RISC versus CISC: A Tale of Two Chips

Dileep Bhandarkar
Intel Corporation
Santa Clara, California, USA

Abstract

This paper compares an aggressive RISC and CISC implementation built with comparable technology. The two chips are the Alpha* 21164 and the Intel Pentium® Pro processor. The paper presents performance comparisons for industry standard benchmarks and uses performance counter statistics to compare various aspects of both designs.

Introduction

In 1991, Bhandarkar and Clark published a paper comparing an example implementation from the RISC and CISC architectural schools (a MIPS* M/2000 and a Digital VAX* 8700) on nine of the ten SPEC89 benchmarks. The organizational similarity of these machines provided an opportunity to examine the purely architectural advantages of RISC. That paper showed that the resulting advantage in cycles per program ranged from slightly under a factor of 2 to almost a factor of 4, with a geometric mean of 2.7. This paper attempts yet another comparison of a leading RISC and CISC implementation, but using chips designed with comparable semiconductor technology. The RISC chip chosen for this study is the Digital Alpha 21164 [Edmondson95]. The CISC chip is the Intel Pentium® Pro processor [Colwell95]. The results should not be used to draw sweeping conclusions about RISC and CISC in general. They should be viewed as a snapshot in time. Note that performance is also determined by the system platform and compiler used.

Chip Overview

Table 1 shows the major characteristics of the two chips. Both chips are implemented in around 0.5µ technology and the die size is comparable. The design approach is quite different, but both represent state of the art implementations that achieved the highest performance for RISC and CISC architectures respectively at the time of their introduction.

Table 1 Chip Comparison

	Alpha 21164	**Pentium® Pro Processor**
Architecture	Alpha	IA-32
Clock Speed	300 MHz	150 MHz
Issue Rate	Four	Three
Function Units	four	five
Out of order issue	no	yes
Rename Registers	none	40
On-chip Cache	8 KB data 8KB instr 96 KB Level 2	8 KB data 8KB instr
Off chip cache	4 MB	256 KB
Branch History Table	2048 entries, 2-bit history	512 entries, 4-bit history
Transistors		
Logic	1.8 million	4.5 million
Total	9.3 million	5.5 million
VLSI Process	CMOS	BiCMOS
Min. Geometry	0.5 µ	0.6 µ
Metal Layers	4	4
Die Size	298 mm^2	306 mm^2
Package	499 pin PGA	387 pin PGA
Power	50 W	20 W incl cache
First Silicon	Feb. 94	4Q 94
Volume Parts	1Q 95	4Q 95
SPECint92/95	341/7.43	245/6.08
SPECfp92/95	513/12.4	220/5.42
SYSmark/NT	529	497

The 21164 is a quad issue superscalar design that implements two levels of caches on chip, but does not implement out of order execution. The Pentium® Pro processor implements dynamic execution using an out-of-order, speculative execution engine, with register renaming of integer, floating point and flags variables. Consequently, even though the die size is comparable, the total transistor count is quite different for the two chips. The aggressive design of the Pentium Pro processor is much more logic intensive; and logic transistors are less dense. The on-chip 96 KB L2 cache of the 21164 inflates its transistor count. Even though the Alpha 21164 has an on-chip L2 cache, most systems use a 2 or 4 MB board level cache to achieve their performance goal.

.

Alpha 21164

The Alpha 21164 microprocessor consists of five independent functional units as shown in Figure 1:

- Instruction fetch and decode unit (Ibox), which includes:
  - Instruction prefetcher and instruction decoder
  - Branch prediction
  - Instruction translation buffer
  - Interrupt support
- Integer execution unit (Ebox)
- Floating-point execution unit (Fbox)
- Memory address translation unit (Mbox), which includes:
  - Data translation buffer (DTB)
  - Miss address file (MAF)
  - Write buffer
- Cache control and bus interface unit (Cbox) with external interface

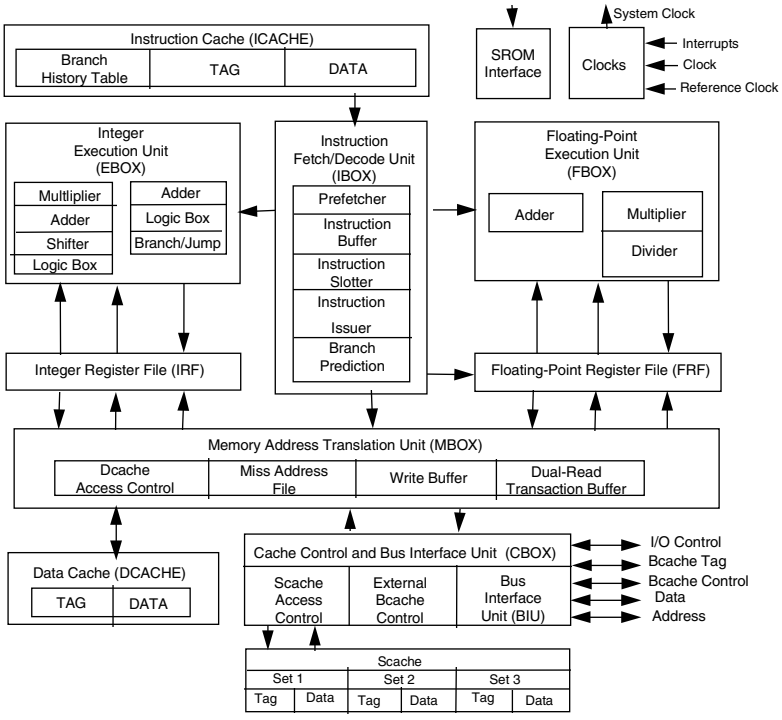

Figure 1 Alpha 21164 Block Diagram

It also includes three on-chip caches; an instruction cache, a data cache, and a second-level cache. The instruction unit fetches, decodes, and issues instructions to the integer unit, memory management unit, and floating point unit. It manages the pipelines, the program counter, the instruction cache, prefetching, and I-stream memory management. It can decode up to 4 instructions in parallel and check that the required resources are available for each.

The integer execution unit contains two 64-bit integer pipelines. Results of most integer operations are available for use by the subsequent instruction. The integer unit also partially executes all memory instructions by calculating the effective address.

