineering workload taken with a hardware monitor on VAX 11/780 and VAX 8800 machines [Clark83], [Clark88]. The 11/780 has an 8-KB, write through, unified cache with an 8-byte block size and associativity of two. The 8800 has a 64-KB, write-through, direct mapped, unified cache with a 64-byte block size. On the VAX 11/780 it is possible to disable half of the two-way associative cache through special control bits; a technique which allowed for measurement of a 4-KB, direct mapped, unified cache configuration as well.

1.7.6 Agarwall Mul3 [Agar88]

In [Agar88] an analysis of the effect of operating system references and multiprogramming was presented for a workload of eleven application programs (30 traces in all). The platform used to gather the traces was a VAX 11/780 running either the Ultrix or VMS operating system. All of the traces were gathered through the ATUM scheme of microcode modification, and were roughly 400,000 references long (approximately half a second of execution time). A technique termed *trace sampling* was used to concatenate smaller traces to better simulate the full trace length of a running program. We utilize their three way multiprogrammed workload for comparison.

Integer	Description	Instruction	Load	Store
099.go	AI plays "Go"	32,465,547,697	8,168,233,239	2,409,111,305
124.m88ksim	Motorola 88K simulator	73,820,020,885	13,680,509,709	7,402,458,586
126.gcc	GCC, builds SPARC code	209,948,942	49,135,874	24,970,849
129.compress	Lempel-Ziv coding	60,850,564,695	11,722,190,689	4,458,902,946
130.li	Lisp interpreter	65,730,724,167	16,730,950,984	9,385,736,837
132.jpeg	DCT image compression	34,116,845,423	6,023,880,310	1,870,937,674
134.perl	Anagrams/primes in Perl	29,378,239,506	6,868,627,321	4,173,646,937
147.vortex	Database	87,529,422,776	20,621,987,100	11,989,029,595
Floating Point	Description			
101.tomcatv	Mesh generation	31,932,871,503	6,481,320,753	3,324,212,603
102.swim	Shallow water model	32,575,361,567	5,702,422,493	2,773,604,570
103.su2cor	Monte Carlo simulation	39,599,980,228	7,061,086,677	3,214,956,672
104.hydro2d	Hydrodynamical Eqns	48,836,577,610	10,074,625,107	3,468,630,651
107.mgrid	Multigrid solver in 3D field	72,429,596,913	18,187,911,190	3,911,599,079
110.applu	Differential equations	33,716,422,315	7,722,618,047	3,267,428,916
125.turb3d	Turbulence in cube	104,351,000,000	15,919,035,949	13,253,927,515
141.apsi	Temp, Wind, Potential	33,155,694,301	6,254,642,763	3,445,287,831
145.fpppp	Quantum chemistry	147,327,000,000	43,911,029,435	11,283,084,727
146.wave5	Plasma physics	34,209,423,864	6,831,165,556	4,455,048,459
Total		962,235,242,392	212,011,373,196	94,112,575,752
Arithmetic Mean		53,457,513,466	11,778,409,622	5,228,476,431

Table 7: SPEC95 Benchmark Suite Characteristics

1.7.7 Amdahl 470 [Smit82]

In [Smit82], hardware monitor measurements taken at Amdahl Corporation on Amdahl 470V machines are presented. A standard internal benchmark was run containing supervisor, commercial and scientific code. Supervisor state miss ratios were found to be much higher than problem state miss ratios.

References

- [Agar88] Anant Agarwal, John Hennessy, Mark Horowitz, "Cache Performance of Operating System and Multiprogramming Workloads," *ACM Trans. on Computer Systems*, Vol. 6, No. 4, November 1988, pp. 393-431
- [AMD] Advanced Micro Devices, Inc., "AMD Athlon Processor x86 Code Optimization Guide," Publication #22007G/0, April 2000, <http://www.amd.com/products/cpg/athlon/techdocs/pdf/22007.pdf>, retrieved April 24, 2000
- [Borg90] Anita Borg, R. E. Kessler, David W. Wall, "Generation and Analysis of Very Long Address Traces," *Proc. 17th Int'l Symp. on Computer Architecture*, Seattle, Washington, May 28-31, 1990, pp. 270-279
- [Brow97] A. Brown, and M. Seltzer. "Operating System Benchmarking in the Wake of Lmbench: A Case Study of the Performance of NetBSD on the Intel x86 Architecture." *Proc. 1997 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, Seattle, Washington, June 15-18, 1997, pp. 214-224
- [Burd] Tom Burd, "CPU Info Center: General Processor Info," <http://bwrc.eecs.berkeley.edu/CIC/summary/local/summary.pdf>, retrieved April 24, 2000
- [Clark83] D. W. Clark, "Cache Performance in the VAX-11/780," *ACM Trans. on Computing Systems*, Vol. 1, No. 1, February 1983, pp. 24-37
- [Clark88] D. W. Clark, P. J. Bannon, J. B. Keller, "Measuring VAX 8800 Performance with a Histogram Hardware Monitor," *Proc. 15th Int'l Symp. on Computer Architecture*, Honolulu, Hawaii, May 30-June 2, 1988, pp. 176-185
- [Gee93] Jeffrey D. Gee, Mark D. Hill, Dionisios N. Pnevmatikatos, Alan Jay Smith, "Cache Performance of the SPEC92 Benchmark Suite," *IEEE Micro*, Vol. 13, No. 4, August 1993, pp. 17-27
- [Hill87] Mark D. Hill, "Aspects of Cache Memory and Instruction Buffer Performance," Ph.D. Thesis, University of California at Berkeley, Computer Science Division, *Technical Report UCB/CSD 87/381* (November, 1987)
- [Hill89] Mark D. Hill, Alan Jay Smith, "Evaluating Associativity in CPU Caches," *IEEE Trans. on Computers*, Vol. 38, No. 12, December 1989, pp. 1612-1630
- [Koba86] Makoto Kobayashi, "An Empirical Study of Task Switching Locality in MVS," *IEEE Transactions on Computers*, Vol. C-35, No. 8, August 1986, pp. 720-731
- [McVoy96] Larry McVoy, Carl Staelin, "Imbench: Portable tools for performance analysis," *Proc. of Winter 1996 USENIX Conference*, San Diego, California, January 22-26, 1996, pg 279-294
- [Noer] Kim Noer, "Heat Dissipation Per Square Millimeter Die Size Specifications," <http://home.worldonline.dk/~noer/>, retrieved April 24, 2000
- [Pett] Mikael Pettersson, "Perfect Counters Home Page," <http://www.csd.uu.se/~mikep/linux/perfctr>, retrieved April 24, 2000
- [Saav95] Rafael H. Saavedra, Alan Jay Smith, "Measuring Cache and TLB Performance and Their Effect on Benchmark Runtimes," *IEEE Trans. on Computers*, Vol. 44 No. 10, October 1995, pp. 1223-1235
- [Smit82] Alan Jay Smith, "Cache Memories," *ACM Computing Surveys*, Vol. 14, No. 3, September 1982, pp. 473-530
- [Smit85] Alan Jay Smith, "Cache Evaluation and the Impact of Workload Choice," *Proc. 12th Int'l Symp. on Computer Architecture*, Boston, Massachusetts, June 17-19, 1985, pp. 64-73
- [Smit87] Alan Jay Smith, "Line (Block) Size Choice for CPU Cache Memories," *IEEE Trans. on Computers*, Vol. C-36, No. 9, September 1987, pp. 1063-1075

Berkeley Multimedia Workload

Multimedia						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.05622	0.05594	0.05259	0.05103	0.05180	
2K	0.02969	0.02815	0.02561	0.02622	0.02664	
4K	0.01850	0.01506	0.01316	0.01140	0.01046	
8K	0.01128	0.00861	0.00668	0.00598	0.00544	
16K	0.00637	0.00390	0.00298	0.00280	0.00285	
32K	0.00377	0.00151	0.00119	0.00093	0.00075	
64K	0.00274	0.00089	0.00074	0.00069	0.00068	
128K	0.00163	0.00069	0.00067	0.00067	0.00067	
256K	0.00128	0.00068	0.00067	0.00067	0.00067	
512K	0.00077	0.00067	0.00067	0.00067	0.00067	
1M	0.00068	0.00067	0.00067	0.00067	0.00067	
2M	0.00067	0.00067	0.00067	0.00067	0.00067	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.12894	0.09619	0.08640	0.08325	0.08019	
2K	0.09326	0.07223	0.06574	0.06360	0.06244	
4K	0.06587	0.05530	0.05154	0.05108	0.05102	
8K	0.04903	0.04176	0.04092	0.04068	0.04067	
16K	0.03762	0.03131	0.02995	0.02996	0.02994	
32K	0.02924	0.02527	0.02432	0.02381	0.02365	
64K	0.02556	0.02194	0.02131	0.02112	0.02099	
128K	0.02154	0.02074	0.02046	0.02047	0.02048	
256K	0.02079	0.02011	0.02002	0.02000	0.01999	
512K	0.01932	0.01966	0.01982	0.01981	0.01981	
1M	0.01756	0.01744	0.01736	0.01737	0.01737	
2M	0.01746	0.01734	0.01732	0.01732	0.01732	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.11725	0.09241	0.08662	0.08609	0.08794	
2K	0.07795	0.05897	0.05644	0.05513	0.05451	
4K	0.05180	0.03546	0.03112	0.02913	0.02860	
8K	0.03314	0.02224	0.01923	0.01851	0.01780	
16K	0.02215	0.01388	0.01197	0.01148	0.01141	
32K	0.01613	0.00954	0.00841	0.00826	0.00836	
64K	0.01254	0.00677	0.00590	0.00562	0.00546	
128K	0.00776	0.00549	0.00509	0.00502	0.00501	
256K	0.00652	0.00499	0.00490	0.00490	0.00489	
512K	0.00524	0.00484	0.00482	0.00481	0.00481	
1M	0.00471	0.00432	0.00430	0.00430	0.00430	
2M	0.00463	0.00430	0.00429	0.00429	0.00429	

