Memory Performance and Scalability of Intel's and AMD's Dual-Core Processors: A Case Study

Lu Peng[1], Jih-Kwon Peir[2], Tribuvan K. Prakash[1], Yen-Kuang Chen[3] and David Koppelman[1]

[1]*Electrical & Computer Engineering, Louisiana State University, Baton Rouge, LA 70803*
[2]*Computer & Information Science & Engineering, University of Florida, Gainesville, FL 32611*
[3]*Architecture Research Lab, Intel Corporation, Santa Clara, CA 95052*

Abstract

As Chip Multiprocessor (CMP) has become the mainstream in processor architectures, Intel and AMD have introduced their dual-core processors to the PC market. In this paper, performance studies on an Intel Core 2 Duo, an Intel Pentium D and an AMD Athlon 64X2 processor are reported. According to the design specifications, key derivations exist in the critical memory hierarchy architecture among these dual-core processors. In addition to the overall execution time and throughput measurement using both multi-programmed and multi-threaded workloads, this paper provides detailed analysis on the memory hierarchy performance and on the performance scalability between single and dual cores. Our results indicate that for the best performance and scalability, it is important to have (1) fast cache-to-cache communication, (2) large L2 or shared capacity, (3) fast L2 to core latency, and (4) fair cache resource sharing. Three dual-core processors that we studied have shown benefits of some of these factors, but not all of them. Core 2 Duo has the best performance for most of the workloads because of its microarchitecture features such as shared L2 cache. Pentium D shows the worst performance in many aspects due to its technology-remap of Pentium 4.

1. Introduction

Due to advances in circuit integration technology and performance limitations in wide-issue, super-speculative processors, Chip-Multiprocessor (CMP) or multi-core technology has become the mainstream in CPU designs. It embeds multiple processor cores into a single die to exploit thread-level parallelism for achieving higher overall chip-level Instruction-Per-Cycle (IPC) [4,9,12,23,24]. Combined with increased clock frequency, a multi-core, multithreaded processor chip demands higher on- and off-chip memory bandwidth and suffers longer average memory access delays despite an increasing on-chip cache size. Tremendous pressures are put on memory hierarchy systems to supply the needed instructions and data timely.

In this paper, we report performance measurement results on three available dual core desktop processors: Intel Core 2 Duo E6400 with 2.13GHz [10], Intel Pentium D 830 with 3.0GHz [14] and AMD Athlon 64X2 4400+ with 2.2GHz [2]. The Core 2 Duo E6400 was manufactured using 65nm technology with 291 million transistors [10], while the Pentium D 830 and the Athlon 64X2 4400+ were manufactured under 90nm technology with about 230 million transistors [1,18]. In contrast to existing performance evaluations [8,16,17] that usually provide overall execution time and throughput, this paper emphasizes on the memory hierarchy performance. We measure memory access latency and bandwidth as well as cache-to-cache communication delay. We also examine the performance scalability between single and dual cores on the three tested processors.

There are several key design choices for the memory subsystem of the three processors. All three have private L1 caches with different sizes. At the next level, the Intel Core 2 Duo processor adapts a shared L2 cache design, called *Intel Advanced Smart Cache* for the dual cores [12]. The shared L2 approach provides a larger cache capacity by eliminating data replications. It also permits naturally sharing of cache space among multiple cores. When only one core is active, the entire shared L2 can be allocated to the single active core. However, the downside for the shared L2 cache is that it suffers longer hit latency and may encounter competitions of its shared cache resources. Both the Intel Pentium D and the AMD Athlon 64X2 have a private L2 cache for each core, enabling fast L2 accesses, but restricting any capacity sharing among the two cores.

The shared L2 cache in the Core 2 Duo eliminates on-chip L2-level cache coherence. Furthermore, it resolves coherence of the two core's L1 caches internally within the chip for fast access to the L1 cache of the

1-4244-1338-6/07/$25.00/©2007 IEEE

other core. The Pentium D uses an off-chip Front-Side Bus (FSB) for inter-core communications. The Pentium D is basically a technology remap of the Pentium 4 Symmetric Multiprocessor (SMP) that requires to access the FSB for maintaining cache coherence. AMD Athlon 64X2 uses a HyperTransport interconnect technology for faster inter-chip communication. Given an additional ownership state in the Athlon 64X2, cache coherence between the two cores can be accomplished without off-chip traffic. In addition, the Athlon 64X2 has an on-die memory controller to reduce memory access latency.

To examine memory bandwidth and latency, we use *lmbench* [22], a suite of memory measurement benchmarks. *Lmbench* attempts to measure the most commonly found performance bottlenecks in a wide range of system applications. These bottlenecks can be identified, isolated, and reproduced in a set of small micro-benchmarks, which measure system latency and bandwidth of data movement among the processor, memory, network, file system, and disk. We also use a small *lockless* program [19] to measure the cache-to-cache latency of the three processors. The *lockless* program records the duration of ping-pong procedures of a small token bouncing between two caches to get the average cache-to-cache latency. Finally, we run a set of single- and multi- threaded workloads on the three systems to examine the dual-core speedups over a single core. For single-thread programs, we experiment a set of mixed SPEC CPU2000 and CPU2006 benchmarks [20]. For multi-threaded workloads, we select *blastp* and *hmmpfam* from the BioPerf suites [6], SPECjbb2005 [21], as well as a subset of SPLASH2 [26].

According to our experiment results, we can summarize a few interesting findings.

(1) In general, Core 2 Duo and Athlon 64X2 have better overall memory bandwidth and lower latency than Pentium D. The Core 2 Duo processor handles cache coherence between L1 caches on chip and employs aggressive memory dependence predictors. Its shared L2 generates less off-chip traffic than the other two. Athlon 64X2 handles private L2 coherence through on-chip system interfaces. It benefits from its on-chip memory controller for lower memory latency.