The memory management unit processes all load and store instructions. Two loads can be executed in parallel due to the dual-ported data cache. Up to 24 load instructions can be in progress at any time. The memory management unit also manages the data cache and logic that serializes and merges load instructions that miss the data cache.

The cache control and bus interface unit processes all accesses sent by the memory management unit and implements all memory related external interface functions. It controls the second level cache and

.

manages the external backup cache. Its 40 bit physical address supports 1 terabyte of physical address space.

The floating point unit contains a floating point multiply pipeline and a floating point add pipeline. Divides are associated with the add pipeline but are not pipelined themselves.

The instruction cache (Icache) is an 8KB direct mapped, virtually indexed physical cache. The data cache (Dcache) is a dual ported, 8KB write through, read allocate, direct mapped, virtually indexed physically addresssed cache with 32-byte blocks. The access time is 2 cycles and the bandwidth is 16 bytes per cycle. The on-chip second level cache (Scache) is a 96KB, 3-way set associative, physical, write back, write allocate, combined data and instruction cache. It is fully pipelined and supports both 32-byte and 64-byte blocks. Access time is 6 cycles after I or D cache miss, and bandwidth is 16 bytes per cycle.

The 21164 supports and fully manages an optional direct mapped external backup cache (Bcache). This cache is a physical, write back, write allocate cache with 32-byte or 64-byte blocks. The backup cache controller supports wave pipelining in the 64-byte mode. It is a mixed data and instruction cache. The system designer can select a Bcache size of 1, 2, 4, 8, 16, 32, or 64 megabytes. The minimum latency is 4 cycles and a maximum rate of 16 bytes every 4 cycles.

The 21164 chip has a 7-stage pipeline for integer operate and memory reference instructions, and a 9-stage pipeline for floating point operate instructions [Bannon95]. The Ibox maintains state for all pipeline stages to track outstanding register write operations. The first four stages are executed in the Ibox. The Ibox pipeline is the same for all types of instructions. Instructions are fetched from the instruction cache in stage 0 (S0). The instructions are decoded and up to two four-instruction blocks are buffered in S1. Instructions are steered to available issue slots during S2, and instruction issue occurs in S3. Remaining stages are executed by the Ebox, Fbox, Mbox, and Cbox. There are bypass paths that allow the result of one instruction to be used as a source operand of a following instruction before it is written to the register file.

Table 2 Alpha 21164 Instruction Pipeline

	Memory Access Pipeline	**Integer Pipeline**	**Floating Point Pipeline**
Stage 0:	IC: Icache read		
Stage 1:	IB: Instruction buffer, branch decode, determine next PC		
Stage 2:	SL: Slot by function unit		
Stage 3:	AC: Check issue conflicts, access integer registers		
Stage 4:	D1: Begin Dcache read	first integer operate	FP register file access
Stage 5:	D2: End Dcache read	next integer operate	FP operate 1
Stage 6:	WR: If Dcache hit, write register file and use data. If Dcache miss, begin Scache tag access	write integer result	FP operate 2
Stage 7:	ST: Finish Scache tag access		FP operate 3
Stage 8:	S0: Begin Scache access		write FP register file
Stage 9:	S1: End Scache access		
Stage 10:	BD: Begin Dcache fill		
Stage 11:	FD: Finish Dcache fill		
Stage 12:	DA: Data available		

The pipeline is shown in Table 2. The register file is read in S3; execution begins in S4 and completes at the end of S4, except for the integer multiply and conditional move instructions; the second and last operate stage for conditional move instructions is S5; and results are written to the integer register file in S6. Multiplies have variable latencies. The floating point register file is read in S4; execution occurs in S5 through S8; and results are written to the floating point register file at the end of S8.

.

Virtual address calculation occurs and data cache access begins in S4; data cache access completes, address translation occurs, and cache hit is calculated in S5. If the access hits, the data is written to the register file (load case) or the cache (store case) in S6. The pipeline continues in the case of a data cache miss as follows: second-level cache access begins in S6 and ends in S9; the data cache fill begins in S10; the integer register file write occurs and the data cache fill completes in S11; and a dependent instruction can begin execution in S12. In every case except fills to the data cache, bypasses are implemented so that dependent instructions can begin execution as soon as the result is available. For data cache fills, dependent instructions can begin execution immediately after the fill data is written to the register file.

Pentium® Pro Processor

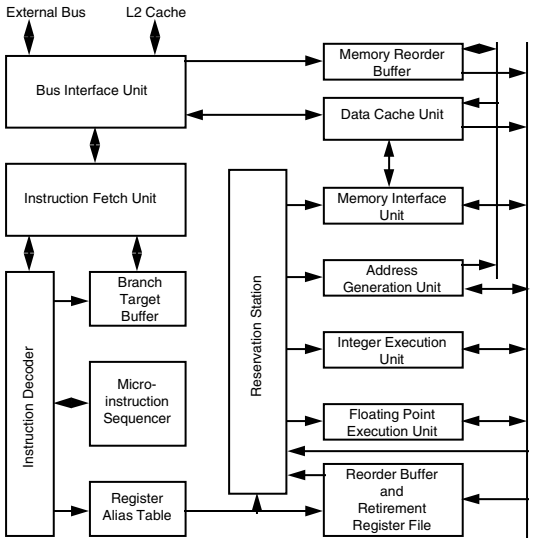

Figure 2 Pentium® Pro Processor Block diagram

The Intel Pentium® Pro processor implements dynamic execution using an out-of-order, speculative execution engine, with register renaming of integer, floating point and flags variables, multiprocessing bus support, and carefully controlled memory access reordering. The flow of Intel Architecture instructions is predicted and these instructions are decoded into micro-operations (uops), or series of uops. These uops are register- renamed, placed into an out-of-order speculative pool of pending operations, executed in dataflow order (when operands are ready), and retired to permanent machine state in source program order. This is accomplished with one general mechanism to handle unexpected asynchronous events such as mispredicted branches, instruction faults and traps, and external interrupts. Dynamic execution, or the combination of branch prediction, speculation and micro-dataflow, is the key to its high performance [Colwell95].

Figure 2 shows a block diagram of the processor. The basic operation of the microarchitecture is as follows:

1. The 512 entry Branch Target Buffer (BTB) helps the Instruction Fetch Unit (IFU) choose an instruction cache line for the next instruction fetch. ICache line fetches are pipelined with a new instruction line fetch commencing on every CPU clock cycle.

