An Assignment
On

Pntium® 4 Processor

Course Title: Microprocessor
Course Code: CSE – 311

© Moshiur Rahman Khan
CSE 1st Batch, PSTU.

Submitted To

Md. Ileas Pramanik
Lecturer,
CSE Faculty, PSTU.

Submitted By

Md.Moshiur Rahman Khan
Reg No.: 00600
Roll No.: 21
L-IV, S- I
mehedi.cse@gmail.com
CSE, PSTU.

Submission Date: 12-10-2008

Signature of the Course Teacher:

Patuakhali Science & Technology University, Patuakhali

Contents

Abstract

Developing a new leading-edge Intel ® Architecture microprocessor is an immensely complicated undertaking. The microarchitecture of the Pentium® 4 processor is significantly more complex than any previous Intel Architecture microprocessor, so the challenge of validating the logical correctness of the design in a timely fashion was indeed a daunting one.

Introduction to Pentium® 4 Processor

The **Pentium® 4 processor** is Intel's new flagship microprocessor that was introduced at 1.5GHz in November of 2000. It implements the new Intel Net Burst micro architecture that features significantly higher clock rates and world-class performance. It includes several important new features and innovations that will allow the Intel **Pentium® 4 processor** to deliver industry-leading performance for the next several years.

The **Pentium® 4 Processor** is Intel's most advanced IA-32 microprocessor, incorporating a host of new micro architectural features including a 400MHz system bus, hyper pipelined technology, advanced dynamic execution, rapid execution engine, advanced transfer cache, execution trace cache, and Streaming Single Instruction, Multiple Data (SIMD) Extensions 2 (SSE2).

The Microarchitecture of the Pentium® 4 Processor

The **Pentium® 4 processor** is designed to deliver performance across applications where end users can truly appreciate and experience its performance. For example, it allows a much better user experience in areas such as Internet audio and streaming video, image processing, video content creation, speech recognition, 3D applications and games, multi-media, and multi-tasking user environments.

The **Pentium® 4 processor** enables real-time MPEG2 video encoding and near real-time MPEG4 encoding, allowing efficient video editing and video conferencing. It delivers world-class performance on 3D applications and games, such as Quake 3□, enabling a new level of realism and visual quality to 3D applications.

The **Pentium® 4 processor** has 42 million transistors implemented on Intel's 0.18u CMOS process; with six levels of aluminum interconnect. It has a die size of 217 mm2 and it consumes 55 watts of power at 1.5GHz. Its 3.2 GB/second system bus helps provide the high data bandwidths needed to supply data to today's and tomorrow's demanding applications. It adds 144 new 128-bit single Instruction Multiple Data (SIMD) instructions called SSE2 (Streaming SIMD Extension 2) that improve performance for multi-media, content creation, scientific, and engineering applications.

Overview of the NETBURST™ Microarchitechture

A fast processor requires balancing and tuning of many micro architectural features that compete for processor die cost and for design and validation efforts. **Figure 1** shows the basic Intel Net Burst micro architecture of the **Pentium 4 processor**.

There are four main sections:

(I) The in-order front end,
(II) The out-of-order execution engine,
(III) The integer and floating-point execution units, and
(IV) The memory subsystem.