(2) The cache-to-cache latency plays an important role in multithreaded workload performance. The cache-to-cache latencies of the selected Core 2 Duo, Pentium D and Athlon 64X2 processors are measured at 33ns, 133ns and 68ns respectively. Core 2 Duo benefits from its on-chip access to the other L1 cache. Pentium D requires off-chip FSB for inter-core communications. Athlon 64X2 employs a fast on-die communication. This benefit is evident when running multithreaded workload with heavy data sharing among multiple cores. The dual-threaded version of selected programs' execution time range from 6.3-490, 8.7-526, and 7.3-621 in second for Core 2 Duo, Pentium D and Athlon 64X2 respectively.

(3) For single threaded benchmarks, Core 2 Duo shows the best performance for most of selected SPEC CPU2000 and CPU2006 workloads running on one core because its shared L2 cache. Execution time of single thread of all workloads range from 56-1500, 75-1703, and 73-1993 in second for Core 2 Duo, Pentium D, and Athlon 64X2 respectively. All three processors demonstrate limited performance scalability for dual-core, where Athlon 64X2 has the best. Core 2 Duo's speed-ups are constraint due to its ability to use the entire L2 cache for running the single thread.

This paper is organized as follows. Section 2 briefly introduces the architectures of the three processors. Section 3 describes the methodology and the workloads of our experiments. Section 4 reports the detailed measurement results and the comparison between the three processors. Section 5 describes related work. Finally, we give a brief conclusion in section 6.

2. Architectures of Dual-Core Processors

The Intel Core 2 Duo (Figure 1(a)) E6400 emphasizes mainly on cache efficiency and does not stress on the clock frequency for high power efficiency. Although clocking at a slower rate than that of the Pentium D, a shorter stages and wider issuing pipeline compensates the performance with higher IPCs. In addition, the Core 2 Duo processor has more ALU units [8]. Core 2 Duo employs a shared L2 cache to increase the effective on-chip cache capacity. Upon a miss from the core's L1 cache, the shared L2 and the L1 of the other core are looked up in parallel before sending the request to the memory [13]**.** The cache block located in the other L1 cache can be fetched without off-chip traffic. Both memory controller and FSB are still located off-chip. The off-chip memory controller can adapt the new DRAM technology with the cost of longer memory access latency. Core 2 Duo employs aggressive memory dependence predictors for memory disambiguation. A load instruction is allowed to be executed before an early store instruction with an unknown address. It also implements a macro-fusion technology to combine multiple micro-operations. Other important features involve support for new SIMD instructions called Supplemental Streaming SIMD Extension 3, coupled with better power saving technologies.

The Pentium D 830 (Figure 1 (b)) glues two Pentium 4 cores together and connects them with the

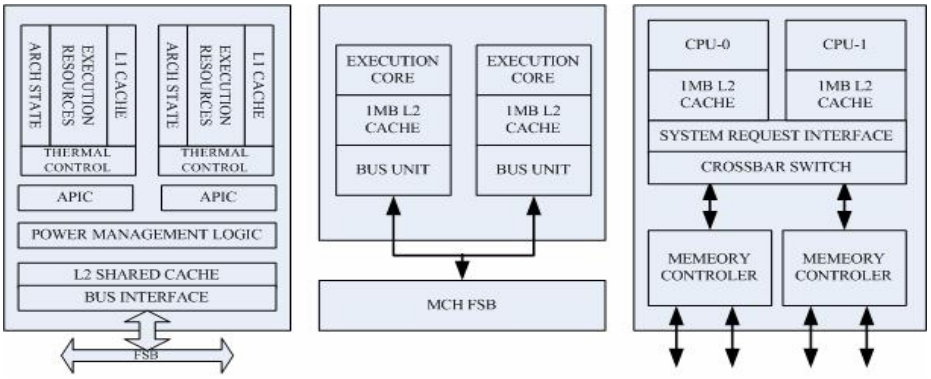


Figure 1. (a) Intel Core 2 Duo; (b) Intel Pentium D; and (c) AMD Athlon 64X2

memory controller through the north-bridge. The off-chip memory controller provides flexibility to support the newest DRAM with the cost of longer memory access latency. The MESI coherence protocol from Pentium SMP is adapted in Pentium D that requires a memory update in order to change a modified block to shared. The system interconnect for processors remains through the Front-Side Bus (FSB). To accommodate the memory update, the FSB is located off-chip that increases the latency for maintaining cache coherence.

The Athlon 64X2 is designed specifically for multiple cores in a single chip (Figure 1(c)). Similar to the Pentium D processor, it also employs private L2 caches. However, both L2 caches share a system request queue, which connects with an on-die memory controller and a HyperTransport. The HyperTransport removes system bottlenecks by reducing the number of buses required in a system. It provides significantly more bandwidth than current PCI technology [3]. The system request queue serves as an internal interconnection between the two cores without involvements of an external bus. The Athlon 64X2 processor employs MOESI protocol, which adds an "Ownership" state to enable blocks to be shared on both cores without the need to keep the memory copy updated.

Another important aspect to alleviate cache miss penalty is data prefetching. According to the hardware specifications, the Intel Core 2 Duo includes a stride prefetcher on its L1 data cache [12] and a next line prefetcher on its L2 cache [8]. The L2 prefetcher can be triggered after detecting consecutive line requests twice. The Pentium D's hardware prefetcher allows stride-based prefetches beyond the adjacent lines. In addition, it attempts to trigger multiple prefetches for staying 256 bytes ahead of current data access locations [11]. The advanced prefetching in Pentium D enables more overlapping of cache misses. The Athlon 64X2 has a next line hardware prefetcher. However, accessing data in increments larger than 64 bytes may fail to trigger the hardware prefetcher [5].

