Execution Characteristics of Multimedia Applications on a Pentium II Processor

Deependra Talla and Lizy Kurian John
Laboratory of Computer Architecture
Department of Electrical and Computer Engineering
The University of Texas at Austin
{deepu, ljohn}@ece.utexas.edu

Abstract

With the widespread use of 3D graphics, animation, speech recognition, and other media applications, general-purpose processors are increasingly spending their cycles on video and audio processing. However, the characteristics of media applications when executed on general purpose processors are not well understood. Such knowledge is extremely important in guiding the design of future microprocessors and development of media applications. In this paper we characterize the performance of multimedia applications on an Intel Pentium II processor based system. Six different commercial multimedia applications belonging to 3D graphics, streaming video or streaming audio categories are executed on an Intel Pentium II processor and performance is measured. Architectural data pertaining to utilization of various hardware resources on the chip are collected, using on-chip performance monitoring counters. Multimedia applications are seen to have fewer branch instructions than SPECint benchmarks, however more than SPECfp benchmarks. Despite a regular control flow and more available parallelism, the average number of cycles taken to execute an instruction is seen to be higher than that of SPECint. In many aspects, media applications exhibit a behavior between that of SPECint and SPECfp.

Keywords: MMX, workload characterization, multimedia, speculative execution, streaming video/audio.

1. Introduction

For the last two decades, general-purpose processor design has been driven largely by non-realtime, stand-alone applications. Multimedia applications are now starting to become exceedingly important for computer systems as a dominant computing workload [4][5]. Dynamic multimedia component technologies such as video-conferencing, video authoring, visualization, 3D graphics, animation, realistic simulation, speech recognition, and broadband communications hold a great promise. The importance of multimedia technology, services and applications is being widely recognized by microprocessor designers. A number of manufacturers are offering multimedia processors that are claimed to be able to decode coded video streams in real-time in software. Most of such processors like the Trimedia processor from Philips and the Multimedia signal processor from Samsung usually have hardware assists for one or more of the multimedia decoding functions. A number of general-purpose CPU manufacturers are offering multimedia enhanced versions of their CPUs for accelerating audio and video processing. The UltraSPARC processor enhanced with the VIS (Visual Instruction Set) from Sun, the multimedia-enhanced MMX [8] and streaming SIMD Pentium processors from Intel [10], AMD's 3DNow! [16], and Motorola's AltiVec technology are examples. Such CPUs will likely take over multimedia functions like audio-video decoding/encoding, modem, telephony functions, and network access functions on a PC/workstation platform, along with the general purpose computing they currently perform.

Multimedia applications possess several distinguishing characteristics than the normal workloads on desktop computing systems. In contrast to traditional applications, multimedia-rich applications will involve significant computational demands on the processor. Diefendorff and Dubey [4] specified the following characteristics of the media-centric applications: real-time response, processing of continuous-media data types, significant fine and coarse grained parallelism, high instruction-reference locality, and high network and memory bandwidth. However, media applications are still not well understood. For instance,

This work is supported in part by the State of Texas Advanced Technology program grant #403, the National Science Foundation under Grants CCR-9796098 and EIA-9807112, and by Dell, Intel, Microsoft, and IBM.

- Do multimedia applications have fewer branches per instruction and better branch-mispredict ratios (because of regularity in instruction sequence) than integer and desktop workloads?
- Is the cycles per instruction (CPI) of multimedia applications lower than other workloads because of available parallelism, regular code structure, and potentially better speculation?
- Are L1 instruction cache miss rates lower for multimedia applications than other workloads due to presence of loops and processing of multiple pixel points/data?
- Are special addressing modes used to access the regular data structures and do they result in an increased use of complex instructions?
- Recent additions to instruction sets have been floating-point multimedia instructions; what amount of floating-point computations do multimedia applications use?
- Many of these applications can heavily use vectors of packed 8-, 16-, and 32-bit integers and floating-point numbers that allows potential benefits of Single Instruction Multiple Data (SIMD) architectures like the MMX for the Pentium family of processors. Do they take advantage of architectures that support SIMD processing such as MMX?

This paper is an effort to characterize multimedia workloads on the X86 architecture with multimedia-enhanced extensions (MMX). We compare execution characteristics of multimedia workloads with the SPEC benchmarks and other desktop applications (non-multimedia applications). By trying to answer the above questions, we provide insight and analysis based on measurements using built-in performance counters of the processor.

There have been some characterizations of desktop applications running under the Windows NT operating system [1][2][13][14][15]. Bhandarkar and Ding [1] characterized the performance of a Pentium Pro processor for both the SPEC benchmarks and the SYSmark/NT benchmark suite (contains Word, Excel, Powerpoint, Texim, and MaxEDA). Lee et al [2] have examined the performance of common desktop applications (acroread, Netscape, photoshoppe, Powerpoint, and word) on the X86 platform in addition to benchmarking five SPEC programs. However, no clear picture of the execution characteristics of multimedia applications running on PC/desktop computing systems on the X86-platform has been published.