2. Three parallel decoders (ID) convert up to 3 Intel Architecture instructions into multiple sets of uops each clock.

3. The sources and destinations of up to 3 uops are renamed every cycle to a set of 40 physical registers by the Register Alias Table (RAT), which eliminates register re-use artifacts, and are forwarded to the 20-entry Reservation Station (RS) and to the 40-entry ReOrder Buffer (ROB).

4. The renamed uops are queued in the RS where they wait for their source data - this can come from several places, including immediates, data bypassed from just-executed uops, data present in a ROB entry, and data residing in architectural registers (such as EAX).

5. The queued uops are dynamically executed according to their true data dependencies and execution unit availability (IEU, FEU, AGU). The order in which any given uops execute in time has no particular relationship to the order implied by the source program.

6. Memory operations are dispatched from the RS to the Address Generation Unit (AGU) and to the Memory Ordering Buffer (MOB). The MOB ensures that the proper memory access ordering rules are observed.

7. Once a uop has executed, and its destination data has been produced, that result data is forwarded to subsequent uops that need it, and the uop becomes a candidate for "retirement".

.

8. Retirement hardware in the ROB uses uop timestamps to reimpose the original program order on the uops as their results are committed to permanent architectural machine state in the Retirement Register File (RRF). This retirement process must observe not only the original program order, it must correctly handle interrupts and faults, and flush all or part of its state on detection of a mispredicted branch. When a uop is retired, the ROB writes that uop's result into the appropriate RRF entry and notifies the RAT of that retirement so that subsequent register renaming can be activated. Up to 3 uops can be retired per clock cycle

The Pentium Pro processor emphasizes clock frequency over pipeline length [Papworth96]. It implements a 14-stage pipeline, consisting of 3 separate segments. The in-order front end has 8 stages as shown in Table 3. The out-of-order core has 3 stages, and the in-order retirement logic has 3 stages.

The in-order front end consists of the IFU (Instruction Fetch Unit), BTB (Branch Target Buffer), ID (Instruction Decoder), MIS (Micro Instruction Sequencer), and RAT (Register Alias Table). The in-order front end or fetch/decode unit involves eight clock cycles. The first one identifies the next instruction pointer (IP) based on the branch target buffer. The next 2-1/2 clock cycles are the Icache access. The next 2-1/2 after that are the decode, including instruction alignment and the 4-1-1 decode logic (one instruction with up to 4 uops, and two single uop instructions). The decoder has a 6 uop queue at its output. In the next stage, up to 3 uops are register renamed and branch information is sent to the BTB. In the last stage, the uops are written into the reservation station and re-order buffer.

The next pipeline segment is the out-of-order dispatch/execute core. The reservation station has 5 ports, allowing up to 5 uops per cycle. One port is used for integer ALU and shifts, FP operations, integer multiply, and address calculation. The other ports are used for integer ALU, loads, store address, and store data operations respectively. A uop is dispatched if it has all its operands and the execution resource is available. It takes two cycles for the reservation station to correctly identify which micro-ops have all the operands and are ready to go, and then one or more cycle for the actual execution and the return of the results. For an integer op, say a register-to-register add, the execute phase is just one cycle. Floating point adds have a latency of 3 cycles, and a throughput of 1 per cycle. FP multiply has a latency of 5 cycles and a repetition rate of 1 every 2 cycles. Integer multiply has a latency of 4 cycles and a throughput of 1 every cycle. Loads have a latency of 3 cycles on a Dcache hit. FDIV is not pipelined; it takes 17 cycles for single, 32 cycles for double, and 37 cycles for extended precision. Once an execution unit has created its result, the result flows back to the reservation station to enable future micro-ops and also flows down into the reorder buffer to enable retirement. Each of these ports has its own writeback path back to the reservation station. There is a full cross bar between all those ports so that any returning result could be bypassed to any other unit for the next clock cycle.

The reservation unit reads the instruction pool in the ROB to find potential candidates for retirement and determines which of these candidates are next in the original program order. It can retire up to 3 uops per cycle. The retirement process takes three clock cycles. Part of that is to make sure that uops are retired only as a group. The retirement process has to make sure that if any uops of an instruction are retired, all of them are retired. Otherwise the machine would have inconsistent state if it happened to take an interrupt at the wrong moment.

Table 3 **Pentium® Pro Processor Pipeline**

	In-order Fetch/Decode Unit
Stage I1	Send Instruction Pointer address to ICache
Stage I2	Instruction Fetch 1
Stage I3	Instruction Fetch 2
Stage I4	Align instruction buffer
Stage I5	Instruction Decode 1
Stage I6	Instruction Decode 2
Stage I7	Register renaming, branch info to BTB
Stage I8	Write Reservation Station.
	Out-of-order Dispatch/Execute Unit
Stage O1	Select reservation station entry
Stage O2	Dispatch uops
Stage O3	Execute uop and write result
	In-order Retire Unit
Stage R1	Retirement Stage 1
Stage R2	Retirement Stage 2
Stage R3	Retirement Stage 3

The processor includes separate data and instruction L1 caches (each of which is 8KB). The instruction cache is 4-way set associative, and the data cache is dual ported, non-blocking, 2-way set associative supporting one load and one store operation per

.

cycle. Both instruction and data cache line sizes are 32 byte wide. The MOB allows loads to pass stores, and loads to pass loads.

The secondary cache (L2 cache), which can be either 256KB or 512KB in size, is located on a separate die (but within the same package). The L2 cache is 4-way set associative unified non-blocking cache for storage of both instructions and data. It is closely coupled with a dedicated 64-bit full clock-speed backside cache bus. The L2 cache line is also 32 bytes wide. The L2 cache fills the L1 cache in a full frequency 4-1-1-1 transfer burst transaction. The processor connects to I/O and memory via a separate 64-bit bus that operates at 1/2, 2/5, 1/3, 2/7 or 1/4 of the CPU clock speed up to a maximum of 66 MHz. The multiprocessor system bus implements a pipelined demultiplexed design with up to 8 outstanding bus transactions. A processor can have up to 4 outstanding bus transactions.