Table 1 lists the specifications of the three processors experimented in this paper. There are no Hyper-threading settings on any of these processors. The Intel Core 2 Duo E6400 has separate 32 KB L1 instruction and data caches per core. A 2MB L2 cache is shared by two cores. Both L1 and L2 caches are 8-way set associative and have 64-byte lines. The Pentium D processor has a Trace Cache which stores 12Kuops. It is also equipped with a write-through, 8-way 16KB L1 data cache with a private 8-way 1MB L2 cache. The Athlon 64X2 processor's L1 data and instruction cache are 2-way 64KB with a private 16-way 1MB L2 cache for each core. Athlon 64X2's L1 and L2 caches in each core is exclusive. All three machines have the same size L2 caches and Memory. The Core 2 Duo and the Pentium D are equipped with DDR2 DRAM using advanced memory controllers in their chipsets. The Athlon 64X2 has a DDR on-die memory controller. All three machines have 2GB memory. The FSB of the Core 2 Duo is clocked at 1066MHz with bandwidth up to 8.5GB/s. The FSB of the Pentium D operates at 800MHz and provides up to 6.4GB/sec bandwidth. The Athlon 64X2 has a 2GHz I/O HyperTransport with bandwidth up to 8GB/s. Bandwidth of hard drive interface for the three machines are 375MB/s, 150MB/s and 300MB/s respectively. Because of our experiments are all in-memory benchmarks, difference in hard drives should have little impact.

3. Methodology

We installed SUSE linux 10.1 with kernel 2.6.16-smp on all three machines. We used maximum level GCC optimization to compile all the C/C++ benchmarks including *lmbench*, SPEC CPU2000, SPEC CPU2006, SPLASH2 and *blastp* and *hmmpfam* from BioPerf. SPECjbb2005 was compiled using SUN JDK 1.5.0.

CPU	Intel Core 2 Duo E6400 (2 x 2.13GHz)	Intel Pentium D 830 (2 x 3.00GHz)	AMD Athlon64 4400+ (2 x 2.20GHz)
Technology	65nm	90nm	90nm
Transistors	291 Millions	230 Millions	230 Millions
Hyperthreading	No	No	No
L1 Cache	Code and Data: 32 KB X 2, 8 way, 64–byte cache line size, write-back	Trace cache: 12Kuops X 2, data: 16KB X 2, 8-way, 64-byte line size, write-through	Code and data: 64KB X 2, 2-way, 64-byte cache line size, write-back
L2 Cache	2MB shared cache (2MB x 1), 8-way, 64-byte line size, non-inclusive with L1 cache.	2MB private cache (1MB x 2), 8-way, 64-byte line size, inclusive with L1 cache.	2MB private cache (1MB x 2), 16-way, 64-byte line size, exclusive with L1 cache.
Memory	2GB (1GB x 2) DDR2 533MHz	2GB(512MBx4) DDR2 533MHz	2GB(1GB x 2) DDR 400MHz
FSB	1066MHz Data Rate 64-bit	800MHz Data Rate 64-bit	HyperTransport 16bit up/down 2GHz Data Rate (up+down)
FSB bandwidth	8.5GB/s	6.4GB/s	8GB/s
HD Interface	SATA 375MB/s	SATA 150MB/s	SATA 300MB/s

Table 1. Specifications of the selected processors

We used *lmbench* suite running on the three machines to measure bandwidth and latency of memory hierarchy. *Lmbench* attempts to measure performance bottlenecks in a wide range of system applications. These bottlenecks have been identified, isolated, and reproduced in a set of small microbenchmarks, which measure system latency and bandwidth of data movement among the processor, memory, network, file system, and disk. In our experiments, we focus on the memory subsystem and measure memory bandwidth and latency with various operations [22]. We can run variable stride accesses to get average memory read latency. We also run multi-copies *lmbench*, one on each core to test the memory hierarchy system.

We measured the cache-to-cache latency using a small *lockless* program [19]. It doesn't employ expensive read-modify-write atomic primitives. Instead, it maintains a *lockless* counter for each thread. The c-code of each thread is as follows.

```
*pPong = 0;
for (i = 0; i < NITER; ++i)
{
    while (*pPing < i)
      *pPong = i+1;
}
```

Each thread increases its own counter *pPong* and keeps reading the peer's counter by checking *pPing*. The counter *pPong* is in a different cache line from the counter *pPing*. A counter *pPong* can be increased by one only after verifying the update of the peer's counter. This generates a heavy read-write sharing between the two cores and produces a Ping-Pong procedure between the two caches. The average cache-to-cache latency is measured by repeating the procedure.

For multiprogrammed workloads, the cross-product of mixed SPEC CPU2000/2006 benchmarks were run on the three machines to examine the dual-core speedups over a single core. All the SPEC CPU2000/2006 programs were run with their respective *ref* inputs. In our simulations, when two programs were run together, we guaranteed that each program was repeated at least four times. The shorter programs may run more than four iterations until the longer program completes its four full iterations. We discarded the results obtained in the first run and used the average execution time and other metrics from the remainder three repeated runs to determine the speedups. We calculated the dual-core speedup for multiprogrammed workloads similarly to that used in [25]. Firstly, the single program's running time were collected individually and were considered as the base runtime. Secondly, the average execution time of each workload when run simultaneously was recorded. Then, the dual-core speedup of each workload is calculated by finding the ratio of average run time when run individually (single core) by the average runtime when run together (dual core). Finally, we add the speedups of the two programs run together to obtain the dual-core speedup. For example, if the speedups of two programs are 0.8 and 0.9 when run simultaneously, the respective dual-core speedup will be 1.7.

We used the same procedure for homogeneous multithreaded workloads including *blastp* and *hmmpfam* from the BioPerf suites, a subset of SPLASH2, as well

Workload	Input parameters
blastp	Swissprot database, large input
hmmpfam	Large input
barnes	1048576 bodies
fmm	524288 particles
ocean-continuous	2050 X 2050 grid
fft	2^24 total complex data points transformed
lu-continuous	4096 X 4096 node matrix
lu-non-continuous	4096 X 4096 node matrix
radix	134217728 keys to sort
SPECjbb2005	Default ramp up time 30s, measurement time 240s, from 1 to 8 warehouses

Table 2. Input parameters of the selected multi-threaded workloads

as SPECjbb2005. The BioPerf suite has emerging Bio-informatics programs. SPLASH2 is a widely used scientific workload suite. SPECjbb2005 is a java based business database program. Table 2 lists the input parameters of the multithreaded workloads used. We ran each of these workloads long enough to compensate overheads of sequential portions of the workloads.