Benchmarks for evaluation of multimedia applications are in their infancy and evolving. A recent effort in this direction is MediaBench [11], which is a collection of complete applications/algorithms for multimedia and communications systems. Intel has its own Media benchmark suite [3], which is a collection of audio, video, graphics and image processing applications. Multimedia extensions for signal processing and image/video processing were evaluated in [6][12]. However, no commercial applications were studied. Also, those studies were on a simulator rather than an actual platform. Our studies characterize several popular commercial media applications on a popular commercial platform, the X86 architecture. In addition, this study is performed with hardware assisted measurements on a real system rather than simulations. Hardware monitoring techniques are potentially more accurate and non-invasive.

We examine the characteristics of multimedia applications in the 3D graphics, streaming video and streaming audio domains that account for 95% of the general multimedia workloads [3]. Existing, popular, and complete commercial multimedia applications have been chosen for this evaluation. Measurements were performed using the built-in performance counters of the processor. Results are presented for several statistics including instruction characteristics, branch-related information, memory performance, and MMX related characteristics. The rest of the paper is organized as follows: Section 2 describes our methodology and our benchmark application suite. Section 3 presents the various execution characteristics of the workloads and we compare them with the SPEC and SYSmark/NT characteristics presented in [1].

2. Methodology

2.1 Workloads

Multimedia workload domain can be broken up into 3D graphics, streaming video and streaming audio. These three sub-domains represent about 95% of the multimedia spectrum [3] and we chose to benchmark two of each sub-domain for a total of six complete applications. Table 1 summarizes the benchmarks used for this work. These workloads represent the most recent and extremely popular applications being run on common desktops/PCs. All of our benchmarks were run for 90 seconds in real-time for the measurement of each statistic such as number of clock cycles, total instructions executed, number of branches, etc. Applications executed from 350 million to 24 billion dynamic instructions (see appendix) for running the ninety seconds. For the case of streaming video and audio applications, the data files were saved on disk rather than load it from the network to eliminate network delays in this evaluation.

Each of the streaming applications use different encoding and decoding algorithms for different file formats and bit transfer rate. For example, *RealAudio* provides 10 different encoding algorithms to provide 14.4 kbps modems users mono feeds with a frequency response of

Table. 1. Summary of the Benchmarks

Application	Description
QuakeII	One of the most popular 3D games ever with excellent graphics, sounds, and smart combat enemies. Processor vendors and graphics accelerator manufacturers use this benchmark as a standard gaming benchmark. The game demo is run with a 1024x768 resolution on a 19-inch monitor. Executed over 17 billion instructions.
Unreal	A recent and feature-rich 3D game that is touted to heavily use the MMX instruction set. The graphics engine in Unreal is more advanced than in QuakeII and the audio engine in Unreal outperforms the QuakeII audio. The game demo is run with a 1024x768 resolution on a 19-inch monitor. Executed over 24 billion instructions.
RealVideo	Delivers high quality digital video at much lower bit-rates than other non-streaming solutions, such as compressed QuickTime, AVI, or MPEG. This technology allows Intranets to deliver video training, corporate communications and presentations to the desktop. A video clip of 4.5-inch by 3.5-inch was played. Executed 2.7 billion instructions.
QuickTime	QuickTime; a multimedia architecture, was developed by Apple to synchronize graphics, text, video, and sound. QuickTime is ideal for synchronizing picture and sound. QuickTime is an economical solution, in terms of bandwidth, for both music and video. An AVI video clip of 9-inch by 7-inch was played. Executed over 7 billion instructions.
Winamp	Winamp is a fast, flexible, high-fidelity music player for Windows 95/98/NT. Winamp supports MP3, MP2, CD, MOD, WAV and other audio formats, custom interfaces called *skins* and *audio visualization* and *audio effect plug-ins.* An MPEG audio stream was played. Executed 1.7 billion instructions.
RealAudio	RealAudio is a great system to deliver streaming audio, both speech and music. The player does not cache downloaded files, hence, users cannot steal clips. Synchronization with video, flash, and a sequence of HTML files provides an excellent vehicle for multimedia presentation. A RealAudio audio stream was played. Executed 350 million instructions.

4kHz, 28.8 kbps modems users mono feeds with a frequency response of 5.5kHz, single channel ISDN users (64 kbps) mono feeds with a frequency response of 10kHz, and dual channel ISDN users (128 kbps) mono feeds with a frequency response of 20kHz (CD audio quality). The applications were executed with the best video and audio quality supported by each of the software.