CPU Performance

Figure 3 shows the relative performance of the two chips on the SPEC92 benchmark suite. All performance measurements were done on the Digital AlphaServer 8400 5/300 [Bhandarkar95] and the Intel Alder platform. The two systems are not comparably priced; the higher priced RISC system has the benefit of a larger cache and a higher bandwidth bus.

On the SPEC92 suite, the RISC system is 16% to 53% faster than the CISC system on the integer benchmarks, with a 39% higher SPECint92 rating. On the floating point benchmarks, the RISC system is 72% to 261% faster, with a 133% higher SPECfp92 rating. On the SPEC95 suite, the Alpha 21164 is 5% to 68% faster on the integer benchmarks with a 22% higher SPECint95 rating; and 53% to 200% faster on FP benchmarks with a 128% higher SPECfp95 rating as shown in Figure 4.

The numbers reflect the difference in the FP hardware of the two processors. The fully pipelined 300 MHz Alpha, with its separate FP add and multiply units, has a peak floating point power of 600 megaflops. The Pentium® Pro processor can issue only one FP operation every cycle for a peak megaflops rate of 150. On integer benchmarks, the Pentium Pro processor is much closer because its advanced branch prediction scheme and out of order execution.

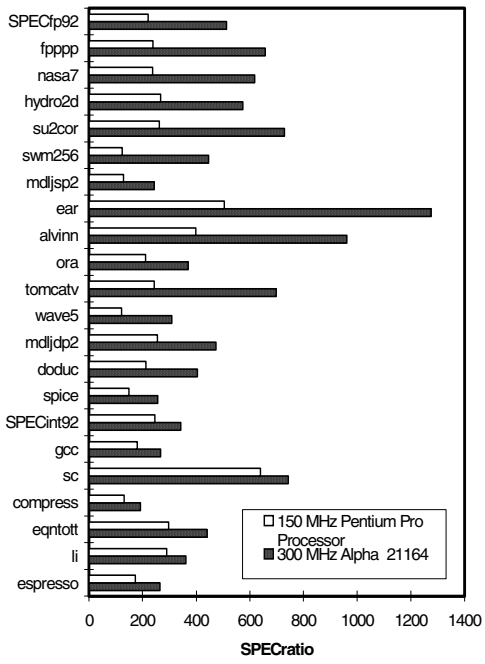

Figure 3 SPEC92 Performance Comparison

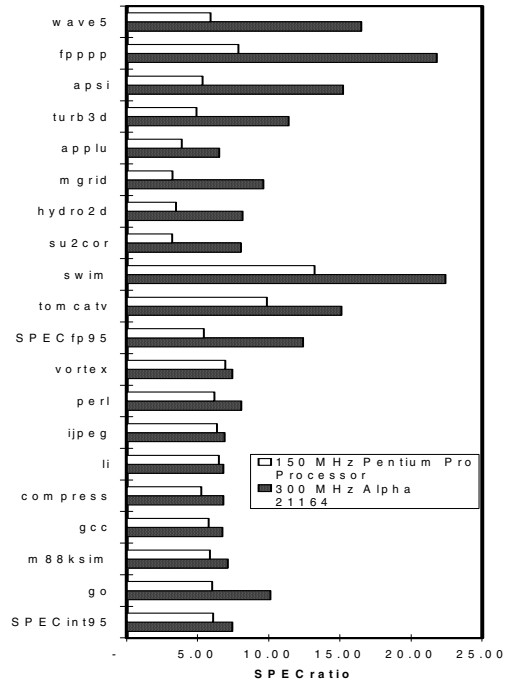

Figure 4 **SPEC95 Performance Comparison**

For CPU benchmarks, performance is measured in terms of execution time. Given that,
execution time = path length * CPI * cycle time

.

the performance ratio[1] can be expressed as

$$\frac{\text{PPro Execution Time}}{\text{Alpha Execution Time}} = \frac{\text{PPro Path Length}}{\text{Alpha Path Length}} \times \frac{\text{PPro CPI}}{\text{Alpha CPI}} \times \frac{\text{Alpha MHz}}{\text{PPro MHz}}$$

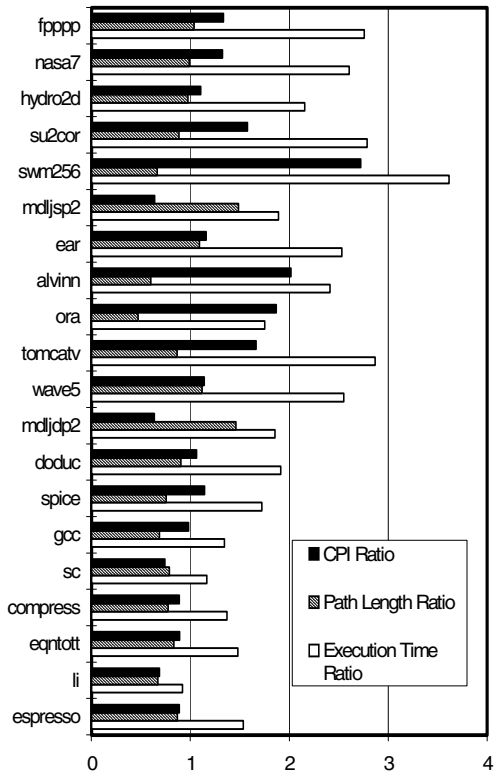

Figure 5 Elements of Performance Ratio

Figure 5 shows the ratios for the execution time, path length, and CPI for the SPEC92 benchmarks. The MHz ratio is 2. Benchmarks with a performance ratio greater than 2 indicate that the product of the path length ratio and CPI ratio exceeds 1. For all integer benchmarks, the Intel architecture has a 13 to 33% lower path length. On the FP benchmarks, the Intel path ranges from 50% lower to 50% higher. Intel architecture path length is negatively impacted by the register stack architecture used for FP operations. On ora, the Intel architecture shows the biggest path length advantage. This is mostly because the Alpha architecture does not implement SQRT in hardware. Pure CPI comparisons can be misleading, because CPI can be lowered by a higher path length. For this reason, we will avoid looking at per instruction statistics for the rest of this study, and focus instead on per operation or per benchmark statistics.

[1] All performance ratios in this section are shown so that >1 means that Alpha is faster.