Document						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.08120	0.08365	0.08482	0.08198	0.08455	
2K	0.05397	0.04785	0.04111	0.04335	0.04540	
4K	0.03315	0.02371	0.01935	0.01118	0.00580	
8K	0.01968	0.01183	0.00607	0.00372	0.00083	
16K	0.00818	0.00361	0.00151	0.00049	0.00047	
32K	0.00592	0.00044	0.00038	0.00033	0.00034	
64K	0.00572	0.00028	0.00023	0.00021	0.00020	
128K	0.00215	0.00019	0.00018	0.00018	0.00018	
256K	0.00193	0.00018	0.00018	0.00018	0.00018	
512K	0.00018	0.00018	0.00018	0.00018	0.00018	
1M	0.00018	0.00018	0.00018	0.00018	0.00018	
2M	0.00018	0.00018	0.00018	0.00018	0.00018	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.16031	0.11569	0.09283	0.08694	0.08255	
2K	0.10863	0.07608	0.05455	0.04709	0.04573	
4K	0.06906	0.05406	0.04080	0.03988	0.03948	
8K	0.04902	0.03662	0.03549	0.03413	0.03393	
16K	0.03782	0.03018	0.02937	0.02936	0.02900	
32K	0.03260	0.02730	0.02625	0.02609	0.02608	
64K	0.03016	0.02526	0.02485	0.02475	0.02472	
128K	0.02514	0.02471	0.02453	0.02452	0.02453	
256K	0.02464	0.02435	0.02434	0.02434	0.02434	
512K	0.01975	0.02297	0.02428	0.02429	0.02427	
1M	0.01278	0.01266	0.01265	0.01265	0.01265	
2M	0.01264	0.01263	0.01263	0.01263	0.01263	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.14715	0.13057	0.12367	0.12248	0.12521	
2K	0.10053	0.08617	0.07801	0.07543	0.07352	
4K	0.06529	0.04708	0.03836	0.03235	0.03137	
8K	0.04125	0.02437	0.01541	0.01244	0.00904	
16K	0.02037	0.01164	0.00811	0.00714	0.00699	
32K	0.01496	0.00656	0.00611	0.00592	0.00588	
64K	0.01324	0.00570	0.00550	0.00544	0.00543	
128K	0.00762	0.00536	0.00525	0.00522	0.00521	
256K	0.00723	0.00519	0.00516	0.00516	0.00516	
512K	0.00476	0.00488	0.00515	0.00515	0.00515	
1M	0.00328	0.00273	0.00273	0.00273	0.00273	
2M	0.00301	0.00272	0.00272	0.00272	0.00272	

Audio						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.01575	0.01348	0.01284	0.01274	0.01277	
2K	0.00770	0.00722	0.00758	0.00776	0.00785	
4K	0.00240	0.00212	0.00223	0.00221	0.00216	
8K	0.00093	0.00074	0.00070	0.00072	0.00071	
16K	0.00060	0.00046	0.00032	0.00028	0.00027	
32K	0.00037	0.00028	0.00028	0.00026	0.00026	
64K	0.00034	0.00026	0.00026	0.00026	0.00026	
128K	0.00026	0.00026	0.00026	0.00026	0.00026	
256K	0.00026	0.00026	0.00026	0.00026	0.00026	
512K	0.00026	0.00026	0.00026	0.00026	0.00026	
1M	0.00026	0.00026	0.00026	0.00026	0.00026	
2M	0.00026	0.00026	0.00026	0.00026	0.00026	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.16773	0.12483	0.11573	0.11242	0.11204	
2K	0.12176	0.10471	0.10424	0.10325	0.10329	
4K	0.08938	0.07868	0.07853	0.07833	0.07821	
8K	0.05846	0.05305	0.05281	0.05375	0.05397	
16K	0.03739	0.02686	0.02336	0.02407	0.02496	
32K	0.01788	0.01419	0.01347	0.01299	0.01309	
64K	0.01026	0.00698	0.00658	0.00617	0.00564	
128K	0.00653	0.00530	0.00518	0.00517	0.00516	
256K	0.00619	0.00516	0.00515	0.00515	0.00515	
512K	0.00532	0.00514	0.00514	0.00514	0.00514	
1M	0.00515	0.00514	0.00514	0.00514	0.00514	
2M	0.00515	0.00514	0.00514	0.00514	0.00514	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.08687	0.05124	0.04419	0.04235	0.04191	
2K	0.05540	0.03974	0.03559	0.03488	0.03452	
4K	0.04166	0.02996	0.02793	0.02746	0.02730	
8K	0.02723	0.02010	0.01998	0.02089	0.02127	
16K	0.01634	0.01014	0.00821	0.00832	0.00861	
32K	0.00896	0.00600	0.00427	0.00392	0.00390	
64K	0.00591	0.00232	0.00214	0.00200	0.00208	
128K	0.00454	0.00141	0.00127	0.00126	0.00125	
256K	0.00154	0.00127	0.00124	0.00124	0.00124	
512K	0.00133	0.00124	0.00124	0.00124	0.00124	
1M	0.00127	0.00124	0.00124	0.00124	0.00124	
2M	0.00127	0.00124	0.00124	0.00124	0.00124	

Video						
Instruction Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.03242	0.03540	0.02537	0.02841	0.03139	
2K	0.01254	0.01063	0.01074	0.01131	0.01136	
4K	0.00594	0.00432	0.00417	0.00416	0.00416	
8K	0.00337	0.00247	0.00259	0.00276	0.00293	
16K	0.00241	0.00098	0.00042	0.00033	0.00032	
32K	0.00133	0.00048	0.00032	0.00032	0.00032	
64K	0.00032	0.00032	0.00032	0.00032	0.00032	
128K	0.00032	0.00032	0.00032	0.00032	0.00032	
256K	0.00032	0.00032	0.00032	0.00032	0.00032	
512K	0.00032	0.00032	0.00032	0.00032	0.00032	
1M	0.00032	0.00032	0.00032	0.00032	0.00032	
2M	0.00032	0.00032	0.00032	0.00032	0.00032	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.06711	0.04604	0.03986	0.03827	0.03049	
2K	0.03887	0.02450	0.02206	0.02250	0.01891	
4K	0.02085	0.01171	0.01065	0.01066	0.01077	
8K	0.01514	0.00795	0.00745	0.00727	0.00724	
16K	0.00979	0.00707	0.00666	0.00665	0.00665	
32K	0.00790	0.00669	0.00656	0.00658	0.00662	
64K	0.00675	0.00632	0.00627	0.00631	0.00640	
128K	0.00617	0.00584	0.00584	0.00584	0.00585	
256K	0.00597	0.00581	0.00580	0.00580	0.00580	
512K	0.00589	0.00580	0.00580	0.00580	0.00580	
1M	0.00585	0.00580	0.00580	0.00580	0.00580	
2M	0.00583	0.00580	0.00580	0.00580	0.00580	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.08038	0.06239	0.05470	0.05439	0.05855	
2K	0.04081	0.02330	0.01924	0.01803	0.01755	
4K	0.02232	0.01073	0.00891	0.00860	0.00778	
8K	0.01340	0.00585	0.00530	0.00536	0.00543	
16K	0.00840	0.00326	0.00229	0.00205	0.00192	
32K	0.00556	0.00226	0.00147	0.00140	0.00140	
64K	0.00232	0.00177	0.00136	0.00135	0.00136	
128K	0.00199	0.00128	0.00127	0.00127	0.00128	
256K	0.00165	0.00127	0.00127	0.00127	0.00127	
512K	0.00172	0.00127	0.00127	0.00127	0.00127	
1M	0.00161	0.00127	0.00127	0.00127	0.00127	
2M	0.00159	0.00127	0.00127	0.00127	0.00127	

Berkeley Multimedia Workload

Multimedia						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.03410	0.03361	0.03194	0.03103	0.03091	
2K	0.01865	0.01720	0.01594	0.01600	0.01618	
4K	0.01155	0.00958	0.00818	0.00751	0.00704	
8K	0.00718	0.00575	0.00433	0.00410	0.00338	
16K	0.00388	0.00250	0.00192	0.00169	0.00173	
32K	0.00229	0.00092	0.00071	0.00054	0.00042	
64K	0.00167	0.00051	0.00041	0.00038	0.00037	
128K	0.00099	0.00038	0.00037	0.00036	0.00036	
256K	0.00080	0.00037	0.00036	0.00036	0.00036	
512K	0.00044	0.00036	0.00036	0.00036	0.00036	
1M	0.00037	0.00036	0.00036	0.00036	0.00036	
2M	0.00036	0.00036	0.00036	0.00036	0.00036	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.11991	0.08056	0.07028	0.06593	0.06326	
2K	0.07919	0.05499	0.04668	0.04335	0.04183	
4K	0.05145	0.03852	0.03409	0.03355	0.03351	
8K	0.03549	0.02718	0.02601	0.02563	0.02552	
16K	0.02560	0.01985	0.01896	0.01893	0.01896	
32K	0.01916	0.01572	0.01509	0.01472	0.01453	
64K	0.01615	0.01328	0.01270	0.01252	0.01243	
128K	0.01300	0.01225	0.01199	0.01195	0.01196	
256K	0.01231	0.01168	0.01163	0.01163	0.01163	
512K	0.01084	0.01126	0.01145	0.01144	0.01144	
1M	0.00916	0.00906	0.00899	0.00899	0.00900	
2M	0.00908	0.00898	0.00897	0.00897	0.00897	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.09476	0.06649	0.06108	0.05964	0.05945	
2K	0.06216	0.04222	0.03956	0.03806	0.03703	
4K	0.03947	0.02537	0.02149	0.02055	0.02079	
8K	0.02497	0.01545	0.01305	0.01240	0.01164	
16K	0.01614	0.00915	0.00779	0.00720	0.00715	
32K	0.01160	0.00617	0.00540	0.00525	0.00529	
64K	0.00901	0.00431	0.00368	0.00351	0.00340	
128K	0.00498	0.00339	0.00299	0.00291	0.00290	
256K	0.00412	0.00290	0.00282	0.00282	0.00282	
512K	0.00313	0.00275	0.00276	0.00275	0.00275	
1M	0.00265	0.00226	0.00224	0.00224	0.00224	
2M	0.00251	0.00224	0.00223	0.00223	0.00223	
Document						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.05230	0.05437	0.05363	0.05234	0.05268	
2K	0.03529	0.03171	0.02936	0.02934	0.02980	
4K	0.02202	0.01694	0.01310	0.01041	0.00822	
8K	0.01392	0.00963	0.00454	0.00404	0.00055	
16K	0.00535	0.00285	0.00133	0.00032	0.00031	
32K	0.00381	0.00027	0.00023	0.00020	0.00020	
64K	0.00367	0.00017	0.00014	0.00013	0.00013	
128K	0.00154	0.00011	0.00011	0.00010	0.00010	
256K	0.00141	0.00010	0.00010	0.00010	0.00010	
512K	0.00010	0.00010	0.00010	0.00010	0.00010	
1M	0.00010	0.00010	0.00010	0.00010	0.00010	
2M	0.00010	0.00010	0.00010	0.00010	0.00010	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.15503	0.11245	0.09384	0.08770	0.08201	
2K	0.10071	0.07014	0.04815	0.03775	0.03546	
4K	0.06302	0.04526	0.03124	0.02959	0.02906	
8K	0.04174	0.02793	0.02672	0.02524	0.02495	
16K	0.03057	0.02228	0.02173	0.02171	0.02144	
32K	0.02600	0.02042	0.01979	0.01968	0.01967	
64K	0.02393	0.01921	0.01897	0.01891	0.01890	
128K	0.01928	0.01886	0.01876	0.01875	0.01876	
256K	0.01881	0.01859	0.01858	0.01859	0.01860	
512K	0.01385	0.01718	0.01853	0.01854	0.01852	
1M	0.00678	0.00670	0.00669	0.00669	0.00669	
2M	0.00667	0.00666	0.00666	0.00666	0.00666	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.11834	0.09634	0.08842	0.08750	0.08785	
2K	0.07910	0.06591	0.05973	0.05945	0.05850	
4K	0.05037	0.03890	0.03004	0.02793	0.02974	
8K	0.03278	0.02060	0.01307	0.01075	0.00691	
16K	0.01655	0.00967	0.00691	0.00522	0.00512	
32K	0.01196	0.00544	0.00473	0.00442	0.00439	
64K	0.01053	0.00484	0.00416	0.00412	0.00412	
128K	0.00619	0.00462	0.00400	0.00398	0.00397	
256K	0.00589	0.00394	0.00393	0.00393	0.00393	
512K	0.00375	0.00364	0.00391	0.00391	0.00391	
1M	0.00226	0.00145	0.00145	0.00145	0.00145	
2M	0.00171	0.00144	0.00144	0.00144	0.00144	