2.2 Performance Monitoring Tool

The P6 micro-architecture implements two performance counters [7]. Each performance counter has an associated event select register that controls what is counted. The counters are accessed via the RDMSR and WRMSR instructions. Since there are a number of events of interest that are to be measured and the performance counters can only record two events at most, several runs of the benchmarks are performed. Our performance monitoring utility provides us the option of reading only Ring3 events or both Ring3 and Ring0 events. Ring3 events correspond to the user level processes that are active at a particular time. Ring0 events correspond to the operating system processes. For this work, we masked the Ring0 events since we are only interested in the execution characteristics of each multimedia workload without intervention from operating system related events. While evaluating the workloads, no other user process was kept active to minimize pollution from intervention. For a detailed listing of the various performance monitoring events, the interested reader is referred to [1].

3. Detailed Characterization of multimedia workloads

This section presents a detailed characterization of Pentium II processor running the various multimedia applications presented earlier. The performance counter measurements presented in the rest of this paper were done using a 300 MHz Pentium II processor (has MMX by default) with 16KB L1 data and instruction caches, a 512 KB unified L2 cache, and 128 MB DRAM running Windows NT 4.0. We use the SPEC95 integer & floating-point and the SYSmark/NT results presented in [1] on a similar configuration and compare with the results of our

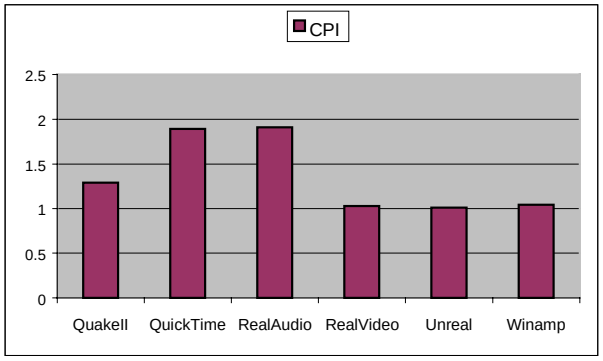
work. Results from their work are indicated with an asterisk (*) and we take the geometric mean of the results.

3.1 Cycles per Instruction

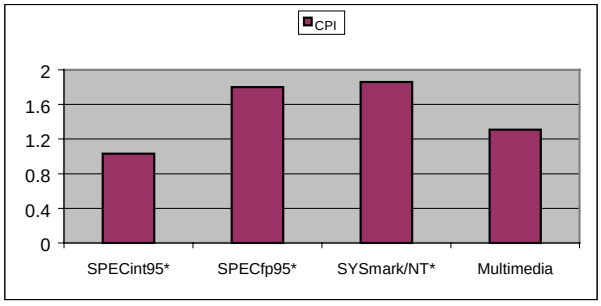
Figure 1 shows the cycles per instruction for each of the individual multimedia applications and mean of multimedia, SPEC and SYSmark/NT benchmarks. The geometric mean of the CPI for the multimedia workloads is 1.31, which lies between the SPECint95 and the SPECfp95 benchmarks. Factors affecting CPI are discussed in detail in subsequent sections. The CPI is seen to be correlated with cache miss ratios and I-stream and resource stalls.

3.2 Resource Stalls and Instruction-stream stalls

Figure 2 shows the I-stream stalls and resource stalls, measured in terms of the cycles in which the stall conditions occur. I-stream stalls are caused by I-cache misses and ITLB misses. Resource stalls show the number of cycles in which resources like register renaming or reorder buffer entries, memory entries, and execution units are full; but these stalls may be overlapped with the execution latency of previously executing instructions.

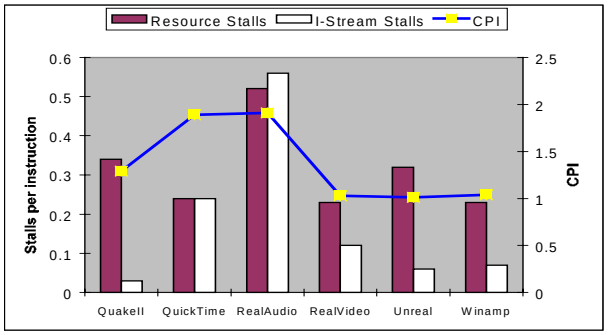


(a)

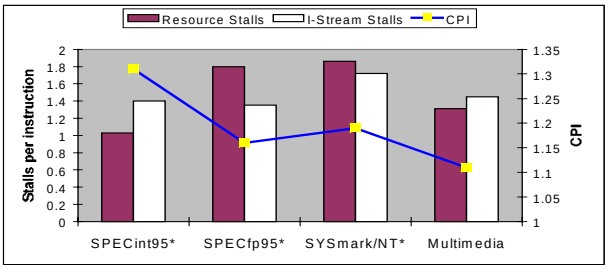


(b)