Figure 6 shows the performance of a 150 MHz Pentium Pro processor based Digital Celebris 6150 compared to a 300 MHz Alpha 21164 processor based AlphaStation 600 on the SYSmark* for Windows NT suite from BAPCO, which contains project management software (Welcom Software Technology Texim* Project 2.0e), computer-aided PCB design tool (Orcad MaxEDA* 6.0) and Microsoft Office* applications for word processing (Word* 6.0), presentation graphics (PowerPoint* 4.0), and spreadsheets (Excel 5.0). While the SPEC95 benchmarks were optimized for each processor using the latest compilers, the SYSmark/NT benchmarks are based on old binaries that have been shipping for many years and were probably not generated with all optimizations turned on. Powerpoint uses 16-bit Intel Architecture binaries that are emulated on Alpha, except for calls to the Windows subsystem that are redirected to native Alpha code in the operating system. Overall, the Alpha 21164 is only 6% faster than the Pentium Pro processor.

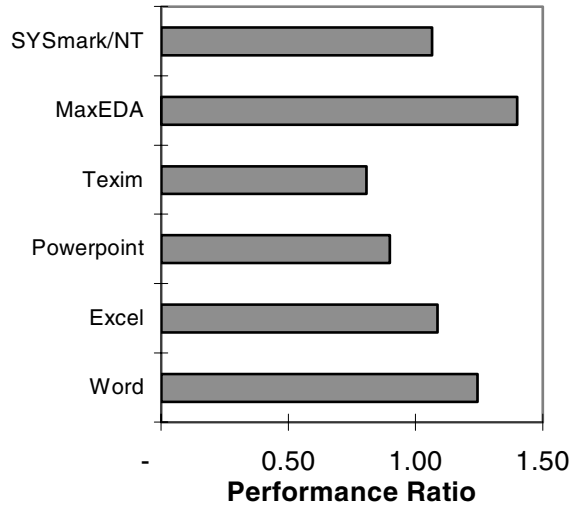

Figure 6 SYSmark/NT Performance Ratio

SPEC92 Characterization

The rest of this paper will explore some of the detailed performance characteristics of the two processors using the SPEC92 benchmark suite. Even though the SPEC92 suite is now obsolete, it is used here because detailed performance characteristics were available on this suite for the Alpha 21164 processor [Cvetanovic96]. A detailed characterization of the Pentium Pro processor using SPEC95 and

.

SYSmark/NT can be found in another recent publication [Bhandarkar97].

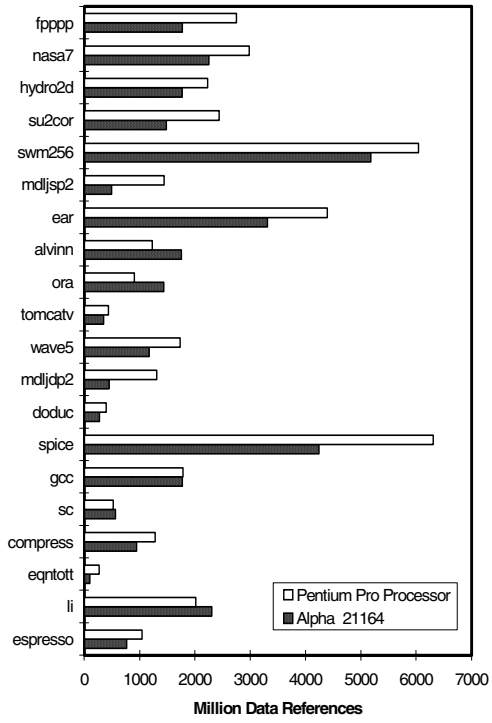

Figure 7 Data References

Data references

The Alpha architecture provides 32 integer and 32 FP registers. The Intel architecture has only eight of each. This results in more memory accesses on the Intel architecture as shown in ***Figure 7***.

Dcache misses

The Alpha 21164 implements an 8KB L1 Dcache. It is a write-through, read-allocate, direct-mapped, physical cache with 32-byte blocks. The load latency for Dcache hits is 2 cycles, and 8 cycles for a Dcache miss with a L2 cache hit. There can be up to 6 cache line misses outstanding.

The Pentium® Pro processor also implements an 8KB L1 Dcache, but it is 2-way set associative with a line size of 32 bytes. The load latency is 3 cycles for a Dcache hit, and another 4 cycles for a L2 cache hit. Up to 4 Dcache misses can be queued at the L2 cache. The Pentium Pro processor has fewer cache misses as shown in Figure ***8***. Inspite of having more data memory accesses, the Pentium Pro processor has fewer accesses to the second level cache. Data that would be kept in registers on Alpha is accessed directly from memory thereby increasing cache hits. The 2-way cache design also lowers cache misses over a direct mapped design.

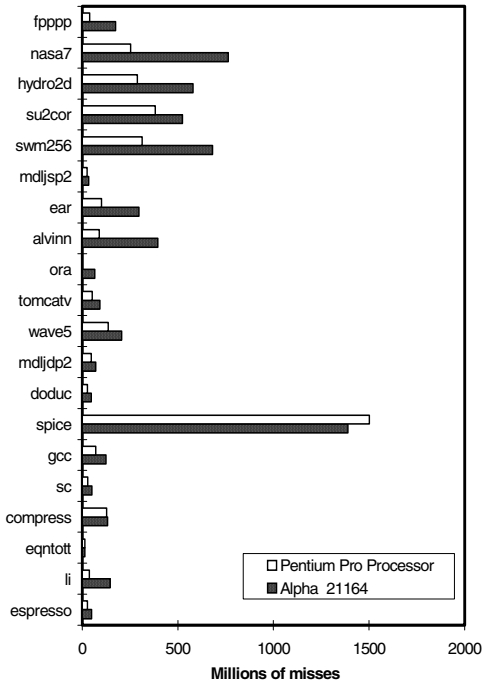

Figure 8 Data Cache Misses

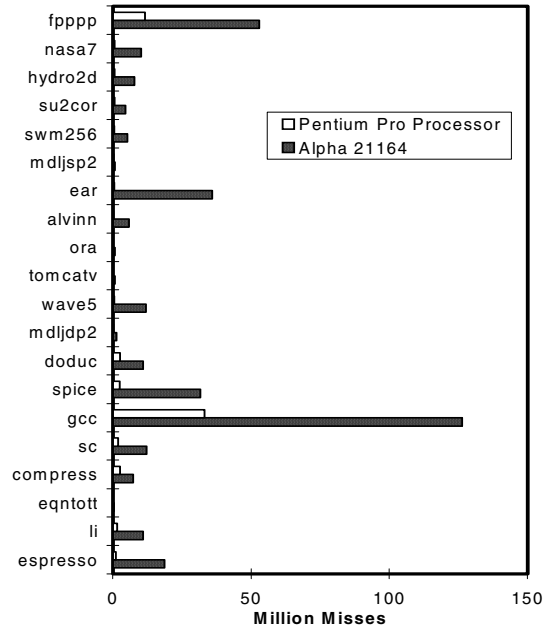

Figure 9 Instruction Cache Misses

.