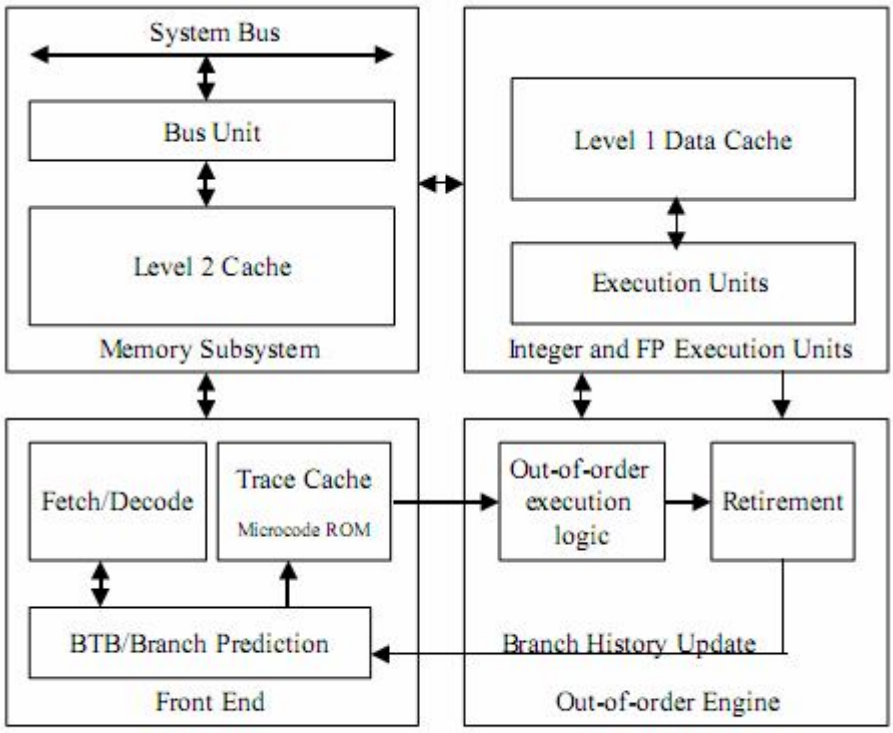


Figure 1: Basic block diagram

In-Order Front End

The in-order front end is the part of the machine that fetches the instructions to be executed next in the program and prepares them to be used later in the machine pipeline. The front end has highly accurate branch prediction logic that uses the past history of program execution to speculate where the program is going to execute next. The predicted instruction address, from this front-end branch prediction logic, is used to fetch instruction bytes from the Level 2 (L2) cache.

The Net Burst micro architecture has an advanced form of a Level 1 (L1) instruction cache called the Execution Trace Cache. Unlike conventional instruction caches, the Trace Cache sits between the instructions decode logic and the execution core as shown in **Figure 1**.

Out-of-Order Execution Logic

The out-of-order execution engine is where the instructions are prepared for execution. The out-of-order execution logic has several buffers that it uses to smooth and re-order the flow of instructions to optimize performance as they go down the pipeline and get scheduled for execution. This out-of-order execution allows instructions in the program following delayed instructions to proceed around them as long as they do not depend on those delayed instructions. Out-of-order execution allows the execution resources such as the ALUs and the cache to be kept as busy as possible executing independent instructions that are ready to execute. The retirement logic is what reorders the instructions, executed in an out-of-order manner, back to the original program order. The **Pentium 4 processor** can retire up to three uops per clock cycle.

Integer and Floating-Point Execution Units

The execution units are where the instructions are actually executed. This section includes the register files that store the integer and floating-point data operand values that the instructions need to execute. The execution units include several types of integer and floating-point execution units that compute the results and also the L1 data cache that is used for most load and store operations.

Memory Subsystem

Figure 1 also shows the memory subsystem. This includes the L2 cache and the system bus. The L2 cache stores both instructions and data that cannot fit in the Execution Trace Cache and the L1 data cache. The external system bus is connected to the backside of the second-level cache and is used to access main memory when the L2 cache has a cache miss, and to access the system I/O resources.

Clock Rates

Processor micro architectures can be pipelined to different degrees. The degree of pipelining is a micro architectural decision. The final frequency of a specific processor pipeline on a given silicon process technology depends heavily on how deeply the processor is pipelined. When designing a new processor, a key design decision is the target design frequency of operation. The frequency target determines how many gates of logic can be included per pipeline stage in the design. This then helps determine how many pipeline stages there are in the machine.

Historical Trend of Processor Frequencies

Figure 2 shows the relative clock frequency of Intel's last six processor cores. The vertical axis shows the relative clock frequency, and the horizontal axis shows the various processors relative to each other.