Figure. 1. Cycles per instruction (a) for individual multimedia benchmarks (b) Comparison of media applications (average) with previous characterizations [1]



(a)



(b)

Figure. 2. Stalls per instruction (a) for individual multimedia benchmarks (b) Comparison of media applications (average) with previous characterizations [1]

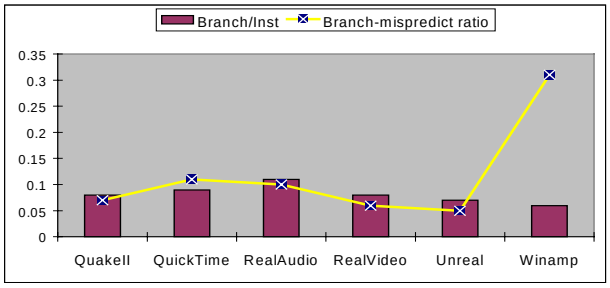
The increase in CPI is directly proportional to the sum of I-stream and resource stalls as observed for Figure 2(a). *RealAudio* has the highest number of Resource and I-stream stalls and exhibits the largest CPI among the multimedia benchmarks. The geometric mean of the resource stalls for the multimedia workload is 0.30 and the I-stream stalls is 0.11. The number of resource stalls in the case of the multimedia applications is over twice the number of stalls for the SPECint95 benchmarks. We suspect that resource stalls would reduce in the Pentium III with the ability of Katmai New Instructions to perform 4 floating-point operations simultaneously. However experiments need to be performed to verify this. Resource stalls for the case of the SYSmark/NT is comparable to the multimedia benchmarks. SPECfp benchmarks incur significantly more resource stalls due to long dependency chains. Interestingly, the number of I-stream stalls per instruction of the multimedia benchmarks is similar to that of the SPECint95 and almost one-third of the SYSmark/NT benchmarks. The number of I-stream stalls for both 3D graphics applications is smaller than that of the audio and video applications.

The combined resource stall and I-stream stall ratios of the multimedia applications are between the SPECint95 and the SPECfp95 ratios. Hence the CPI of the multimedia applications lies in between the SPECint95

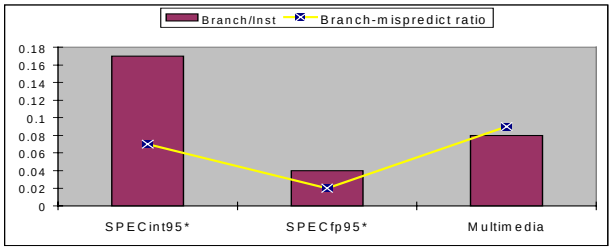
and SPECfp95 benchmark suites as was observed earlier in the CPI ratios.

3.3 Branch Statistics

Figure 3 shows the number of branches per instruction and the branch-mispredict ratio for each of the benchmarks. Branch statistics are not available for the SYSmark/NT. The SPECint95 programs had a branch per instruction ratio of 0.17, SPECfp95 is 0.04 and the ratio for the multimedia suite is 0.08. This means that while one out of every six instructions is a branch in the SPECint95 benchmark suite, only one out of every 12.5 instructions is a branch in multimedia applications and one out of 25 instructions is a branch in the case of floating-point. Thus the average available ILP of multimedia applications is potentially larger than the average ILP of SPECint95 programs. Moreover, in the case of these multimedia applications as will be explained later, MMX instructions operate on four data elements at the same time in a single instruction. In spite of such a processing, the basic block size of multimedia applications is over twice the SPEC suite. However, the CPI for multimedia applications is more than the integer benchmarks. The negative effect of having higher resource stalls is more than the positive effect of fewer branches per instruction. In the case of floating-point benchmarks, longer latencies added with higher resource stalls increase the CPI considerably even when the number of branches is far lesser than any other types of benchmarks.



(a)



(b)

Figure. 3. Branch Statistics (a) for individual multimedia benchmarks (b) Comparison of media applications with previous characterizations [1]

Approximately 7% of all branches are mispredicted in SPECint95 and 2% in SPECfp95, while in multimedia applications 9% of all branches are mispredicted. The number of mispredicted branches range from about 2 to 40 per thousand instructions for the integer benchmarks, about 0.1 to 4 for the floating-point benchmarks and about 3.5 to 16 for the multimedia benchmarks. The SPECint95 had a BTB miss ratio of 0.18, SPECfp95 had 0.07 while the multimedia benchmarks had a BTB miss ratio of 0.15.