Icache misses

The instruction cache (Icache) in the Alpha 21164 is an 8KB, virtual, direct-mapped cache with 32-byte blocks. On the Pentium Pro processor, the instruction cache is also 8KB, but it is 4-way set associative with a line size of 32 bytes. Icache misses are also lower on the Pentium Pro processor as shown in Figure 9. The 4-way design and shorter instruction lengths benefit the Pentium Pro processor.

Higher level cache misses

The Alpha 21164 has a on-chip second-level cache. It is a 96KB, 3-way set associative, physical, write-back, write-allocate cache with 32- or 64-byte blocks. It is a mixed data and instruction cache. A maximum of two second-level cache misses can be queued for external access to the off-chip cache and memory. The processor implements control for an optional, external, direct-mapped, physical, write-back, write-allocate cache with 32- or 64-byte blocks. The 21164 system supported a third-level off-chip cache of 4 megabytes. The external read and write timing was 6 cycles and 5 cycles.

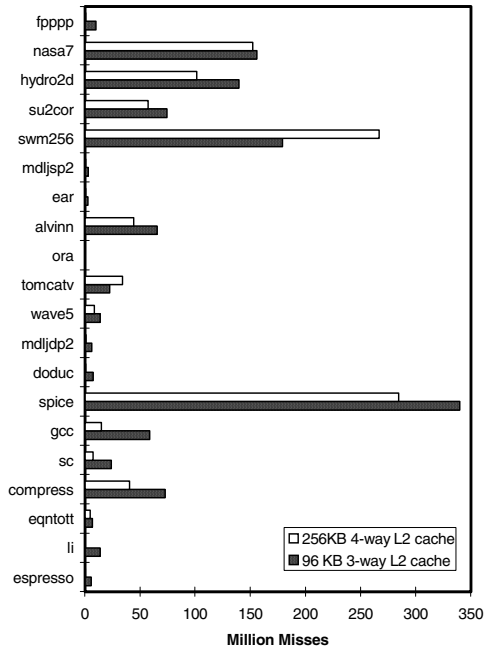

Figure 10 Second Level Cache Misses

The secondary cache (L2 cache) on the Pentium® Pro processor can be either 256KB or 512KB in size, is located on a separate die (but within the same package). The L2 cache is 4-way set associative unified non-blocking cache for storage of both instructions and data. It is closely coupled with a dedicated 64-bit full clock-speed backside cache bus. The L2 cache line is also 32 bytes wide. The L2 cache fills the L1 cache in a full frequency 4-1-1-1 transfer burst transaction.

The larger L2 cache on the Pentium Pro processor does result in significantly fewer cache misses as shown in Figure 10. One exception is swm256, where the Alpha compiler transforms loop accesses into blocks that reuse cached data more efficiently.

Branches

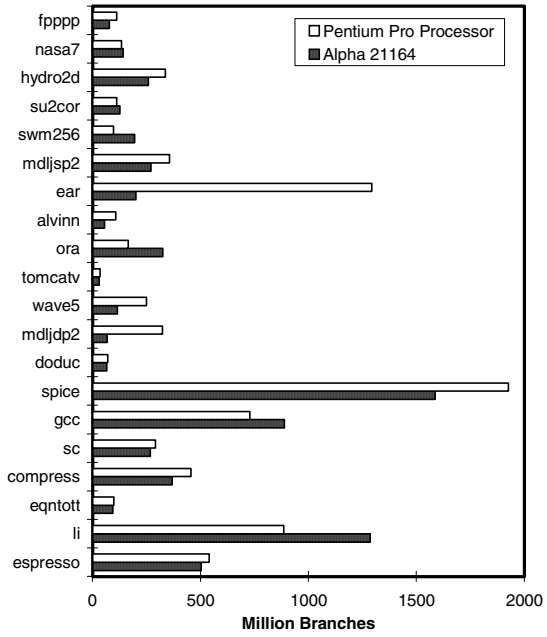

Figure 11 Branch Instructions

The Alpha architecture has fewer branches for most of the benchmarks, especially floating point, as shown in Figure 11. The Alpha compiler can do use the 32 available registers and do loop unrolling, thereby reducing branches.

Branch mispredicts

The 21164 uses a 2048 entry BTB with a 2-bit history per entry. The penalty for mispredicted branches is 5 cycles. The Pentium® Pro processor implements a novel branch prediction scheme, derived from the two-level adaptive scheme of Yeh and Patt [Yeh91]. The BTB contains 512 entries with a 4-bit history per entry. The penalty for mispredicted branches is about 10-15 cycles. Given the factor of 2 in clock speed, the time penalty for a mispredicted branch is 4 to 6 times higher on the Pentium Pro processor.

.

The Pentium Pro processor has a lower branch mispredict ratio, despite having fewer BTB entries, due to the more sophisticated 2-level adaptive branch prediction scheme. The lower mispredict ratio also results in fewer mispredicted branches, inspite of higher number of branch instructions. These branch predict statistics are shown in Figure 12 and Figure 13.