Block Size: 32 bytes

Audio						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.00912	0.00750	0.00715	0.00711	0.00713	
2K	0.00480	0.00427	0.00425	0.00417	0.00421	
4K	0.00142	0.00122	0.00120	0.00120	0.00117	
8K	0.00054	0.00043	0.00040	0.00042	0.00043	
16K	0.00034	0.00026	0.00019	0.00016	0.00015	
32K	0.00021	0.00015	0.00015	0.00014	0.00014	
64K	0.00019	0.00014	0.00014	0.00014	0.00014	
128K	0.00014	0.00014	0.00014	0.00014	0.00014	
256K	0.00014	0.00014	0.00014	0.00014	0.00014	
512K	0.00014	0.00014	0.00014	0.00014	0.00014	
1M	0.00014	0.00014	0.00014	0.00014	0.00014	
2M	0.00014	0.00014	0.00014	0.00014	0.00014	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.14798	0.08724	0.07229	0.06688	0.06588	
2K	0.09297	0.06738	0.06094	0.05816	0.05796	
4K	0.06065	0.04592	0.04454	0.04481	0.04490	
8K	0.03583	0.03077	0.03008	0.03070	0.03092	
16K	0.02207	0.01496	0.01291	0.01308	0.01339	
32K	0.01047	0.00758	0.00706	0.00677	0.00682	
64K	0.00597	0.00373	0.00346	0.00326	0.00297	
128K	0.00387	0.00275	0.00267	0.00267	0.00266	
256K	0.00365	0.00266	0.00265	0.00265	0.00265	
512K	0.00283	0.00265	0.00265	0.00265	0.00265	
1M	0.00266	0.00265	0.00265	0.00265	0.00265	
2M	0.00265	0.00265	0.00265	0.00265	0.00265	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.07608	0.03967	0.03139	0.03005	0.02811	
2K	0.04449	0.02702	0.02189	0.02031	0.01973	
4K	0.03148	0.01835	0.01611	0.01567	0.01544	
8K	0.01891	0.01192	0.01153	0.01193	0.01208	
16K	0.01034	0.00589	0.00479	0.00480	0.00486	
32K	0.00579	0.00338	0.00228	0.00208	0.00206	
64K	0.00370	0.00128	0.00116	0.00110	0.00116	
128K	0.00284	0.00075	0.00066	0.00065	0.00065	
256K	0.00092	0.00066	0.00065	0.00065	0.00065	
512K	0.00072	0.00065	0.00064	0.00064	0.00064	
1M	0.00066	0.00065	0.00064	0.00064	0.00064	
2M	0.00066	0.00064	0.00064	0.00064	0.00064	
Video						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.01935	0.02141	0.01824	0.02011	0.01989	
2K	0.00796	0.00594	0.00597	0.00618	0.00620	
4K	0.00355	0.00257	0.00235	0.00234	0.00235	
8K	0.00204	0.00155	0.00167	0.00180	0.00191	
16K	0.00149	0.00060	0.00028	0.00018	0.00018	
32K	0.00079	0.00029	0.00018	0.00018	0.00018	
64K	0.00018	0.00018	0.00018	0.00018	0.00018	
128K	0.00018	0.00018	0.00018	0.00018	0.00018	
256K	0.00018	0.00018	0.00018	0.00018	0.00018	
512K	0.00018	0.00018	0.00018	0.00018	0.00018	
1M	0.00018	0.00018	0.00018	0.00018	0.00018	
2M	0.00018	0.00018	0.00018	0.00018	0.00018	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.07605	0.04808	0.04069	0.03794	0.03307	
2K	0.04510	0.02379	0.02065	0.02010	0.01726	
4K	0.02389	0.01115	0.00982	0.00998	0.01019	
8K	0.01594	0.00620	0.00540	0.00513	0.00504	
16K	0.00816	0.00479	0.00438	0.00436	0.00435	
32K	0.00573	0.00423	0.00407	0.00389	0.00394	
64K	0.00446	0.00381	0.00368	0.00362	0.00372	
128K	0.00365	0.00325	0.00326	0.00316	0.00317	
256K	0.00337	0.00313	0.00313	0.00313	0.00313	
512K	0.00325	0.00313	0.00313	0.00313	0.00313	
1M	0.00319	0.00313	0.00313	0.00313	0.00313	
2M	0.00316	0.00313	0.00313	0.00313	0.00313	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.06950	0.04756	0.04346	0.04166	0.04210	
2K	0.03824	0.01884	0.01507	0.01358	0.01322	
4K	0.02034	0.00861	0.00660	0.00653	0.00612	
8K	0.01157	0.00424	0.00364	0.00356	0.00358	
16K	0.00663	0.00240	0.00190	0.00183	0.00180	
32K	0.00409	0.00155	0.00096	0.00082	0.00082	
64K	0.00174	0.00110	0.00080	0.00077	0.00078	
128K	0.00136	0.00072	0.00071	0.00069	0.00069	
256K	0.00108	0.00069	0.00069	0.00069	0.00069	
512K	0.00100	0.00069	0.00069	0.00069	0.00069	
1M	0.00096	0.00069	0.00069	0.00069	0.00069	
2M	0.00094	0.00069	0.00069	0.00069	0.00069	

Berkeley Multimedia Workload

Multimedia						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.02357	0.02324	0.02162	0.02228		
2K	0.01324	0.01212	0.01162	0.01165	0.01131	
4K	0.00820	0.00699	0.00593	0.00562	0.00585	
8K	0.00512	0.00398	0.00308	0.00315	0.00242	
16K	0.00275	0.00168	0.00141	0.00110	0.00110	
32K	0.00160	0.00064	0.00048	0.00036	0.00029	
64K	0.00118	0.00032	0.00024	0.00022	0.00022	
128K	0.00071	0.00022	0.00021	0.00021	0.00021	
256K	0.00059	0.00021	0.00021	0.00021	0.00021	
512K	0.00027	0.00021	0.00021	0.00021	0.00021	
1M	0.00021	0.00021	0.00021	0.00021	0.00021	
2M	0.00021	0.00021	0.00021	0.00021	0.00021	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.13961	0.08760	0.07270	0.06831		
2K	0.08974	0.05433	0.04239	0.03895	0.03721	
4K	0.05364	0.03298	0.02676	0.02508	0.02481	
8K	0.03259	0.02072	0.01862	0.01810	0.01800	
16K	0.02088	0.01394	0.01287	0.01267	0.01254	
32K	0.01471	0.01070	0.01018	0.01008	0.00997	
64K	0.01174	0.00889	0.00840	0.00822	0.00811	
128K	0.00876	0.00792	0.00769	0.00762	0.00762	
256K	0.00806	0.00738	0.00735	0.00735	0.00736	
512K	0.00653	0.00696	0.00718	0.00717	0.00717	
1M	0.00487	0.00477	0.00473	0.00473	0.00473	
2M	0.00480	0.00472	0.00470	0.00470	0.00470	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.09301	0.05838	0.05116	0.04820		
2K	0.06026	0.03662	0.03179	0.03018	0.03035	
4K	0.03690	0.02125	0.01744	0.01667	0.01631	
8K	0.02186	0.01204	0.01012	0.00969	0.00880	
16K	0.01295	0.00672	0.00560	0.00492	0.00484	
32K	0.00878	0.00427	0.00364	0.00350	0.00347	
64K	0.00643	0.00294	0.00253	0.00242	0.00239	
128K	0.00361	0.00226	0.00192	0.00183	0.00181	
256K	0.00291	0.00182	0.00176	0.00175	0.00175	
512K	0.00202	0.00168	0.00170	0.00169	0.00169	
1M	0.00156	0.00119	0.00118	0.00118	0.00118	
2M	0.00144	0.00118	0.00117	0.00117	0.00117	
Document						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.03697	0.03832	0.03783	0.03684		
2K	0.02702	0.02462	0.02392	0.02357	0.02222	
4K	0.01741	0.01402	0.01149	0.01109	0.01252	
8K	0.01099	0.00754	0.00377	0.00408	0.00041	
16K	0.00444	0.00196	0.00123	0.00023	0.00021	
32K	0.00288	0.00019	0.00016	0.00013	0.00012	
64K	0.00278	0.00011	0.00009	0.00009	0.00009	
128K	0.00146	0.00007	0.00006	0.00006	0.00006	
256K	0.00137	0.00006	0.00006	0.00006	0.00006	
512K	0.00006	0.00006	0.00006	0.00006	0.00006	
1M	0.00006	0.00006	0.00006	0.00006	0.00006	
2M	0.00006	0.00006	0.00006	0.00006	0.00006	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.16984	0.12365	0.10317	0.09943		
2K	0.11173	0.07683	0.05868	0.05008	0.04472	
4K	0.06610	0.04835	0.03039	0.02468	0.02375	
8K	0.04343	0.02604	0.02249	0.02056	0.02020	
16K	0.02949	0.01840	0.01775	0.01770	0.01750	
32K	0.02319	0.01684	0.01640	0.01634	0.01635	
64K	0.02090	0.01601	0.01581	0.01578	0.01577	
128K	0.01628	0.01574	0.01568	0.01567	0.01568	
256K	0.01577	0.01555	0.01555	0.01556	0.01556	
512K	0.01077	0.01415	0.01551	0.01551	0.01550	
1M	0.00368	0.00361	0.00361	0.00361	0.00361	
2M	0.00359	0.00357	0.00357	0.00357	0.00357	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.11566	0.08226	0.07496	0.07176		
2K	0.07705	0.05765	0.05085	0.04910	0.04862	
4K	0.04919	0.03524	0.02861	0.02738	0.02617	
8K	0.03259	0.01895	0.01378	0.01259	0.00813	
16K	0.01680	0.00881	0.00629	0.00424	0.00416	
32K	0.01100	0.00461	0.00389	0.00366	0.00364	
64K	0.00921	0.00412	0.00345	0.00342	0.00341	
128K	0.00544	0.00394	0.00333	0.00332	0.00332	
256K	0.00515	0.00329	0.00328	0.00328	0.00328	
512K	0.00301	0.00298	0.00326	0.00326	0.00326	
1M	0.00151	0.00079	0.00079	0.00078	0.00079	
2M	0.00104	0.00078	0.00078	0.00078	0.00078	