3.4 Floating-point operations

Figure 4 shows the amount of floating-point computation being performed in each benchmark. Except *Winamp* and *RealAudio*, the rest of the benchmarks did not exhibit significant amount of floating-point. For the case of SYSmark/NT, the GM of floating-point instructions was 1.35%. Even both the 3D graphics applications were using integer computations as opposed to floating-point operations. Since SPECfp exhibited the highest number of resource stalls (see section 3.2), we suspected that resource stalls were primarily due to inadequate number of floating-point units or their long latency. However, Winamp that had the highest percentage of floating-point operations was not the application that resulted in the highest number of resource stalls. RealAudio which ranked second in percentage of floating-point operations resulted in the highest resource stalls.

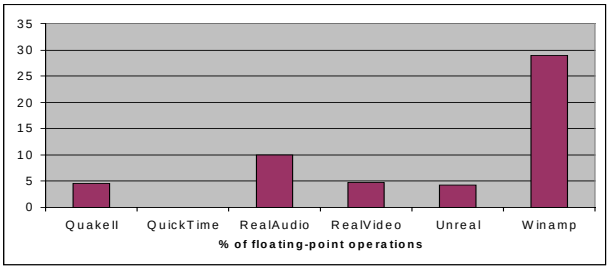


Figure. 4. % of floating-point instructions

3.5 Data Memory References

Figure 5 shows the number of data references per instruction and the number of memory transactions per thousand instructions. On the average, both the SPECint95 and the multimedia benchmarks generate about 1 data reference every two instructions. The IA-32 architecture results in more data references than most RISC architectures because it has fewer registers (8 vs. 32). The memory transactions per instruction are higher in general if the miss rate of the L2 cache is higher. Memory transactions arise from fetching of missed data/instructions and write-back of dirty blocks during replacement.

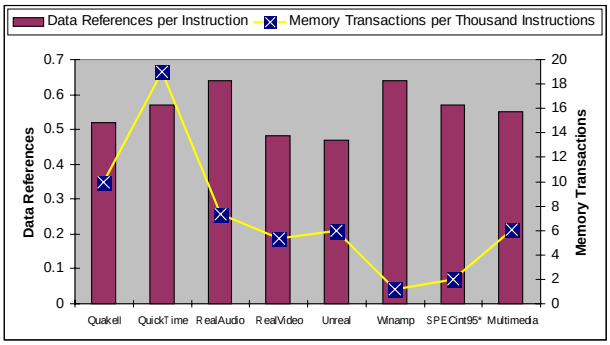
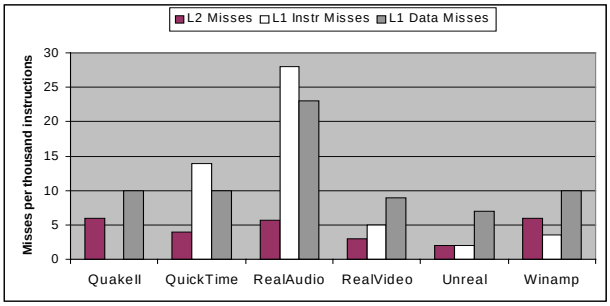


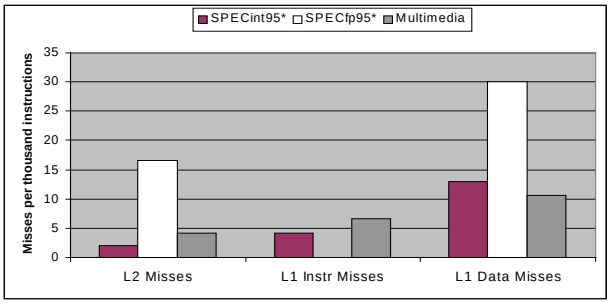
Figure. 5. Memory Reference Statistics

3.6 Cache Misses

The L1 data cache can accept a new load or store every cycle and has a latency of three cycles for loads. It can handle as many as four simultaneously outstanding misses. Figure 6 shows the L1 data and instruction cache misses, and L2 cache misses. The SPECfp programs are dominated by loops, which result in a very predictable control flow, and excellent cache performance as reflected in minimal instruction cache misses. Multimedia applications have fewer branches, however encounter more L1 instruction misses and more I-stream stalls than SPECint95. This is suspected to be due to data dependencies and relatively poor branch prediction.



(a)



(b)

Figure. 6. Cache statistics (a) for individual multimedia benchmarks (b) Comparison of media applications (average) with previous characterizations [1]

Figure 7 shows the correlation between the L2 misses and the L1 misses to the CPI. Using a L1 miss latency of 3 cycles and a L2 miss latency of 50 cycles, correlation with CPI is also shown in figure. It is observed that there is a strong correlation between cache misses and CPI. The L2 cache size was 512 KB in our case as opposed to 256 KB in [1]. Also the size of the L1 cache in the Pentium II is 16 KB each for the instruction and data caches as opposed to 8 KB in the Pentium and Pentium Pro processors.