4. Measurement Results

4.1. Memory Bandwidth and Latency

Figure 2 shows memory bandwidth for many operations from *lmbench*. Figure 2(a), 2(c) and 2(e) present data collected while running one copy of *lmbench* on the three machines. Several observations can be made:

(1) In general, Core 2 Duo and Athlon 64 X2 have better bandwidth than that of Pentium D. Only exception is that Pentium D shows the best *memory read* bandwidth when the array size is less than 1MB. The shared cache of Core 2 Duo demands longer access latency though providing larger effective capacity. For Athlon 64X2, because the equipped DRAM has lower bandwidth, its *memory read* bandwidth is lower than that of Pentium D when memory bus is not saturated. The *memory read* bandwidth for the three machines drops when the array size is larger than 32KB, 16KB and 64KB respectively. These reflect the sizes of their L1 cache. When the array size is larger than 2MB, 1MB and 1MB for the respective three systems, we can see another dropping, reflecting their L2 cache sizes.

(2) The *memory bzero* operation shows different behaviors: when the array size is larger than their L1 data cache sizes, i.e., 32KB for Core 2 Duo and 64KB for Athlon 64X2, the memory bandwidth drops sharply. This is not true for Pentium D. The L1 cache of Core 2 Duo and Athlon 64X2 employ a write-back policy while the L1 cache of Pentium D uses a write-through policy. When the array size is smaller than their L1 data cache sizes, the write-back policy updates the L2 cache less frequently than the write-through policy, leading to higher bandwidth. However, when the array size is larger than their L1 data cache sizes, the write-back policy does not have any advantage as indicated by the sharp decline of the bandwidth.

(3) For Athlon 64X2, *libc bcopy unaligned* and *libc bcopy aligned* show a big difference while alignment does not have much difference for Core 2 Duo and Pentium D. 'Aligned' here means the memory segments are aligned to the page boundary. The operation *bcopy* could be optimized if the segments are page aligned. In Figure 2(a), 2(c) and 2(e), Core 2 Duo and Pentium D show optimizations for *unaligned bcopy* access while Athlon 64X2 does not.

Figure 2(b), 2(d) and 2(f) plot the bandwidth while running two copies of *lmbench* on three machines. The scale of the vertical axis of these three figures is doubled compared to their one-copy counterparts. We can observe that memory bandwidth of Pentium D and Athlon 64X2 are almost doubled for all operations. Core 2 Duo has increased bandwidth, but not doubled. This is because of the access contention when two *lmbench* copies compete with the shared cache. When the array size is larger than its L2 cache size 2MB, Athlon 64X2 provides almost doubled bandwidth for two-copy *lmbench memory read* operation compared with its one-copy counterpart. Athlon 64X2 benefits from its on-die memory controller and separate I/O HyperTransport. Intel Core 2 Duo and Pentium D processors suffer FSB bandwidth saturation when the array size exceeds the L2 capacity.

We also measured the memory load latency of the three machines running with one copy and two copies of *lmbench.* From our measurements, Core 2 Duo benefits from its shared L2 cache, which generates lower external traffic. Athlon 64X2 takes the advantage of on-chip memory controller and separate I/O Hyper-Transport. Pentium D's latencies jumps when many memory requests saturate its Front Side Bus. Due to the number of pages limitation, we don't show figures here. Interested readers may request figures from authors or find them in our follow-on journal version paper.

4.2 Multiprogrammed Workload Measurements

We measured execution time of a subset of SPEC CPU2000 and CPU 2006 benchmarks running on the three systems. In figure 3(a) and 3(c), the Core 2 Duo