Block Size: 64 bytes

Audio						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.00548	0.00443	0.00441	0.00405		
2K	0.00307	0.00243	0.00242	0.00226	0.00222	
4K	0.00089	0.00075	0.00069	0.00069	0.00066	
8K	0.00033	0.00025	0.00024	0.00025	0.00026	
16K	0.00020	0.00015	0.00011	0.00010	0.00009	
32K	0.00013	0.00009	0.00009	0.00008	0.00008	
64K	0.00011	0.00008	0.00008	0.00008	0.00008	
128K	0.00008	0.00008	0.00008	0.00008	0.00008	
256K	0.00008	0.00008	0.00008	0.00008	0.00008	
512K	0.00008	0.00008	0.00008	0.00008	0.00008	
1M	0.00008	0.00008	0.00008	0.00008	0.00008	
2M	0.00008	0.00008	0.00008	0.00008	0.00008	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.16880	0.07865	0.05483	0.04751		
2K	0.09495	0.05388	0.03998	0.03571	0.03506	
4K	0.05083	0.03019	0.02718	0.02721	0.02720	
8K	0.02591	0.01937	0.01803	0.01815	0.01828	
16K	0.01545	0.00895	0.00765	0.00755	0.00754	
32K	0.00772	0.00434	0.00384	0.00366	0.00367	
64K	0.00415	0.00216	0.00191	0.00181	0.00165	
128K	0.00281	0.00146	0.00141	0.00140	0.00139	
256K	0.00263	0.00140	0.00139	0.00139	0.00139	
512K	0.00163	0.00139	0.00139	0.00139	0.00139	
1M	0.00140	0.00139	0.00139	0.00139	0.00139	
2M	0.00139	0.00139	0.00139	0.00139	0.00139	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.07907	0.03945	0.02680	0.02489		
2K	0.04473	0.02278	0.01518	0.01377	0.01288	
4K	0.02859	0.01251	0.01011	0.00963	0.00930	
8K	0.01570	0.00761	0.00694	0.00707	0.00713	
16K	0.00765	0.00367	0.00300	0.00300	0.00299	
32K	0.00441	0.00199	0.00127	0.00115	0.00111	
64K	0.00263	0.00075	0.00065	0.00064	0.00066	
128K	0.00202	0.00042	0.00036	0.00035	0.00034	
256K	0.00066	0.00035	0.00034	0.00034	0.00034	
512K	0.00042	0.00034	0.00034	0.00034	0.00034	
1M	0.00035	0.00034	0.00034	0.00034	0.00034	
2M	0.00035	0.00034	0.00034	0.00034	0.00034	
Video						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.01242	0.01421	0.01368	0.01568		
2K	0.00572	0.00400	0.00413	0.00403	0.00406	
4K	0.00255	0.00189	0.00167	0.00170	0.00161	
8K	0.00145	0.00107	0.00112	0.00119	0.00120	
16K	0.00105	0.00044	0.00024	0.00015	0.00011	
32K	0.00051	0.00018	0.00011	0.00011	0.00011	
64K	0.00014	0.00011	0.00011	0.00011	0.00011	
128K	0.00012	0.00011	0.00011	0.00011	0.00011	
256K	0.00012	0.00011	0.00011	0.00011	0.00011	
512K	0.00012	0.00011	0.00011	0.00011	0.00011	
1M	0.00012	0.00011	0.00011	0.00011	0.00011	
2M	0.00011	0.00011	0.00011	0.00011	0.00011	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.11294	0.06777	0.05383	0.04645		
2K	0.06915	0.03212	0.02507	0.02473	0.02222	
4K	0.03775	0.01376	0.01186	0.01174	0.01216	
8K	0.02305	0.00688	0.00556	0.00538	0.00534	
16K	0.00906	0.00425	0.00335	0.00314	0.00310	
32K	0.00557	0.00317	0.00291	0.00296	0.00296	
64K	0.00393	0.00288	0.00273	0.00252	0.00250	
128K	0.00275	0.00208	0.00208	0.00190	0.00191	
256K	0.00222	0.00180	0.00177	0.00177	0.00176	
512K	0.00199	0.00176	0.00176	0.00176	0.00176	
1M	0.00186	0.00176	0.00176	0.00176	0.00176	
2M	0.00180	0.00176	0.00176	0.00176	0.00176	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.07808	0.04759	0.04261	0.04140		
2K	0.04363	0.01955	0.01548	0.01304	0.01222	
4K	0.02349	0.00864	0.00613	0.00606	0.00599	
8K	0.01284	0.00370	0.00284	0.00276	0.00277	
16K	0.00669	0.00208	0.00172	0.00173	0.00176	
32K	0.00389	0.00123	0.00078	0.00064	0.00066	
64K	0.00158	0.00081	0.00057	0.00051	0.00050	
128K	0.00109	0.00046	0.00044	0.00041	0.00044	
256K	0.00076	0.00040	0.00039	0.00039	0.00039	
512K	0.00065	0.00039	0.00038	0.00038	0.00038	
1M	0.00059	0.00039	0.00038	0.00038	0.00038	
2M	0.00056	0.00038	0.00038	0.00038	0.00038	

Berkeley Multimedia Workload

Multimedia						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.01738	0.01599	0.01564	0.00842		
2K	0.00938	0.00828	0.00831			
4K	0.00581	0.00491	0.00455	0.00422	0.00429	
8K	0.00382	0.00293	0.00256	0.00260	0.00188	
16K	0.00203	0.00127	0.00102	0.00073	0.00072	
32K	0.00124	0.00047	0.00034	0.00030	0.00025	
64K	0.00092	0.00020	0.00014	0.00012	0.00012	
128K	0.00048	0.00012	0.00011	0.00011	0.00011	
256K	0.00039	0.00011	0.00011	0.00011	0.00011	
512K	0.00018	0.00011	0.00011	0.00011	0.00011	
1M	0.00012	0.00011	0.00011	0.00011	0.00011	
2M	0.00011	0.00011	0.00011	0.00011	0.00011	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.21064	0.11964	0.08934			
2K	0.14187	0.06990	0.04902	0.04515		
4K	0.06439	0.03596	0.02661	0.02389	0.02295	
8K	0.03771	0.01995	0.01679	0.01591	0.01557	
16K	0.02247	0.01234	0.01099	0.01070	0.01055	
32K	0.01379	0.00864	0.00776	0.00766	0.00760	
64K	0.01036	0.00675	0.00628	0.00616	0.00609	
128K	0.00724	0.00584	0.00562	0.00553	0.00553	
256K	0.00611	0.00524	0.00519	0.00519	0.00519	
512K	0.00443	0.00479	0.00502	0.00501	0.00501	
1M	0.00272	0.00259	0.00256	0.00256	0.00256	
2M	0.00265	0.00255	0.00254	0.00254	0.00254	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.11710	0.06483	0.05151			
2K	0.07503	0.03941	0.02996	0.02809		
4K	0.04469	0.02295	0.01684	0.01583	0.01530	
8K	0.02500	0.01129	0.00916	0.00862	0.00824	
16K	0.01460	0.00631	0.00515	0.00454	0.00420	
32K	0.00979	0.00366	0.00286	0.00264	0.00256	
64K	0.00713	0.00233	0.00196	0.00188	0.00190	
128K	0.00327	0.00175	0.00140	0.00131	0.00129	
256K	0.00251	0.00129	0.00121	0.00120	0.00120	
512K	0.00166	0.00114	0.00116	0.00116	0.00115	
1M	0.00119	0.00065	0.00064	0.00064	0.00064	
2M	0.00088	0.00064	0.00063	0.00063	0.00063	
Document						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.02833	0.02756	0.02797			
2K	0.02119	0.01910	0.02022	0.02057		
4K	0.01363	0.01139	0.01090	0.01003	0.01044	
8K	0.00910	0.00630	0.00460	0.00450	0.00091	
16K	0.00384	0.00200	0.00118	0.00018	0.00017	
32K	0.00255	0.00014	0.00012	0.00009	0.00009	
64K	0.00248	0.00008	0.00006	0.00006	0.00006	
128K	0.00099	0.00005	0.00004	0.00004	0.00004	
256K	0.00091	0.00004	0.00004	0.00004	0.00004	
512K	0.00004	0.00004	0.00004	0.00004	0.00004	
1M	0.00004	0.00004	0.00004	0.00004	0.00004	
2M	0.00004	0.00004	0.00004	0.00004	0.00004	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.21413	0.15310	0.13570			
2K	0.13383	0.09635	0.07610	0.06996		
4K	0.07930	0.05551	0.03486	0.02659	0.02278	
8K	0.05123	0.02638	0.02141	0.01853	0.01773	
16K	0.03277	0.01720	0.01587	0.01571	0.01550	
32K	0.02453	0.01544	0.01471	0.01465	0.01464	
64K	0.02170	0.01449	0.01424	0.01421	0.01421	
128K	0.01646	0.01420	0.01413	0.01412	0.01412	
256K	0.01431	0.01403	0.01403	0.01404	0.01404	
512K	0.00927	0.01263	0.01400	0.01400	0.01401	
1M	0.00213	0.00206	0.00205	0.00205	0.00205	
2M	0.00203	0.00201	0.00201	0.00202	0.00202	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.13484	0.08254	0.07237			
2K	0.08699	0.05580	0.04690	0.04531		
4K	0.05494	0.03484	0.03027	0.02837	0.02949	
8K	0.03690	0.02021	0.01509	0.01390	0.01312	
16K	0.01884	0.01052	0.00735	0.00516	0.00374	
32K	0.01278	0.00543	0.00384	0.00328	0.00326	
64K	0.01056	0.00398	0.00310	0.00307	0.00306	
128K	0.00611	0.00381	0.00300	0.00299	0.00299	
256K	0.00547	0.00296	0.00295	0.00295	0.00295	
512K	0.00364	0.00266	0.00294	0.00294	0.00294	
1M	0.00213	0.00046	0.00045	0.00045	0.00045	
2M	0.00073	0.00044	0.00044	0.00044	0.00044	