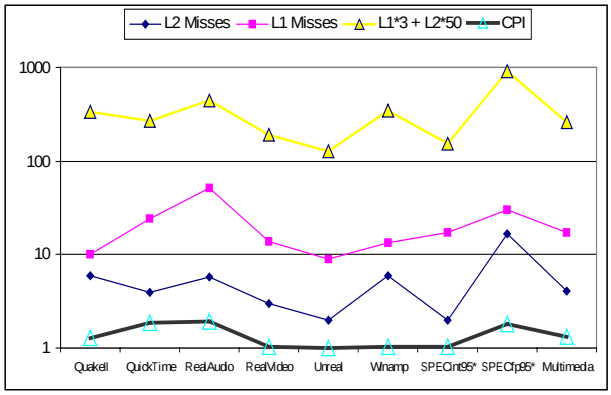


Figure. 7. Log plot of CPI versus L2 and L1 cache misses (for 1000 instructions)

3.7 Multimedia Extensions (MMX)

Multimedia applications can exploit available data parallelism by using SIMD architectures. Unfortunately, not a lot of real applications make use of MMX instructions; mainly because they were developed before MMX technology was introduced. Compiler technology is yet to catch up with SIMD processing as in MMX technology. MMX is especially suited for audio applications, and hence we expected *RealAudio* and *Winamp* to take advantage of MMX instructions. Surprisingly, neither of them uses any MMX instructions. Moreover, *RealAudio* is a part of RealPlayer, which also has *RealVideo*. While *RealVideo* uses MMX instructions, *RealAudio* fails to use any. The percentage of MMX instructions in each of the benchmarks is shown in figure 8. *QuakeII* has been developed before MMX was announced and hence it does not make use of MMX. *Unreal* on the other hand is one of the recent games to use MMX and reflects 50% of all instructions to be MMX.

The total number of MMX instructions can be subdivided into 6 categories: packed multiply, packed shift, pack operations, unpack operations, packed logical operations, and packed arithmetic operations. The overhead involved in MMX computations is the packing and unpacking of instructions. Figure 9 shows the overhead percentage in each of the benchmarks.

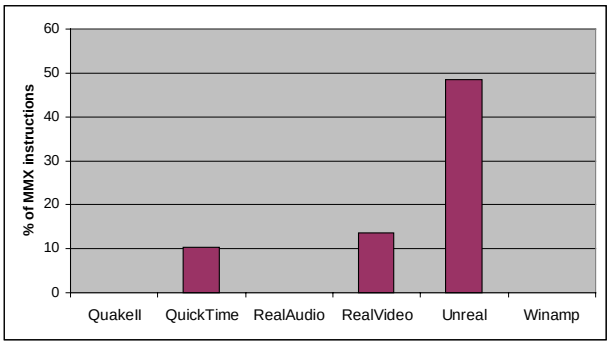


Figure. 8. Percentage of MMX instructions retired of all instructions

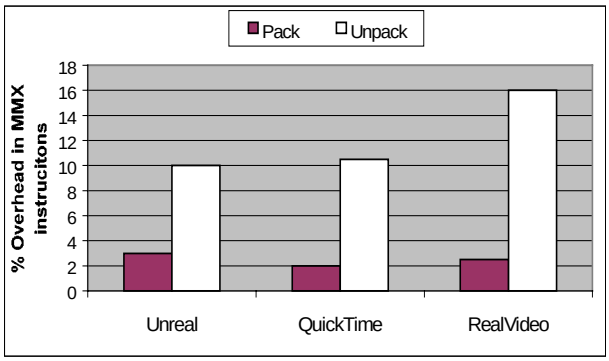
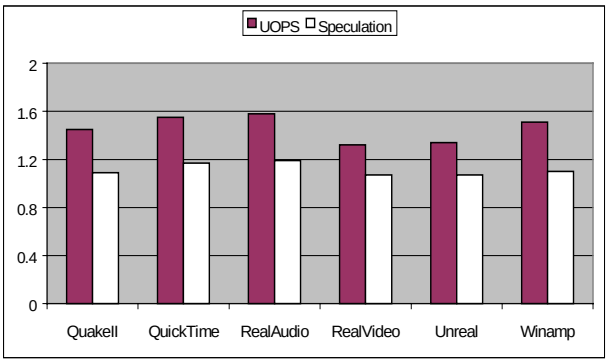


Figure. 9. Packing and unpacking instructions as a percentage of all MMX instructions