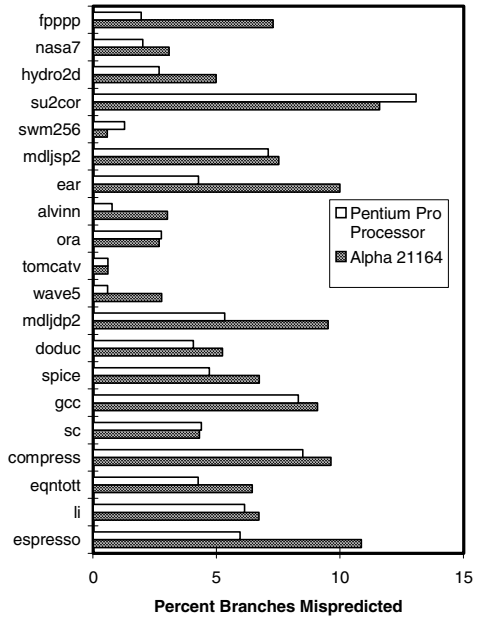

Figure 12 Branch Mispredict Ratio

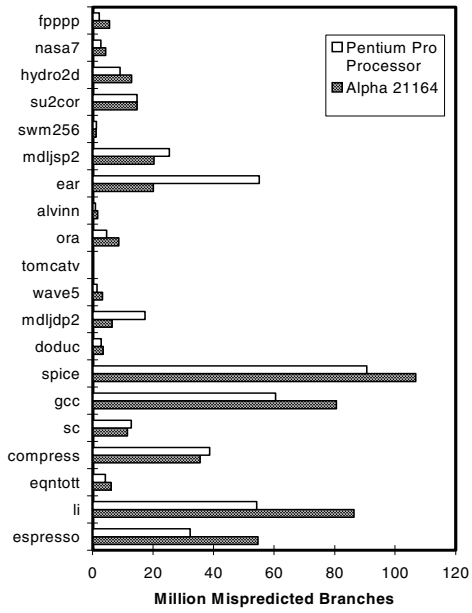

Figure 13 Number of Mispredicted Branches

TLB misses

The Alpha 21164 includes a 48-entry, fully associative instruction translation buffer (ITLB). The standard page size is 8 KB, but larger pages are also supported. The Pentium® Pro processor has separate TLBs for 4-Kbyte and 4-Mbyte page sizes. The ITLB for 4KB pages has 32 entries and is 4 way set associative. The ITLB for large pages has 4 entries, and it is fully associative. The SPEC92 benchmarks do not incur many ITLB misses.

The Alpha 21164 includes a 64-entry, fully associative, data translation buffer (DTLB). Each entry permits translation for up to 512 contiguously mapped, 8K-byte pages, using a single DTB entry. On the Pentium Pro processor, the DTLB for 4KB pages has 64 entries and it is 4 way set associative. The DTLB for large pages has 8 entries; and it is 4-way set associative. Even though the DTLB misses are somewhat higher than ITLB misses, they are not a major factor in determining overall performance for the SPEC92 benchmarks.

uops vs. RISC instructions

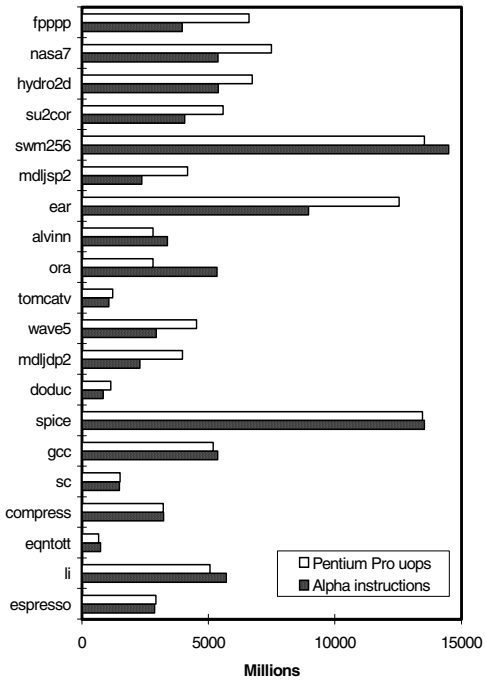

Figure 14 Micro-operations vs. RISC instructions

The Pentium® Pro processor converts CISC instruction on the fly into simple RISC-like micro-operations. Since these uops are generated on a per-instruction basis in hardware, one would expect more

.

uops than a compiler could generate. Figure 14 shows the uop count versus the RISC instructions. For integer benchmarks and spice (lowest FP content), the uops are fairly close to RISC instructions. On FP benchmarks, in most cases there are fewer RISC instructions. One notable exception is ora, where Alpha has to generate multiple instructions to calculate SQRT.

Accounting for all cycles

It is interesting to analyze where all the time is spent. The Alpha 21164 is an in-order execution machine with non-blocking caches. Some cache misses can be overlapped with the execution of subsequent instructions until a data dependency is encountered. Figure 15 shows where all the cycles are spent - issuing 1, 2, 3, or 4 instructions, or being stalled. Stalls are separated into either dry (Istream stalls and branch mispredicts) or frozen (Dstream stalls, register conflicts, execution units busy). About 30 to 50% of the time is spent in stalls on the SPEC integer benchmarks, 45 to 63% on most SPEC FP benchmarks, and 80% on the debit-credit transaction processing workload.

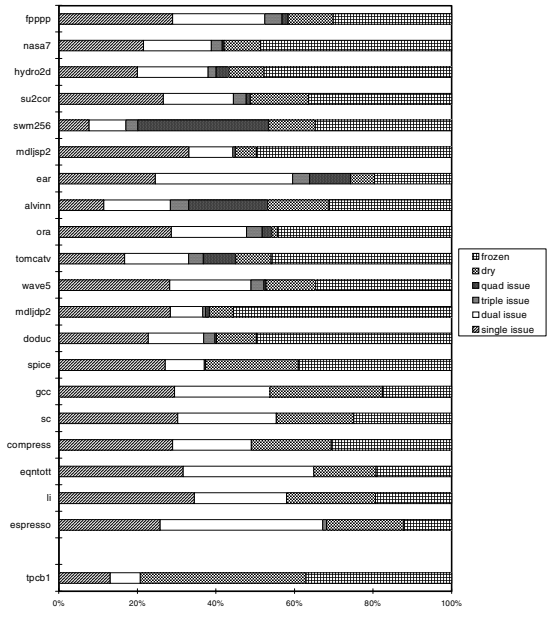

Figure 15 Issue and Stall cycles for Alpha 21164

Accounting for cycles in an out-of-order machine like the Pentium® Pro processor is more difficult due to all the overlapped execution. It is still useful to examine the various components of execution and stalls and compare them to the actual cycles per instruction as shown in Figure 16. The CPI is generally much lower than the individual components due to overlapped execution. The figure also shows resource stall cycles in which some resource such as execution unit or buffer entry is not available. Execution can proceed during a resource stall cycle in some other part of the machine.