Block Size: 128 bytes

Audio						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.00347	0.00291	0.00310			
2K	0.00209	0.00148	0.00145	0.00130		
4K	0.00062	0.00054	0.00045	0.00046	0.00047	
8K	0.00022	0.00017	0.00015	0.00015	0.00016	
16K	0.00013	0.00009	0.00007	0.00006	0.00006	
32K	0.00008	0.00005	0.00005	0.00004	0.00004	
64K	0.00007	0.00004	0.00004	0.00004	0.00004	
128K	0.00004	0.00004	0.00004	0.00004	0.00004	
256K	0.00004	0.00004	0.00004	0.00004	0.00004	
512K	0.00004	0.00004	0.00004	0.00004	0.00004	
1M	0.00004	0.00004	0.00004	0.00004	0.00004	
2M	0.00004	0.00004	0.00004	0.00004	0.00004	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.26278	0.11775	0.05698			
2K	0.14867	0.07614	0.03426	0.02865		
4K	0.05735	0.02810	0.02006	0.01975	0.01879	
8K	0.02511	0.01468	0.01278	0.01258	0.01246	
16K	0.01412	0.00642	0.00529	0.00499	0.00483	
32K	0.00723	0.00282	0.00222	0.00208	0.00208	
64K	0.00368	0.00136	0.00112	0.00109	0.00097	
128K	0.00258	0.00080	0.00076	0.00076	0.00075	
256K	0.00241	0.00075	0.00075	0.00075	0.00075	
512K	0.00105	0.00075	0.00075	0.00075	0.00075	
1M	0.00076	0.00075	0.00075	0.00075	0.00075	
2M	0.00075	0.00075	0.00075	0.00075	0.00075	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.10156	0.05102	0.03267			
2K	0.05620	0.02752	0.01438	0.01229		
4K	0.03227	0.01074	0.00793	0.00717	0.00658	
8K	0.01655	0.00589	0.00477	0.00473	0.00479	
16K	0.00734	0.00285	0.00232	0.00230	0.00233	
32K	0.00425	0.00140	0.00076	0.00069	0.00067	
64K	0.00242	0.00049	0.00040	0.00040	0.00041	
128K	0.00186	0.00025	0.00020	0.00019	0.00019	
256K	0.00061	0.00019	0.00018	0.00018	0.00018	
512K	0.00028	0.00019	0.00018	0.00018	0.00018	
1M	0.00020	0.00019	0.00018	0.00018	0.00018	
2M	0.00020	0.00018	0.00018	0.00018	0.00018	
Video						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.00970	0.01129	0.01145			
2K	0.00381	0.00228	0.00231	0.00228		
4K	0.00141	0.00105	0.00087	0.00084	0.00084	
8K	0.00081	0.00065	0.00071	0.00072	0.00071	
16K	0.00058	0.00029	0.00019	0.00011	0.00009	
32K	0.00029	0.00011	0.00006	0.00006	0.00006	
64K	0.00006	0.00006	0.00006	0.00006	0.00006	
128K	0.00006	0.00006	0.00006	0.00006	0.00006	
256K	0.00006	0.00006	0.00006	0.00006	0.00006	
512K	0.00006	0.00006	0.00006	0.00006	0.00006	
1M	0.00006	0.00006	0.00006	0.00006	0.00006	
2M	0.00006	0.00006	0.00006	0.00006	0.00006	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.18076	0.11705	0.09384			
2K	0.11735	0.05403	0.03735	0.03209		
4K	0.06085	0.02270	0.01681	0.01649	0.01722	
8K	0.03760	0.00983	0.00730	0.00732	0.00746	
16K	0.01475	0.00537	0.00384	0.00352	0.00346	
32K	0.00691	0.00305	0.00245	0.00229	0.00226	
64K	0.00451	0.00242	0.00221	0.00221	0.00225	
128K	0.00303	0.00189	0.00191	0.00171	0.00173	
256K	0.00218	0.00126	0.00112	0.00108	0.00105	
512K	0.00160	0.00112	0.00105	0.00105	0.00105	
1M	0.00124	0.00105	0.00105	0.00105	0.00105	
2M	0.00114	0.00105	0.00105	0.00105	0.00105	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.09810	0.02592	0.04856			
2K	0.05767	0.02557	0.01994	0.01822		
4K	0.03235	0.01190	0.00839	0.00714	0.00678	
8K	0.01851	0.00423	0.00276	0.00259	0.00247	
16K	0.00912	0.00219	0.00163	0.00161	0.00162	
32K	0.00517	0.00115	0.00082	0.00073	0.00069	
64K	0.00177	0.00068	0.00048	0.00042	0.00044	
128K	0.00114	0.00040	0.00038	0.00034	0.00034	
256K	0.00072	0.00027	0.00024	0.00023	0.00023	
512K	0.00053	0.00024	0.00023	0.00023	0.00023	
1M	0.00042	0.00023	0.00023	0.00023	0.00023	
2M	0.00037	0.00023	0.00023	0.00023	0.00023	

Berkeley Multimedia Workload

Multimedia						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.01498	0.01289				
2K	0.00800	0.00693	0.00706			
4K	0.00490	0.00424	0.00426	0.00408		
8K	0.00313	0.00248	0.00240	0.00204	0.00210	
16K	0.00171	0.00115	0.00086	0.00059	0.00055	
32K	0.00099	0.00035	0.00027	0.00027	0.00028	
64K	0.00072	0.00014	0.00010	0.00008	0.00007	
128K	0.00038	0.00007	0.00007	0.00007	0.00007	
256K	0.00031	0.00007	0.00007	0.00007	0.00007	
512K	0.00011	0.00007	0.00007	0.00007	0.00007	
1M	0.00007	0.00007	0.00007	0.00007	0.00007	
2M	0.00007	0.00007	0.00007	0.00007	0.00007	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.28455	0.18361				
2K	0.19267	0.10525	0.06941			
4K	0.08810	0.05072	0.03424	0.03135		
8K	0.04977	0.02419	0.01883	0.01618	0.01508	
16K	0.02873	0.01374	0.01114	0.01077	0.01065	
32K	0.01621	0.00845	0.00753	0.00741	0.00735	
64K	0.01127	0.00582	0.00526	0.00512	0.00508	
128K	0.00681	0.00481	0.00455	0.00448	0.00448	
256K	0.00534	0.00422	0.00417	0.00418	0.00420	
512K	0.00349	0.00370	0.00391	0.00389	0.00389	
1M	0.00166	0.00147	0.00144	0.00143	0.00143	
2M	0.00155	0.00141	0.00140	0.00140	0.00141	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.16668	0.08779				
2K	0.10191	0.04897	0.03475			
4K	0.05746	0.02595	0.01905	0.01760		
8K	0.03093	0.01259	0.01006	0.00901	0.00885	
16K	0.01785	0.00695	0.00538	0.00471	0.00478	
32K	0.01135	0.00359	0.00283	0.00248	0.00247	
64K	0.00765	0.00203	0.00167	0.00160	0.00159	
128K	0.00345	0.00142	0.00115	0.00108	0.00105	
256K	0.00251	0.00103	0.00096	0.00094	0.00094	
512K	0.00152	0.00086	0.00088	0.00088	0.00087	
1M	0.00104	0.00037	0.00036	0.00036	0.00036	
2M	0.00069	0.00035	0.00035	0.00035	0.00035	
Document						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.02400	0.02356				
2K	0.01826	0.01747	0.01823			
4K	0.01282	0.01121	0.01212	0.01119		
8K	0.00797	0.00607	0.00617	0.00444	0.00471	
16K	0.00365	0.00234	0.00105	0.00015	0.00015	
32K	0.00222	0.00012	0.00010	0.00007	0.00008	
64K	0.00215	0.00006	0.00005	0.00005	0.00004	
128K	0.00094	0.00004	0.00003	0.00003	0.00002	
256K	0.00090	0.00002	0.00002	0.00002	0.00002	
512K	0.00003	0.00002	0.00002	0.00002	0.00002	
1M	0.00002	0.00002	0.00002	0.00002	0.00002	
2M	0.00002	0.00002	0.00002	0.00002	0.00002	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.28528	0.21145				
2K	0.17821	0.13682	0.11056			
4K	0.10887	0.07583	0.05586	0.05178		
8K	0.06873	0.03543	0.02915	0.02038	0.01681	
16K	0.04088	0.02057	0.01536	0.01467	0.01435	
32K	0.02756	0.01491	0.01372	0.01364	0.01363	
64K	0.02365	0.01366	0.01327	0.01325	0.01324	
128K	0.01601	0.01325	0.01317	0.01315	0.01315	
256K	0.01353	0.01311	0.01310	0.01311	0.01311	
512K	0.00841	0.01172	0.01309	0.01309	0.01309	
1M	0.00125	0.00115	0.00114	0.00114	0.00114	
2M	0.00114	0.00111	0.00111	0.00112	0.00113	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.18344	0.10584				
2K	0.10905	0.06504	0.05386			
4K	0.06772	0.03893	0.03361	0.03189		
8K	0.04340	0.02319	0.01885	0.01664	0.01584	
16K	0.02345	0.01352	0.00946	0.00691	0.00735	
32K	0.01504	0.00583	0.00439	0.00309	0.00305	
64K	0.01132	0.00359	0.00289	0.00286	0.00285	
128K	0.00664	0.00336	0.00280	0.00278	0.00278	
256K	0.00557	0.00277	0.00275	0.00275	0.00275	
512K	0.00371	0.00246	0.00274	0.00274	0.00274	
1M	0.00219	0.00026	0.00025	0.00025	0.00025	
2M	0.00063	0.00025	0.00025	0.00025	0.00025	