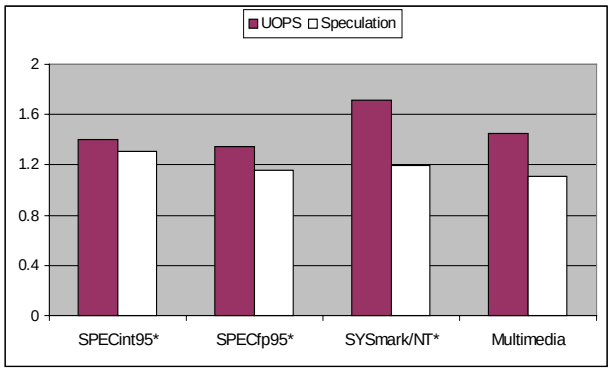
The overall overhead associated in MMX instructions is less than 20% for *RealVideo* and less than 15% for *QuickTime* and *Unreal*. It is interesting to note that the unpacking overhead is several times the packing overhead. Nevertheless, the benefit of using MMX usually exceeds the overhead associated with packing and unpacking of instructions for MMX. *Unreal* has the option of disabling MMX instructions. We observed that the number of frames per second when using MMX was 1.35 times greater than when not using MMX.

3.8 Speculative Execution factor and UOPS per instruction

In the P6 microarchitecture, the instruction fetch unit fetches 16 bytes every clock cycle from the I-cache and delivers them to the instruction decoder. Three parallel decoders decode this stream of bytes and convert them into triadic UOPS. Most instructions are converted directly into single UOPS, some are decoded into one-to-four UOPS, and the complex instructions require microcode. Up to 5 UOPS can be issued every clock cycle to the various execution units, and up to 3 UOPS can be retired every cycle. If a branch is incorrectly predicted, the speculated instructions down the mispredicted path are flushed. Figure 10 shows the speculative execution factor and the number of micro-operations per instruction. Speculative execution factor is defined as the number of instructions decoded, divided by the total number of instructions retired. In the multimedia applications one X86 instruction results in an average of 1.4 micro-ops, very similar to the behavior of SPECint and SPECfp programs. The speculation execution factor of multimedia applications is 1.1, illustrating no significant mis-speculation ratio.



(a)



(b)

Figure. 10. Speculation and UOPS (a) for individual multimedia benchmarks (b) Comparison of media applications (average) with previous characterizations [1]

4. Summary

We evaluated the execution characteristics of several multimedia applications under different domains; 3D graphics, streaming video and streaming audio. Multimedia workloads were compared with the SPECint95 and SYSmark/NT benchmarks presented in [1]. We measured and presented statistics such as CPI, branch statistics, cache statistics, MMX enhancements, etc.

The major observations are summarized below. These answer many of the questions we raised in the introduction.

- The frequency of branches in multimedia applications is less than half of the SPECint workloads, but twice that of SPECfp programs.
- Branch prediction accuracy of media applications is worse than that of SPECint and SPECfp benchmarks. Media kernels would have reflected a branch behavior as in SPECfp, however would not be representative of real commercial applications.
- The CPI of multimedia applications is lower than that of SYSmark/NT and SPECfp suites. The CPI is heavily influenced by resource and I-stream stalls.
- There is at least twice more resource stalls for the multimedia workloads over the SPECint95 suite.
- The L1 instruction cache misses in multimedia applications were observed to be worse than that of SPECint and SPECfp.
- In media applications, one X86 instruction results in an average of 1.4 micro-ops, very similar to the behavior of SPECint and SPECfp.
- Regarding usage of floating-point operations, one of the applications used no floating-point instructions, while others used from 4% to 30% floating-point instructions.
- Three applications used no MMX instructions, while in others, 10% to 50% of all instructions were MMX related.
- Less than 20% of all MMX instructions are used for packing and unpacking operations.
- Roughly one data reference occurs for every two instructions in the X86, regardless of the type of applications being run.

We hope that this paper will be useful for designers of media processors and media applications. We plan to extend our study to assess the benefits of floating-point SIMD techniques like the AMD 3DNow! and the Intel Streaming SIMD extensions.

References

[1] D. Bhandarkar and J. Ding, "Performance Characterization of the Pentium Pro Processor", *Proceedings of High Performance Computer Architecture 97*, pp. 288-297, Feb 1997.

[2] D.C. Lee, P.J. Crowley et al, "Execution Characteristics of Desktop Applications on Windows NT", *Proceedings of International Symposium on Computer Architecture*, pp. 27-38, 1998.

[3] Intel Media Benchmark. http://www.intel.com/procs/perf/icomp/imbwhite/index.htm

[4] K. Diefendorff and P.K. Dubey, "How Multimedia Workloads Will Change Processor Design", *IEEE Computer Magazine*, pp. 43-45, Sep 1997.

[5] C.E. Kozyrakis and D.A. Patterson, "A New Direction for Computer Architecture Research", *IEEE Computer Magazine*, pp. 24-32, Nov 1998.