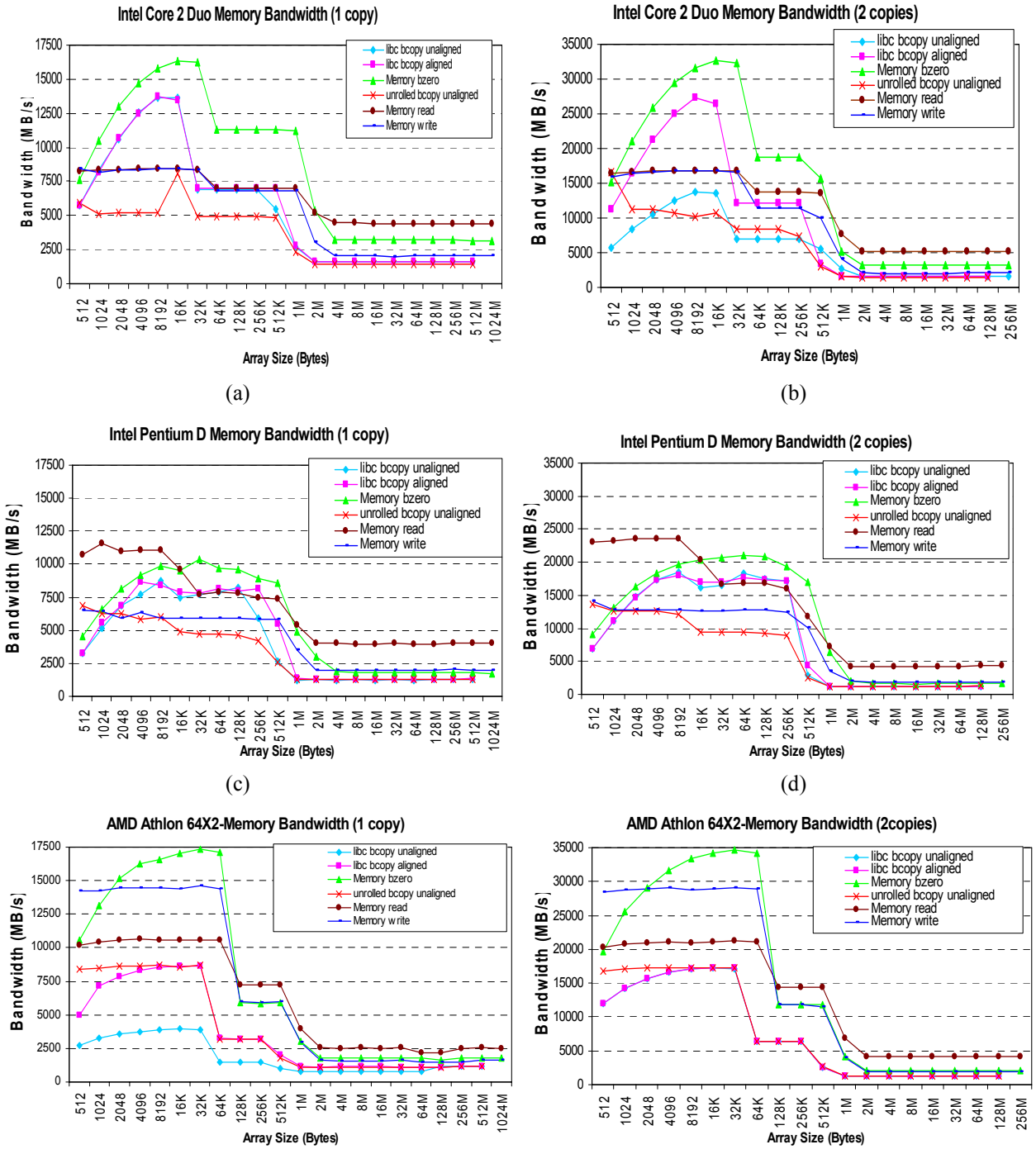


Figure 2. Memory bandwidth collected from the *lmbench* suite with one or two copies.

processor runs fastest for almost all workloads, especially for memory intensive workloads *art* and *mcf*. Core 2 Duo has a wider pipeline, more functional units, and a shared L2 cache that provides bigger cache for single thread. Athlon 64X2 shows the best performance for *ammp*, whose working set is large, resulting in large amount of L2 cache misses for all three machines. Athlon 64X2 benefits from its faster on-chip memory controller.

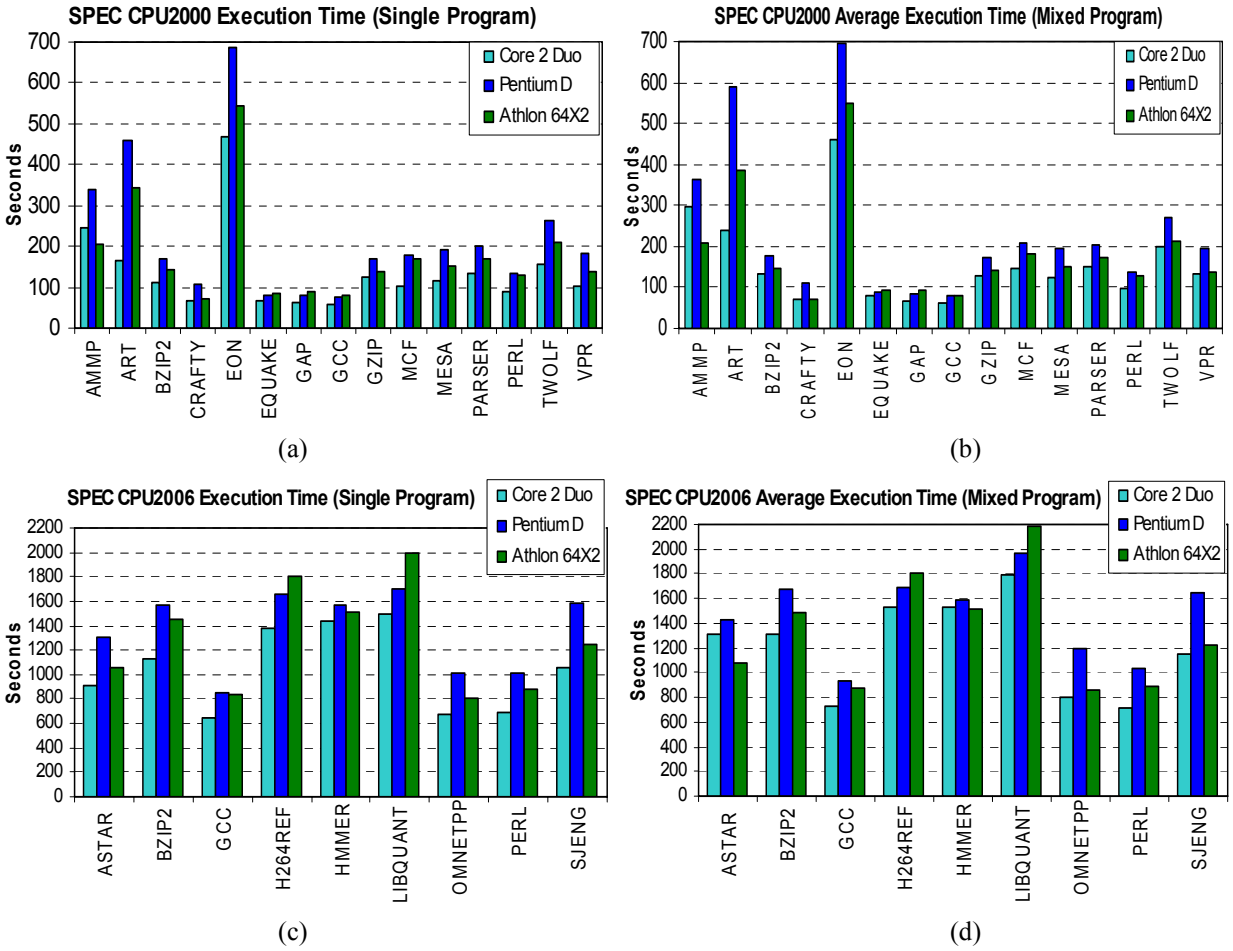


Figure 3. SPEC CPU2000 and CPU2006 benchmarks execution time

Figure 3(b) and 3(d) depict average execution time of each workload when mixed with another program in the same suite. There is an execution time increasing for each workload. For memory bounded programs *art*, *mcf* and *ammp*, execution time increasing is large while CPU bounded workloads such as *crafty*, *mesa, perl* and *sjeng* show a little increasing.