Block Size: 256 bytes

Audio						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.00375	0.00237				
2K	0.00259	0.00093	0.00093			
4K	0.00050	0.00035	0.00028	0.00028		
8K	0.00014	0.00012	0.00010	0.00010	0.00010	0.00010
16K	0.00008	0.00006	0.00005	0.00005	0.00005	0.00005
32K	0.00005	0.00003	0.00003	0.00003	0.00003	0.00003
64K	0.00004	0.00003	0.00003	0.00003	0.00003	0.00003
128K	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
256K	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
512K	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
1M	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
2M	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.35165	0.20891				
2K	0.21643	0.12989	0.03937			
4K	0.07549	0.04043	0.01692	0.01586		
8K	0.02845	0.01307	0.00895	0.00845	0.00815	
16K	0.01524	0.00499	0.00412	0.00398	0.00398	
32K	0.00771	0.00211	0.00142	0.00130	0.00129	
64K	0.00390	0.00102	0.00072	0.00072	0.00072	
128K	0.00282	0.00049	0.00043	0.00043	0.00042	
256K	0.00261	0.00042	0.00042	0.00042	0.00042	
512K	0.00084	0.00042	0.00042	0.00042	0.00042	
1M	0.00043	0.00042	0.00042	0.00042	0.00042	
2M	0.00042	0.00042	0.00042	0.00042	0.00042	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.14623	0.07682				
2K	0.08259	0.04061	0.01901			
4K	0.04330	0.01533	0.00840	0.00655		
8K	0.02110	0.00560	0.00343	0.00325	0.00331	0.00331
16K	0.00836	0.00241	0.00175	0.00175	0.00180	0.00180
32K	0.00465	0.00098	0.00049	0.00045	0.00042	0.00042
64K	0.00251	0.00037	0.00026	0.00026	0.00026	0.00026
128K	0.00198	0.00016	0.00012	0.00011	0.00011	0.00011
256K	0.00067	0.00011	0.00010	0.00010	0.00010	0.00010
512K	0.00024	0.00011	0.00010	0.00010	0.00010	0.00010
1M	0.00012	0.00011	0.00010	0.00010	0.00010	0.00010
2M	0.00011	0.00010	0.00010	0.00010	0.00010	0.00010
Video						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.00816	0.00948				
2K	0.00326	0.00166	0.00162			
4K	0.00092	0.00071	0.00071	0.00080		
8K	0.00053	0.00041	0.00043	0.00044	0.00043	
16K	0.00038	0.00019	0.00016	0.00011	0.00014	0.00014
32K	0.00019	0.00009	0.00003	0.00003	0.00003	0.00003
64K	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
128K	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
256K	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
512K	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
1M	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
2M	0.00003	0.00003	0.00003	0.00003	0.00003	0.00003
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.27129	0.19092				
2K	0.18212	0.09649	0.06970			
4K	0.09691	0.04647	0.02912	0.02655		
8K	0.05647	0.01759	0.01284	0.01190	0.01244	
16K	0.02588	0.00834	0.00581	0.00593	0.00608	
32K	0.01078	0.00402	0.00301	0.00266	0.00266	
64K	0.00598	0.00252	0.00200	0.00189	0.00188	
128K	0.00401	0.00205	0.00185	0.00183	0.00188	
256K	0.00265	0.00139	0.00129	0.00131	0.00136	
512K	0.00192	0.00100	0.00077	0.00068	0.00068	
1M	0.00113	0.00078	0.00073	0.00068	0.00068	
2M	0.00088	0.00068	0.00068	0.00068	0.00068	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.14174	0.07631				
2K	0.08513	0.03650	0.02859			
4K	0.04863	0.01787	0.01172	0.00963		
8K	0.02661	0.00643	0.00450	0.00414	0.00438	
16K	0.01371	0.00282	0.00169	0.00162	0.00162	
32K	0.00763	0.00134	0.00102	0.00100	0.00100	
64K	0.00236	0.00071	0.00051	0.00039	0.00036	
128K	0.00146	0.00043	0.00036	0.00033	0.00034	
256K	0.00087	0.00028	0.00025	0.00025	0.00025	
512K	0.00059	0.00021	0.00016	0.00014	0.00014	
1M	0.00039	0.00016	0.00015	0.00014	0.00014	
2M	0.00031	0.00015	0.00014	0.00014	0.00014	

SPEC95

SPEC95						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.12814	0.12820	0.12890	0.13126	0.13256	
2K	0.09019	0.08968	0.09186	0.09420	0.09518	
4K	0.05465	0.05022	0.04883	0.04757	0.04714	
8K	0.03424	0.02879	0.02446	0.02354	0.02303	
16K	0.02147	0.01632	0.01352	0.01287	0.01250	
32K	0.01038	0.00523	0.00529	0.00722	0.00891	
64K	0.00313	0.00134	0.00106	0.00098	0.00095	
128K	0.00175	0.00086	0.00075	0.00072	0.00071	
256K	0.00104	0.00071	0.00069	0.00069	0.00069	
512K	0.00075	0.00066	0.00065	0.00064	0.00064	
1M	0.00065	0.00064	0.00064	0.00063	0.00063	
2M	0.00063	0.00063	0.00063	0.00063	0.00063	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.29864	0.25513	0.21971	0.20041	0.19626	
2K	0.23241	0.19220	0.16593	0.14810	0.14495	
4K	0.17813	0.15020	0.12613	0.11557	0.11365	
8K	0.13697	0.11848	0.10349	0.09361	0.09158	
16K	0.10358	0.09338	0.08988	0.08570	0.08423	
32K	0.08956	0.07978	0.07828	0.07809	0.07750	
64K	0.07643	0.07014	0.06955	0.06986	0.06937	
128K	0.06893	0.06483	0.06476	0.06426	0.06430	
256K	0.06182	0.05911	0.05907	0.05858	0.05902	
512K	0.05391	0.05274	0.05209	0.05227	0.05249	
1M	0.04866	0.04760	0.04756	0.04729	0.04741	
2M	0.04177	0.04109	0.04063	0.04105	0.04132	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.23544	0.21375	0.20431	0.19959	0.20007	
2K	0.18355	0.16143	0.15234	0.14634	0.14306	
4K	0.13466	0.11240	0.10039	0.09568	0.09346	
8K	0.09468	0.07641	0.06746	0.06316	0.06050	
16K	0.06238	0.04808	0.04327	0.04037	0.03892	
32K	0.04447	0.03205	0.03003	0.03070	0.03030	
64K	0.03011	0.02279	0.02108	0.02066	0.02037	
128K	0.02389	0.01874	0.01827	0.01804	0.01805	
256K	0.01943	0.01629	0.01616	0.01594	0.01606	
512K	0.01602	0.01405	0.01377	0.01379	0.01386	
1M	0.01418	0.01232	0.01227	0.01219	0.01224	
2M	0.01185	0.01018	0.01003	0.01016	0.01024	

SPEC95

SPEC95						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.07052	0.07013	0.07025	0.07159	0.07198	
2K	0.04990	0.04897	0.05023	0.05136	0.05187	
4K	0.03025	0.02813	0.02732	0.02661	0.02634	
8K	0.01891	0.01575	0.01390	0.01337	0.01278	
16K	0.01209	0.00870	0.00731	0.00694	0.00680	
32K	0.00594	0.00282	0.00260	0.00352	0.00435	
64K	0.00162	0.00054	0.00028	0.00022	0.00020	
128K	0.00078	0.00015	0.00007	0.00005	0.00005	
256K	0.00022	0.00005	0.00003	0.00004	0.00004	
512K	0.00007	0.00002	0.00001	0.00001	0.00001	
1M	0.00001	0.00000	0.00000	0.00000	0.00000	
2M	0.00000	0.00000	0.00000	0.00000	0.00000	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.28378	0.23803	0.19159	0.15862	0.14790	
2K	0.21587	0.17150	0.13852	0.10749	0.09949	
4K	0.15872	0.12504	0.09812	0.07684	0.07047	
8K	0.11444	0.09225	0.07706	0.05577	0.05113	
16K	0.08062	0.06063	0.05925	0.04878	0.04469	
32K	0.05780	0.04455	0.04152	0.04046	0.03863	
64K	0.04221	0.03476	0.03382	0.03365	0.03347	
128K	0.03551	0.03099	0.03054	0.02998	0.03001	
256K	0.02959	0.02705	0.02706	0.02651	0.02678	
512K	0.02376	0.02248	0.02177	0.02184	0.02195	
1M	0.02022	0.01931	0.01921	0.01907	0.01912	
2M	0.01637	0.01582	0.01560	0.01581	0.01597	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.18761	0.16169	0.15078	0.14319	0.13936	
2K	0.14239	0.11886	0.10842	0.09863	0.09450	
4K	0.10442	0.08314	0.07249	0.06566	0.06241	
8K	0.07182	0.05654	0.04878	0.04136	0.03852	
16K	0.04620	0.03322	0.03033	0.02606	0.02406	
32K	0.03059	0.02054	0.01834	0.01818	0.01738	
64K	0.01885	0.01312	0.01202	0.01158	0.01139	
128K	0.01465	0.01045	0.01004	0.00982	0.00982	
256K	0.01110	0.00885	0.00875	0.00854	0.00861	
512K	0.00869	0.00733	0.00706	0.00707	0.00711	
1M	0.00752	0.00628	0.00624	0.00619	0.00621	
2M	0.00621	0.00515	0.00507	0.00513	0.00518	

Block Size: 16 bytes

SPECint95						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.13452	0.12990	0.12912	0.13034	0.13103	
2K	0.10387	0.09457	0.09251	0.09268	0.09278	
4K	0.06865	0.05833	0.05457	0.05121	0.05037	
8K	0.04187	0.03103	0.02572	0.02352	0.02267	
16K	0.02344	0.01483	0.01014	0.00884	0.00809	
32K	0.01199	0.00467	0.00299	0.00230	0.00213	
64K	0.00492	0.00145	0.00094	0.00077	0.00071	
128K	0.00213	0.00048	0.00027	0.00020	0.00018	
256K	0.00064	0.00017	0.00013	0.00013	0.00014	
512K	0.00014	0.00005	0.00004	0.00002	0.00001	
1M	0.00003	0.00001	0.00000	0.00000	0.00000	
2M	0.00000	0.00000	0.00000	0.00000	0.00000	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.22898	0.18212	0.16975	0.16442	0.16345	
2K	0.15841	0.12140	0.10755	0.10151	0.09858	
4K	0.10949	0.08466	0.07531	0.07221	0.07096	
8K	0.08181	0.06336	0.05810	0.05627	0.05560	
16K	0.06522	0.05098	0.04762	0.04663	0.04627	
32K	0.05339	0.04292	0.04111	0.04058	0.04048	
64K	0.04440	0.03811	0.03702	0.03679	0.03671	
128K	0.03792	0.03447	0.03393	0.03383	0.03378	
256K	0.03242	0.03052	0.03032	0.03023	0.03017	
512K	0.02758	0.02617	0.02609	0.02606	0.02605	
1M	0.02489	0.02470	0.02458	0.02456	0.02455	
2M	0.02449	0.02438	0.02438	0.02438	0.02438	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.22116	0.19623	0.18833	0.18734	0.18807	
2K	0.17206	0.14704	0.13815	0.13493	0.13418	
4K	0.12419	0.10066	0.09100	0.08613	0.08441	
8K	0.07969	0.06155	0.05081	0.04722	0.04367	
16K	0.04747	0.03234	0.02642	0.02389	0.02301	
32K	0.03116	0.01673	0.01319	0.01160	0.01109	
64K	0.01706	0.00903	0.00709	0.00659	0.00638	
128K	0.01016	0.00533	0.00456	0.00439	0.00434	
256K	0.00492	0.00316	0.00287	0.00279	0.00275	
512K	0.00223	0.00139	0.00127	0.00124	0.00122	
1M	0.00131	0.00081	0.00075	0.00074	0.00073	
2M	0.00103	0.00065	0.00064	0.00064	0.00064	