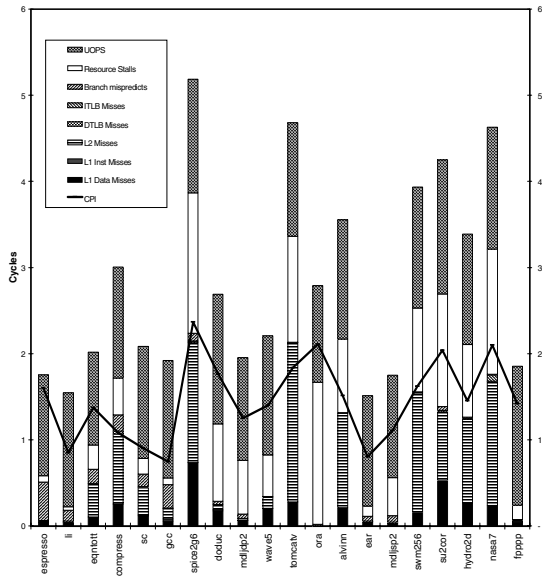

Figure 16 CPI vs. Stalls for Pentium® Pro processor

OLTP Performance

We have presented CPU performance comparisons for the SPEC92 benchmarks. Most of these benchmarks show good locality of instruction and data references. Many commercial workloads exhibit very different memory access patterns that result in more cache misses and significant stall times for memory as shown in Figure 15. On Line Transaction Processing (OLTP) benchmarks like TPC-C model I/O intensive environments which stress the memory systems of machines much more thoroughly. TPC-C simulates a complete computing environment where a population of terminal operators executes transactions against a database. The benchmark is centered around the principal activities (transactions) of an order-entry environment. These transactions include entering and delivering orders, recording payments, checking the status of orders, and monitoring the level of stock at the warehouses.

No measured data is available for the chips we have studied in this paper, but there is data on derivatives of both chips, which are essentially shrinks to the next generation 0.35µ technology running at a 33% faster clock rate. Table 4 shows the highest reported TPC-C performance (measured in transactions per minute) of quad processor systems, based on the 400 MHz

.

Alpha 21164 (12.1 SPECint95) and 200 MHz Pentium® Pro (8.71 SPECint95) processors, circa October 1996. The results show that while the Alpha system is 45% faster on SPECint_rate95, it is 8% slower on the TPC-C benchmark with a 59% higher $/tpmC using the same database software!

Table 4 TPC-C Performance

	Compaq ProLiant 5000 Model 6/200	***Digital AlphaServer 4100 5/400***
CPUs	Four 200 MHz Pentium Pro processors	Four 400 MHz Alpha 21164 processors
L2 cache	512KB	4 MB
SPECint_rate95	292 (est)[2]	422
TPC-C perf	**8311 tpmC @ $95.32/tpmC**	**7598 tpmC @ $152.04/tpmC**
Operating Sys	SCO UnixWare	Digital UNIX
Database	Sybase SQL Server 11.0	Sybase SQL Server 11.0

Concluding Remarks

Studies like this one offer some insight into the performance characteristics of different instruction set architectures and attempts to implement them well in a comparable technology. The overall performance is affected by many factors and strict cause-effect relationships are hard to pinpoint. Such explorations are also hindered by the lack of measured data on common workloads for systems designed by different companies. This study would have been more meaningful if more stressful environments like on-line transaction processing and computer aided design could have been analyzed in detail. Nevertheless, it does provide new quantitative data, that can be used to get a better understanding of the performance differences between a premier RISC and CISC implementation.

Using a comparable die size, the Pentium® Pro processor achieves 80 to 90% of the performance of the Alpha 21164 on integer benchmarks and transaction processing workloads. It uses an aggressive out-of-order design to overcome the instruction set level limitations of a CISC architecture. On floating-point intensive benchmarks, the Alpha 21164 does achieve over twice the performance of the Pentium Pro processor.

[2] measured result for Fujitsu ICL Superserver J654i using the same processor.

Acknowledgments

The author is grateful to Jeff Reilly and Mason Guy of Intel for collecting the performance counter measurement data for the Pentium® Pro processor, and Zarka Cvetanovic of Digital Equipment Corporation for providing the performance counter measurement data for the Alpha 21164.

References

[Bannon95] P. Bannon and J. Keller, "Internal Architecture of Alpha 21164 Microprocessor", Proc. Compcon Spring 95, Mar 1995.

[Bhandarkar91] D. Bhandarkar and D. Clark, "Performance from Architecture: Comparing a RISC and a CISC with Similar Hardware Organization," Proceedings of ASPLOS-IV, April 1991.

[Bhandarkar95] D. Bhandarkar, "Alpha Implementations and Architecture: Complete Reference and Guide", 1995, ISBN: 1-55558-130-7, Digital Press, Newton, MA.

[Bhandarkar97] D. Bhandarkar and J. Ding, "Performance Characterization of the Pentium Pro Processor," Proceedings of HPCA-3, February 1997.

[Colwell95] R. Colwell and R. Steck, "A 0.6um BiCMOS Processor with Dynamic Execution", ISSCC Proceedings, pp 176-177, February 1995.

[Cvetanovic96] Z. Cvetanovic and D. Bhandarkar, "Performance Characterization of the Alpha 21164 Microprocessor using TP and SPEC Workloads," Proceedings of HPCA-2, February 1996.

[Edmondson95] J. Edmondson et al, "Superscalar Instruction Execution in the 21164 Microprocessor", IEEE Micro, April 1995, pp.33-43.

[Papworth96] D. Papworth, "Tuning The Pentium® Pro Microarchitecture," IEEE Micro, April 1996, pp. 8-15.

[Yeh91] Tse-Yu Yeh and Yale Patt, "Two-Level Adaptive Training Branch Prediction," Proc. IEEE Micro-24, Nov 1991, pp. 51-61.