The multi-programmed speedup of the cross-product of mixed SPEC CPU2000 and CPU2006 programs for the three machines are given in the Figure 4, where C2D, PNT and ATH denote the measured Core 2 Duo, Pentium D, and Athlon 64X2 respectively. From Figure 4, we can see that Athlon 64X2 achieves the best speedup 2.0 for all the workloads. *Crafty*, *eon*, *mesa* in CPU 2000 and *perl* in CPU2006 have the best speedup when run simultaneously with other programs because they are CPU bounded instead of memory bounded programs which have comparatively very low L1 D cache misses and hence do not conflict with the other program when running together. On the other hand, in most cases, *art* shows the worst speedup because it is a memory bounded program. Its intensive L2 cache misses occupy the shared memory bus and block another program's execution. In the extreme case, when an instance of *art* was run against another *art*, the speedups were 0.82, 1.11 and 1.36 for Core 2 Duo, Pentium D and Athlon 64X2. Other memory bounded programs, *ammp* and *mcf*, present similar behaviors.

Comparing the three machines, the multi-programmed Athlon 64X2 outperforms those of Core 2 Duo and Pentium D for almost all workload mixes. It is interesting to note that even though Core 2 Duo has better running time than the other two machines, the overall speedup is lesser. The reason again is due to its L2 shared cache.

4.3 Multithreaded Program Behaviors

We use the *lockless* program described in section 3 to measure the dual-core cache-to-cache latency. The average cache-to-cache latency of Core 2 Duo, Pentium D, and Athlon 64X2 are 33ns, 133ns and 68ns respectively. Core 2 Duo resolves L1 cache coherence

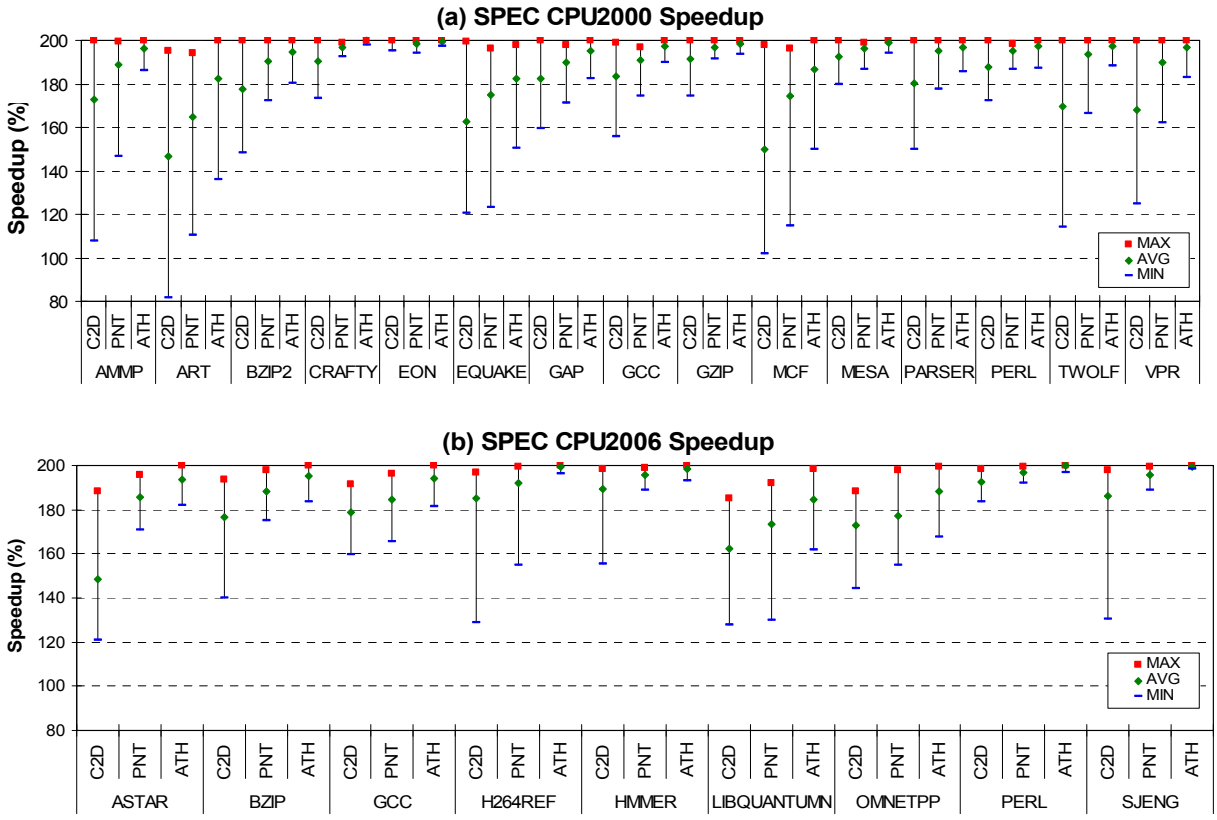
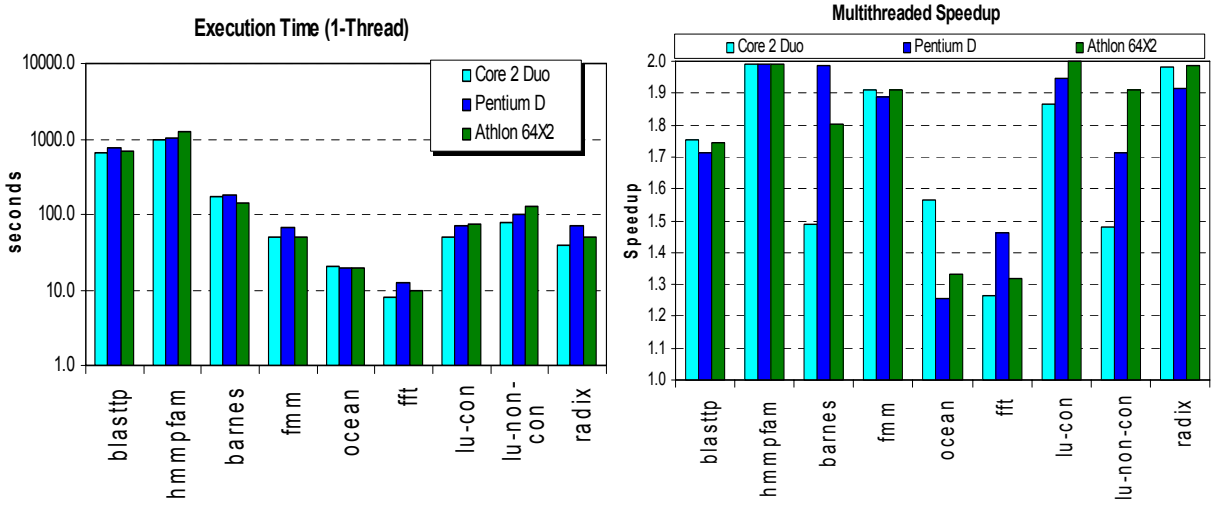