Block Size: 32 bytes

SPECfp95						
Instruction Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.05866	0.06097	0.06169	0.06380	0.06435	
2K	0.03646	0.03968	0.04254	0.04448	0.04545	
4K	0.01887	0.01942	0.01968	0.02003	0.02013	
8K	0.01186	0.01182	0.01103	0.01124	0.01094	
16K	0.00903	0.00763	0.00741	0.00739	0.00749	
32K	0.00428	0.00239	0.00300	0.00507	0.00664	
64K	0.00027	0.00019	0.00001	0.00001	0.00001	
128K	0.00022	0.00001	0.00000	0.00000	0.00000	
256K	0.00005	0.00000	0.00000	0.00000	0.00000	
512K	0.00005	0.00000	0.00000	0.00000	0.00000	
1M	0.00000	0.00000	0.00000	0.00000	0.00000	
2M	0.00000	0.00000	0.00000	0.00000	0.00000	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.33115	0.28153	0.21904	0.16508	0.14762	
2K	0.25796	0.20863	0.17170	0.12057	0.10747	
4K	0.19915	0.16215	0.12782	0.09392	0.08424	
8K	0.15208	0.12916	0.10672	0.07240	0.06497	
16K	0.10786	0.08606	0.08652	0.06851	0.06160	
32K	0.07703	0.06430	0.06058	0.05914	0.05598	
64K	0.05726	0.05094	0.05038	0.05028	0.05001	
128K	0.05209	0.04723	0.04699	0.04608	0.04619	
256K	0.04580	0.04293	0.04313	0.04219	0.04265	
512K	0.03912	0.03808	0.03687	0.03705	0.03726	
1M	0.03499	0.03352	0.03341	0.03318	0.03326	
2M	0.02835	0.02745	0.02705	0.02743	0.02772	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.19540	0.16991	0.15622	0.14448	0.13763	
2K	0.14809	0.12419	0.11180	0.09671	0.08989	
4K	0.10949	0.08765	0.07416	0.06409	0.05863	
8K	0.08021	0.06376	0.05543	0.04471	0.04118	
16K	0.05273	0.03930	0.03765	0.03158	0.02854	
32K	0.03514	0.02602	0.02443	0.02512	0.02399	
64K	0.02294	0.01781	0.01720	0.01679	0.01663	
128K	0.01989	0.01551	0.01535	0.01508	0.01512	
256K	0.01694	0.01403	0.01407	0.01375	0.01388	
512K	0.01433	0.01244	0.01203	0.01207	0.01215	
1M	0.01285	0.01095	0.01089	0.01082	0.01086	
2M	0.01066	0.00898	0.00885	0.00897	0.00906	

SPEC95

SPEC95						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.04346	0.04281	0.04276	0.04326		
2K	0.03117	0.02996	0.03062	0.03121	0.03173	
4K	0.01929	0.01814	0.01753	0.01735	0.01734	
8K	0.01217	0.01017	0.00915	0.00872	0.00843	
16K	0.00769	0.00553	0.00457	0.00431	0.00426	
32K	0.00388	0.00198	0.00164	0.00203	0.00239	
64K	0.00118	0.00040	0.00021	0.00016	0.00014	
128K	0.00054	0.00010	0.00005	0.00004	0.00003	
256K	0.00016	0.00003	0.00002	0.00002	0.00002	
512K	0.00005	0.00001	0.00001	0.00001	0.00001	
1M	0.00001	0.00000	0.00000	0.00000	0.00000	
2M	0.00000	0.00000	0.00000	0.00000	0.00000	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.29918	0.25084	0.20382	0.17029		
2K	0.22604	0.17316	0.14072	0.10492	0.08387	
4K	0.16046	0.12147	0.09434	0.07148	0.05453	
8K	0.11500	0.08648	0.07089	0.05029	0.03710	
16K	0.07997	0.05581	0.05363	0.04132	0.02843	
32K	0.05325	0.03441	0.02957	0.02926	0.02332	
64K	0.02908	0.02179	0.02028	0.01978	0.01970	
128K	0.02332	0.01843	0.01770	0.01717	0.01720	
256K	0.01794	0.01540	0.01538	0.01486	0.01502	
512K	0.01333	0.01203	0.01130	0.01134	0.01140	
1M	0.01070	0.00987	0.00975	0.00967	0.00969	
2M	0.00850	0.00802	0.00792	0.00803	0.00809	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.18375	0.14158	0.12821	0.11922		
2K	0.13518	0.10135	0.08947	0.07792	0.07037	
4K	0.09655	0.06999	0.05894	0.05212	0.04559	
8K	0.06662	0.04688	0.03962	0.03286	0.02721	
16K	0.04228	0.02773	0.02545	0.02087	0.01626	
32K	0.02788	0.01569	0.01310	0.01289	0.01080	
64K	0.01454	0.00860	0.00742	0.00699	0.00683	
128K	0.00993	0.00629	0.00582	0.00561	0.00560	
256K	0.00687	0.00503	0.00494	0.00476	0.00480	
512K	0.00492	0.00392	0.00366	0.00367	0.00369	
1M	0.00406	0.00322	0.00317	0.00314	0.00315	
2M	0.00330	0.00261	0.00258	0.00261	0.00263	

SPEC95

SPEC95						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.03940	0.03553	0.03516			
2K	0.02769	0.02332	0.02296	0.02322		
4K	0.01764	0.01428	0.01322	0.01310	0.01313	
8K	0.00987	0.00802	0.00705	0.00684	0.00636	
16K	0.00641	0.00421	0.00337	0.00319	0.00313	
32K	0.00353	0.00184	0.00138	0.00156	0.00166	
64K	0.00140	0.00066	0.00044	0.00038	0.00037	
128K	0.00062	0.00030	0.00023	0.00022	0.00021	
256K	0.00032	0.00017	0.00014	0.00013	0.00012	
512K	0.00019	0.00011	0.00010	0.00009	0.00009	
1M	0.00011	0.00007	0.00006	0.00006	0.00006	
2M	0.00006	0.00004	0.00004	0.00004	0.00004	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.34370	0.28939	0.24813			
2K	0.25548	0.19705	0.16310	0.12644		
4K	0.18392	0.13663	0.10350	0.08032	0.05125	
8K	0.13131	0.09394	0.07410	0.05353	0.03380	
16K	0.09305	0.06153	0.05436	0.04126	0.02217	
32K	0.06299	0.03601	0.02568	0.02531	0.01605	
64K	0.02834	0.02031	0.01381	0.01343	0.01281	
128K	0.01813	0.01222	0.01127	0.01071	0.01076	
256K	0.01260	0.00957	0.00949	0.00899	0.00909	
512K	0.00844	0.00676	0.00602	0.00604	0.00607	
1M	0.00596	0.00511	0.00497	0.00492	0.00493	
2M	0.00456	0.00407	0.00402	0.00406	0.00409	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.20998	0.14424	0.12874			
2K	0.14893	0.10049	0.08686	0.07645		
4K	0.10435	0.06843	0.05578	0.04911	0.03895	
8K	0.06896	0.04501	0.03667	0.02987	0.02266	
16K	0.04451	0.02770	0.02432	0.01965	0.01309	
32K	0.02946	0.01543	0.01133	0.01101	0.00784	
64K	0.01298	0.00797	0.00532	0.00497	0.00462	
128K	0.00780	0.00425	0.00372	0.00349	0.00348	
256K	0.00492	0.00313	0.00302	0.00285	0.00287	
512K	0.00314	0.00221	0.00195	0.00195	0.00196	
1M	0.00240	0.00167	0.00162	0.00160	0.00161	
2M	0.00190	0.00133	0.00131	0.00132	0.00133	

Block Size: 64 bytes

SPECfp95						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.03100	0.03194	0.03232	0.03342		
2K	0.01935	0.02072	0.02214	0.02307	0.02361	
4K	0.01001	0.01030	0.01037	0.01049	0.01052	
8K	0.00628	0.00622	0.00587	0.00598	0.00589	
16K	0.00473	0.00392	0.00377	0.00374	0.00378	
32K	0.00222	0.00128	0.00157	0.00265	0.00336	
64K	0.00017	0.00010	0.00000	0.00000	0.00000	
128K	0.00013	0.00000	0.00000	0.00000	0.00000	
256K	0.00003	0.00000	0.00000	0.00000	0.00000	
512K	0.00003	0.00000	0.00000	0.00000	0.00000	
1M	0.00000	0.00000	0.00000	0.00000	0.00000	
2M	0.00000	0.00000	0.00000	0.00000	0.00000	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.34067	0.28833	0.22745	0.17401		
2K	0.26299	0.20094	0.16449	0.10644	0.07035	
4K	0.19544	0.14861	0.11503	0.08038	0.05146	
8K	0.15003	0.11608	0.09344	0.06097	0.03812	
16K	0.10626	0.07762	0.07725	0.05629	0.03375	
32K	0.06983	0.04731	0.04081	0.04091	0.03042	
64K	0.03531	0.02912	0.02778	0.02713	0.02706	
128K	0.03145	0.02608	0.02540	0.02456	0.02467	
256K	0.02595	0.02315	0.02330	0.02244	0.02271	
512K	0.02101	0.01991	0.01866	0.01875	0.01887	
1M	0.01841	0.01708	0.01689	0.01676	0.01680	
2M	0.01466	0.01388	0.01371	0.01391	0.01401	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.18741	0.14668	0.13116	0.11826		
2K	0.13573	0.10269	0.08974	0.07201	0.05928	
4K	0.09659	0.07107	0.05781	0.04763	0.03628	
8K	0.06984	0.05044	0.04193	0.03181	0.02332	
16K	0.04564	0.03166	0.03044	0.02382	0.01611	
32K	0.02928	0.01833	0.01592	0.01634	0.01290	
64K	0.01460	0.01032	0.00955	0.00918	0.00911	
128K	0.01223	0.00857	0.00830	0.00805	0.00809	
256K	0.00979	0.00756	0.00759	0.00732	0.00740	
512K	0.00777	0.00651	0.00610	0.00612	0.00617	
1M	0.00683	0.00558	0.00551	0.00547	0.00549	
2M	0.00558	0.00454	0.00449	0.00455	0.00458	