[6] R. Bhargava, L. John, B. Evans and R. Radhakrishnan, "Evaluating MMX Technology Using DSP and Multimedia Applications", Proceedings of *IEEE Micro-31*, pp. 37-46, Dec 1998.

[7] Intel Corporation, "Pentium Pro Family Developer's Manual, Volume 3:Operating System Writer's Manual", Intel Corporation, Order Number 242692, 1996.

[8] D. Bistry, C. Dulong, M. Gutman, M. Julier, and M. Keith, "The Complete Guide to MMX Technology", McGraw-Hill, 1997.

[9] S. Sriram and C. Hung, "MPEG-2 video Decoding on the TMS320C6x DSP Architecture", *Proceedings of Asilomar98 conference*, Pacific Grove, CA.

[10] L. Gwennap, "Intel's MMX Speeds Multimedia" *Microprocessor Report*, vol. 10, no. 3, p. 1, 1996.

[11] C. Lee, M. Potkonjak and W.H. Smith, "MediaBench: A Tool for Evaluating and Synthesizing Multimedia and Communications Systems", *Proceedings of IEEE Micro-30*, pp. 330-335, Dec 1997.

[12] P. Ranganathan, S. Adve and N. Jouppi, "Performance of Image and Video Processing with General-purpose Processors and Media ISA Extensions", *Proceedings of the 26th International Symposium on computer Architecture*, May 1999.

[13] J. Kalamatianos, R. Chaiken and D. Kaeli, "Parameter Value Characterization of Windows NT-based Applications", *Workshop on Workload Characterization held in conjunction with Micro-31*, Nov 1998. Appears in *Workload Characterization: Methodology and Case Studies*, edited by John and Maynard, IEEE Computer Society Press, 1999.

[14] P. Crowley and J. Baer, "Trace Sampling for Desktop Applications on Windows NT", *Workshop on Workload Characterization held in conjunction with Micro-31*, Nov 1998. Appears in *Workload Characterization: Methodology and Case Studies*, edited by John and Maynard, IEEE Computer Society Press, 1999.

[15] R. Radhakrishnan and F. Rawson, "Characterizing the Behavior of Windows NT Web Server Workloads using Processor Performance counters", *Workshop on Workload Characterization held in conjunction with Micro-31*, Nov 1998. Appears in *Workload Characterization: Methodology and Case Studies*, edited by John and Maynard, IEEE Computer Society Press, 1999.

[16] AMD 3DNow! Website. http://www.amd.com/products/cpg/3dnow/index.html.

Appendix

The Table below displays most of the events measured in our work to allow other researchers to verify and repeat similar experiments. Note that all of the events are in thousands.

Major events measured for the Pentium II processor (in thousands)

EVENTS (in thousands)	**Unreal**	**QuakeII**	**RealVideo**	**QuickTime**	**Winamp**	**RealAudio**
Clocks	25013916	22062867	2808289	13743901	1814517	672606
Ins. Retired	24789317	17139537	2721571	7255868	1737520	351527
Ins. Decoded	26553397	18741177	2913809	8470366	1919724	417270
UOPS Retired	33130832	24811233	3592139	11226989	2628574	555421
Resource Stalls	7827052	5774102	631381	1726427	390699	183488
FLOPS	1052103	771457	128144	25	503758	34901
Branch Retired	1933831	1304312	213951	664972	114267	40316
Branch Miss Predicted	93612	89699	12777	73062	35094	4029
Branch Taken Retired	1118971	754740	171098	505340	89629	32261
BTB misses	322997	141761	28845	175776	12563	14243
Branch Decoded	2140852	1501984	243176	802972	133019	48841
Data Memory Refs	11667621	8853177	1319473	4151930	1104235	223780
DCU Lines In	168875	187443	25290	72690	18135	8294
L2 Ifetch	56131	9691	13461	104022	5950	9915
L2 Requests	221091	194721	37153	174955	22509	15658
L2 Lines In	59008	107544	8809	27158	1058	2003
ITLB misses	2498	310	1029	11465	239	718
IFU Memory stall	1472782	444239	329719	1739785	123971	197279
MMX Retired	12025100	0	368112	752094	0	0
MMX Executed	12379746	0	375938	760982	0	0
MMX Saturated	940499	0	27917	13721	0	0
MMX UOPS	15027936	0	403166	775358	0	0
MMX packed Mul	1069116	0	6812	0	0	0
MMX packed shift	2209325	0	44305	77279	0	0
MMX pack ops	327273	0	8744	13007	0	0
MMX unpack ops	1086100	0	57955	78766	0	0
MMX pack log ops	3815866	0	85636	122435	0	0
MMX pack arith ops	2696052	0	96694	244469	0	0