Figure 4. Multi-programmed speedup of mixed SPEC CPU 2000/2006 benchmarks.

within the chip and enables the fastest cache-to-cache transfer. Pentium D requires external FSB for cache-to-cache transfer. Athlon 64X2's on-chip system request interface and the MOESI protocol permits fast cache-to-cache communication.

The multithreaded program execution time and per-



(a) (b)

Figure 5. (a) Execution time for 1-thread version of selected multithreaded programs; (b) Speedup for 2-thread version of selected multithreaded programs

formance speedup of the three systems is presented. We selected *blastp* and *hmmpfam* from the BioPerf suite and a set of the SPLASH2 workloads. Figure 5(a) and 5(b) illustrates execution time of single thread version of the programs and the speedup when running with 2-thread version. In general, Core 2 Duo and Athlon 64X2 do not show performance advantages on bioinformatics and scientific workloads because of less data communication between two cores. Similar results were also reported on Multimedia programs [8]. Core 2 Duo shows the best speedup for *ocean* due to a large amount of cache-to-cache transfers [26]. Pentium D shows the best speed up for *barnes* because of the low cache miss rate. According to our measurement in Section 4.1, the Pentium D processor shows the best *memory read* bandwidth when the array size is small. Bioinformatics workloads have high speedups for all three machines due to their small working sets [6].

Workloads with larger data sharing such as database programs benefit more from the cache-to-cache latency difference. We tested SPECjbb2005 on all the three machines. The throughput for different numbers of warehouses is shown in Figure 6. The throughput reaches its peak point when the number of warehouses equals to 2 due to the dual cores. In all cases, Core 2 Duo shows the best throughput due to its faster FSB and other memory design features. Scalability-wise, the throughput for 2 warehouses of Pentium D and Athlon 64X2 systems are, 1.78 and 1.88 of that for 1 warehouse respectively. The longer cache-to-cache latency in Pentium D accounts for the gap with Athlon 64X2. For the Core 2 Duo system the throughput for 2 warehouses is 1.71 times of that for 1 warehouse. The throughput ratio of Core 2 Duo's 2 warehouses version over 1 warehouse is relatively low because of the competence of its shared L2 cache.

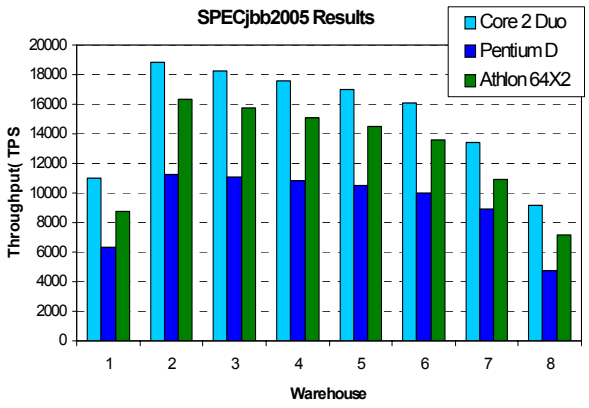


Figure 6. Throughput of SPECjbb2005 running with 1 to 8 warehouses.

From above studies, Core 2 Duo shows its performance advantages for workloads, such as *ocean* and SPECjbb2005, with high data sharing. Basically, Athlon 64X2 doesn't show performance advantage over Pentium D for bioinformatics and scientific workloads though it has faster cache-to-cache data transfer.

5. Related Work

The emergence of Intel and AMD dual core processors intrigues hardware analysts. There are many online reports compare performance of processors from both companies [8,16,17]. Most of them simply present the performance metrics such as running time and throughput without detailed analysis. In this paper, we focus on the memory hierarchy performance analysis and understanding the underlying reasons.

Chip Multiprocessor (CMP) or multi-core technology was first reported in [9]. Companies such as IBM and SUN applied it on their server processors [15,23,24]. In 2005, Intel announced to shelve its plan in pursuing higher frequency and instead switch to building multi-core processors [14]. Similarly, AMD also made the same decision about the same time [4].

Tuck and Tullsen [25] studied thread interactions on an Intel Pentium 4 Hyper-threading processor. They used multi-programmed and multithreaded workloads to measure speedup and synchronization and communication throughput. Bulpin and Pratt [7] measured an SMT processor with consideration about fairness between threads. They also showed the performance gap between SMP and Hyper-threaded SMT for multi-programmed workloads.

6. Conclusion

In this paper, we analyzed the memory hierarchy of selected Intel and AMD dual-core processors. We first measured the memory bandwidth and latency of Core 2 Duo, Pentium D and Athlon 64X2 using *lmbench*. In general, Core 2 Duo and Athlon 64X2 have better memory bandwidth than that of Pentium D. Only exception is that Pentium D shows the best *memory read* bandwidth with small array size.