Block Size: 128 bytes

SPECfp95						
Instruction Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.01735	0.01760	0.01816	0.01283		
2K	0.01104	0.01137	0.01241	0.01283		
4K	0.00559	0.00563	0.00559	0.00560	0.00564	
8K	0.00346	0.00341	0.00328	0.00337	0.00345	
16K	0.00252	0.00206	0.00196	0.00194	0.00197	
32K	0.00118	0.00072	0.00084	0.00138	0.00170	
64K	0.00010	0.00006	0.00000	0.00000	0.00000	
128K	0.00007	0.00000	0.00000	0.00000	0.00000	
256K	0.00002	0.00000	0.00000	0.00000	0.00000	
512K	0.00002	0.00000	0.00000	0.00000	0.00000	
1M	0.00000	0.00000	0.00000	0.00000	0.00000	
2M	0.00000	0.00000	0.00000	0.00000	0.00000	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.39414	0.33376	0.28038			
2K	0.29735	0.22676	0.18689	0.12729		
4K	0.22129	0.16402	0.12131	0.08701	0.03700	
8K	0.16816	0.12175	0.09244	0.06094	0.02677	
16K	0.12262	0.08450	0.07598	0.05379	0.02035	
32K	0.08288	0.04944	0.03373	0.03410	0.01775	
64K	0.03098	0.02672	0.01693	0.01661	0.01561	
128K	0.02163	0.01553	0.01459	0.01372	0.01383	
256K	0.01627	0.01322	0.01330	0.01248	0.01265	
512K	0.01202	0.01074	0.00948	0.00953	0.00960	
1M	0.01014	0.00878	0.00855	0.00847	0.00849	
2M	0.00780	0.00701	0.00693	0.00700	0.00705	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.21794	0.15227	0.13479			
2K	0.15314	0.10400	0.08876	0.07315		
4K	0.10756	0.07122	0.05490	0.04528	0.02793	
8K	0.07471	0.04856	0.03747	0.02719	0.01520	
16K	0.05030	0.03182	0.02829	0.02145	0.01021	
32K	0.03295	0.01801	0.01276	0.01293	0.00750	
64K	0.01285	0.00930	0.00592	0.00574	0.00535	
128K	0.00878	0.00512	0.00477	0.00451	0.00454	
256K	0.00643	0.00431	0.00432	0.00407	0.00412	
512K	0.00461	0.00351	0.00311	0.00312	0.00316	
1M	0.00390	0.00287	0.00279	0.00277	0.00277	
2M	0.00311	0.00230	0.00227	0.00229	0.00231	

SPEC95

SPEC95						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.02236	0.02121				
2K	0.01619	0.01396	0.01396			
4K	0.01040	0.00914	0.00883	0.00893		
8K	0.00623	0.00533	0.00513	0.00507	0.00496	
16K	0.00391	0.00288	0.00238	0.00225	0.00215	
32K	0.00204	0.00124	0.00097	0.00097	0.00102	
64K	0.00072	0.00037	0.00019	0.00014	0.00012	
128K	0.00029	0.00007	0.00004	0.00003	0.00002	
256K	0.00009	0.00002	0.00001	0.00001	0.00001	
512K	0.00003	0.00001	0.00001	0.00001	0.00001	
1M	0.00000	0.00000	0.00000	0.00000	0.00000	
2M	0.00000	0.00000	0.00000	0.00000	0.00000	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.41667	0.36569				
2K	0.30412	0.24760	0.21125			
4K	0.21680	0.16523	0.12738	0.10471		
8K	0.15203	0.11208	0.08499	0.06067	0.03559	
16K	0.10690	0.07543	0.06130	0.04592	0.02279	
32K	0.07109	0.04624	0.02767	0.02480	0.01341	
64K	0.03799	0.02765	0.01280	0.01040	0.00947	
128K	0.01758	0.01060	0.00993	0.00749	0.00753	
256K	0.01057	0.00668	0.00654	0.00605	0.00611	
512K	0.00633	0.00412	0.00338	0.00338	0.00341	
1M	0.00363	0.00272	0.00257	0.00253	0.00254	
2M	0.00260	0.00209	0.00206	0.00207	0.00208	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.28138	0.17229				
2K	0.19118	0.11599	0.10091			
4K	0.13093	0.07715	0.06237	0.05492		
8K	0.07943	0.05023	0.03877	0.03066	0.02233	
16K	0.05024	0.03143	0.02595	0.02053	0.01248	
32K	0.03393	0.01835	0.01186	0.01074	0.00680	
64K	0.01741	0.01029	0.00511	0.00410	0.00368	
128K	0.00795	0.00379	0.00331	0.00243	0.00242	
256K	0.00454	0.00219	0.00207	0.00190	0.00191	
512K	0.00244	0.00135	0.00109	0.00109	0.00110	
1M	0.00169	0.00090	0.00084	0.00083	0.00083	
2M	0.00130	0.00068	0.00067	0.00067	0.00068	

SPECint95						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.03692	0.03401				
2K	0.02813	0.02341	0.02265			
4K	0.01896	0.01640	0.01593	0.01610		
8K	0.01115	0.00957	0.00916	0.00895	0.00867	
16K	0.00665	0.00503	0.00401	0.00373	0.00352	
32K	0.00375	0.00224	0.00160	0.00124	0.00119	
64K	0.00151	0.00079	0.00043	0.00031	0.00026	
128K	0.00059	0.00016	0.00008	0.00006	0.00005	
256K	0.00019	0.00004	0.00002	0.00002	0.00002	
512K	0.00005	0.00001	0.00001	0.00001	0.00001	
1M	0.00001	0.00000	0.00000	0.00000	0.00000	
2M	0.00000	0.00000	0.00000	0.00000	0.00000	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.35612	0.30264				
2K	0.25111	0.20744	0.17771			
4K	0.16943	0.12624	0.10715	0.09882		
8K	0.10831	0.07599	0.06561	0.05567	0.05192	
16K	0.07119	0.04124	0.03438	0.03248	0.03064	
32K	0.04689	0.02272	0.01806	0.01616	0.01562	
64K	0.03068	0.01348	0.00987	0.00907	0.00881	
128K	0.01603	0.00818	0.00676	0.00648	0.00644	
256K	0.00899	0.00470	0.00437	0.00427	0.00426	
512K	0.00465	0.00159	0.00152	0.00149	0.00147	
1M	0.00061	0.00036	0.00032	0.00031	0.00031	
2M	0.00037	0.00025	0.00023	0.00023	0.00023	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.26420	0.15450				
2K	0.18713	0.10727	0.09447			
4K	0.13043	0.07159	0.06196	0.05769		
8K	0.07065	0.04580	0.03806	0.03514	0.03401	
16K	0.04156	0.02468	0.02112	0.01939	0.01851	
32K	0.02763	0.01314	0.01030	0.00941	0.00900	
64K	0.01455	0.00672	0.00474	0.00414	0.00384	
128K	0.00730	0.00324	0.00232	0.00205	0.00198	
256K	0.00325	0.00156	0.00130	0.00122	0.00121	
512K	0.00145	0.00053	0.00045	0.00043	0.00041	
1M	0.00055	0.00013	0.00011	0.00010	0.00010	
2M	0.00041	0.00008	0.00007	0.00007	0.00007	

Block Size: 256 bytes

SPECfp95						
Instruction Cache						
Associativity						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.01072	0.01097				
2K	0.00663	0.00639	0.00701			
4K	0.00356	0.00333	0.00315	0.00319		
8K	0.00229	0.00194	0.00190	0.00197	0.00200	
16K	0.00171	0.00116	0.00107	0.00107	0.00105	
32K	0.00068	0.00044	0.00047	0.00076	0.00089	
64K	0.00009	0.00004	0.00000	0.00000	0.00000	
128K	0.00005	0.00000	0.00000	0.00000	0.00000	
256K	0.00001	0.00000	0.00000	0.00000	0.00000	
512K	0.00001	0.00000	0.00000	0.00000	0.00000	
1M	0.00000	0.00000	0.00000	0.00000	0.00000	
2M	0.00000	0.00000	0.00000	0.00000	0.00000	
Data Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.46511	0.41613				
2K	0.34652	0.27972	0.23807			
4K	0.25468	0.19641	0.14355	0.10942		
8K	0.18700	0.14094	0.10050	0.06468	0.02253	
16K	0.13548	0.10278	0.08283	0.05667	0.01651	
32K	0.09046	0.06506	0.03537	0.03172	0.01164	
64K	0.04385	0.03898	0.01515	0.01146	0.01000	
128K	0.01881	0.01254	0.01247	0.00829	0.00841	
256K	0.01183	0.00826	0.00828	0.00748	0.00760	
512K	0.00767	0.00615	0.00487	0.00490	0.00496	
1M	0.00605	0.00461	0.00436	0.00431	0.00432	
2M	0.00438	0.00356	0.00351	0.00354	0.00357	
Unified Cache						
Size	Direct	2-way	4-way	8-way	16-way	
1K	0.29513	0.18652				
2K	0.19442	0.12296	0.10607			
4K	0.13132	0.08159	0.06271	0.05270		
8K	0.08645	0.05377	0.03934	0.02708	0.01299	
16K	0.05718	0.03683	0.02982	0.02145	0.00766	
32K	0.03896	0.02252	0.01311	0.01181	0.00505	
64K	0.01970	0.01314	0.00541	0.00407	0.00354	
128K	0.00847	0.00423	0.00410	0.00274	0.00277	
256K	0.00558	0.00269	0.00268	0.00244	0.00247	
512K	0.00324	0.00201	0.00161	0.00162	0.00165	
1M	0.00259	0.00151	0.00143	0.00141	0.00141	
2M	0.00201	0.00117	0.00115	0.00116	0.00117	

Table 15: Arithmetic Average Miss Ratios for SPEC95 (256 byte Block Size)