We measured individual execution time of SPEC CPU2000 and CPU2006. We also measured the average execution time of each application when mixed with other programs on the dual cores. In general, Core 2 Duo runs fastest for all single and mixed applications except for *ammp*. We also observed that memory intensive workloads such as *art*, *mcf* and *ammp* have worse speedups. We measured the cache-to-cache latencies. Core 2 Duo has the shortest, while Pentium D has the longest using off-chip FSB. This generic memory performance behavior is consistent with the per-

formance measurement results of multithreaded workloads such as SPECjbb with heavy data sharing between the two cores.

The Core 2 Duo has two distinct advantages: (1) faster core-to-core communication and (2) dynamic cache sharing between cores. Faster core-to-core communication makes the Core 2 Dual the best for multithreaded workloads with heavily data sharing or communication. However, to manage cache resource efficiently is a challenge especially when two cores have very different demands for caches. In summary, for the best performance and scalability, the following are important factors: (1) fast cache-to-cache communication, (2) large L2 or shared capacity, (3) fast L2 access delay, and (4) fair resource (cache) sharing. Three processors that we studied have shown benefits of some of them, but not all of them.

Acknowledgement

This work is supported in part by the Louisiana Board of Regents grants NSF (2006)-Pfund-80 and LEQSF (2006-09)-RD-A-10 and LSU Faculty Research Grant program and Summer Stipend Program. The authors thank Dr. Carl Staelin at HP Labs for his comments on *lmbench* results. Anonymous referees provide helpful comments.

Reference

[1] AMD, AMD Athlon 64X2 Dual-Core Processor Model Number and Feature Comparisons, http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_948513041%5E13076,00.html.

[2] AMD, AMD Athlon 64X2 Dual-Core Product Data Sheet, http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/33425.pdf.

[3] AMD, AMD HyperTransport Technology, http://www.amd.com/us-en/Processors/DevelopWithAMD/0,,30_2252_2353,00.html.

[4] AMD, Multi-core Processors: The Next Evolution in Computing, http://multicore.amd.com/WhitePapers/Multi-Core_Processors_WhitePaper.pdf, 2005.

[5] AMD, Software Optimization Guide for AMD64 Processors, Chap. 5, Page 105, www.amd.com/us-en/assets/ content_type/white_papers_and_tech_docs/25112.PDF.

[6] D. Bader, Y. Li, T. Li, V. Sachdeva, BioPerf: A Benchmark Suite to Evaluate High-Performance Computer Architecture on Bioinformatics Applications, in Proceedings of the *2005 IEEE International Symposium on Workload Characterization*, Oct. 2005.

[7] J. R. Bulpin and I. A. Pratt, Multiprogramming Performance of the Pentium 4 with Hyper-threading, In *Proceedings of Third Annual Workshop on Duplicating, Deconstructing, and Debunking (WDDD),* Jun. 2004.

[8] F. Delattre and M. Prieur, Intel Core 2 Duo – Test, http://www.behardware.com/articles/623-16/intel-core-2-duo-test.html.

[9] L. Hammond, B. A. Nayfeh and K. Olukotun, A Single-Chip Multiprocessor, *IEEE Computer*, Sep. 1997.

[10] Intel, Announcing Intel Core 2 Processor Family Brand, http://www.intel.com /products/processor/core2/index.htm

[11] Intel, IA-32 Intel Architecture Optimization Reference Manual, Chap. 6, Page 6-4, http://www.intel.com/design/pentium4/manuals/248966.htm.

[12] Intel, Inside Intel Core Microarchitecture and Smart Memory Access. http://download.intel.com/technology/architecture/sma.pdf.

[13] Intel, CMP Implementation in Systems Based on the Intel Core Duo Processor, http://www.intel.com/technology/itj/2006/voume10issue02/art02_CMP_Implementation/p03_implementation.htm.

[14] Intel, Intel Pentium D Processor Product Information, http://www.intel.com/products /processor/pentium_d/.

[15] R. Kalla, B. Sinharoy, and J. M. Tendler, IBM Power5 Chip: A Dual Core Multithreaded Processor. *IEEE Micro*, 24(2):40–47, Mar./Apr., 2004.

[16] A. Mitrofanov, Dual-core processors, http://www.digital-daily.com/cpu/dualcore-cpu/index.htm.

[17] www.motherboards.org, AMD Vesus Intel Battle of the Dual-Core CPUs, http://www.motherboards.Org/reviews/hardware/1513_2.html.

[18] V. Romanchenko, Intel processors today and tomorrow, http://www.digital-daily.com/cpu/intel-roadmap/.

[19] Michael S., How can we measure cache-to-cache transfer speed? http://www.aceshardware.com/forums/read_post.jsp? id=20676&forumid=2.

[20] SPEC, SPEC CPU2000 and CPU2006, http://www.spec.org/

[21] SPEC, SPECjbb2005, http://www.spec.org/jbb2005/

[22] C. Staelin. lmbench --- an extensible micro-benchmark suite. HPL-2004-213. Dec. 2004, http://www.hpl.hp.com/techreports/2004/HPL-2004-213.pdf.

[23] Sun Microsystems, "Sun's 64-bit Gemini Chip," *Sunflash*, 66(4), Aug. 2003.

[24] J. M. Tendler, S. Dodson, S. Fields, H. Le, and B. Sinharoy, "IBM eserver Power4 System Microarchitecture," *IBM White Paper*, Oct. 2001.

[25] N. Tuck and D. M. Tullsen. Initial observations of the simultaneous multithreading Pentium 4 processor. In *Proceedings of the 12th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, pages 26-34, Sep. 2003.

[26] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, The SPLASH-2 Programs: Characterization and Metho-dological Considerations, in *Proceedings of the 22nd Annual International Symposium on Computer Architecture (ISCA)*, pages 24-36, Jun. 1995.