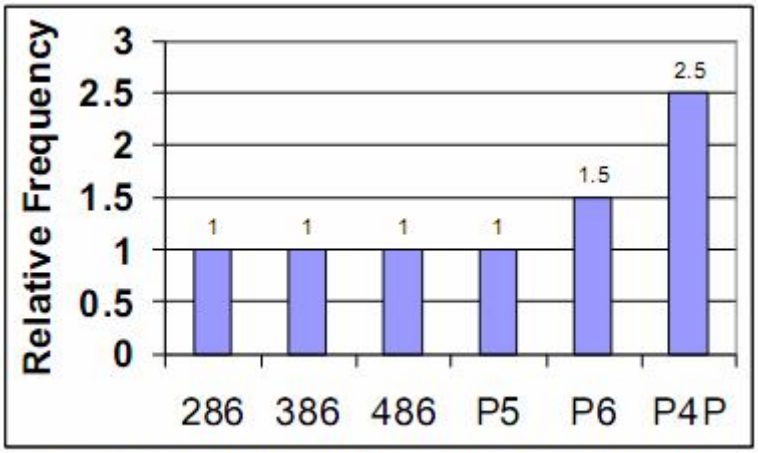


Figure 2: Relative frequencies of Intel's processors

Figure 2 shows that the 286, Intel386™, Intel486™ and Pentium® (P5) processors had similar pipeline depths–they would run at similar clock rates if they were all implemented on the same silicon process technology. They all have a similar number of gates of logic per clock cycle. The P6 micro architecture lengthened the processor pipelines, allowing fewer gates of logic per pipeline stage, which delivered significantly higher frequency and performance. The P6 micro architecture approximately doubled the number of pipeline stages compared to the earlier processors and was able to achieve about a 1.5 times higher frequency on the same process technology.

At its introduction in November 2000, the Pentium 4 processor was at 1.5 times the frequency of the Pentium III processor. Over time this frequency delta will increase as the Pentium 4 processor design matures.

Different parts of the **Pentium 4 processor** run at different clock frequencies. As an example of the pipelining differences, **Figure 3** shows a key pipeline in both the P6 and the **Pentium 4 processors**: the mispredicted branch pipeline. This pipeline covers the cycles it takes a processor to recover from a branch that went a different direction than the early fetch hardware predicted at the beginning of the machine pipeline. As shown, the **Pentium 4 processor** has a 20-stage misprediction pipeline while the P6 microarchitecture has a 10-stage misprediction pipeline. By dividing the pipeline into smaller pieces, doing less work during each pipeline stage (fewer gates of logic), and the clock rate can be a lot higher.

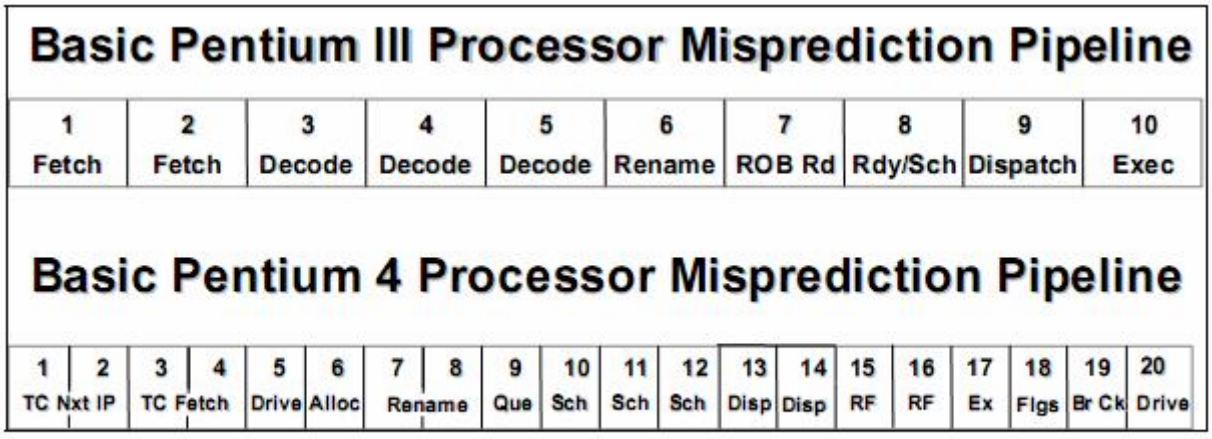


Figure 3: Misprediction Pipeline

NETBURST Microarchitechture

Figure 4 shows a more detailed block diagram of the NetBurst micro architecture of the **Pentium 4 processor**. The top-left portion of the diagram shows the front end of the machine. The middle of the diagram illustrates the out-of-order buffering logic, and the bottom of the diagram shows the integer and floating-point execution units and the L1 data cache. On the right of the diagram is the memory subsystem.

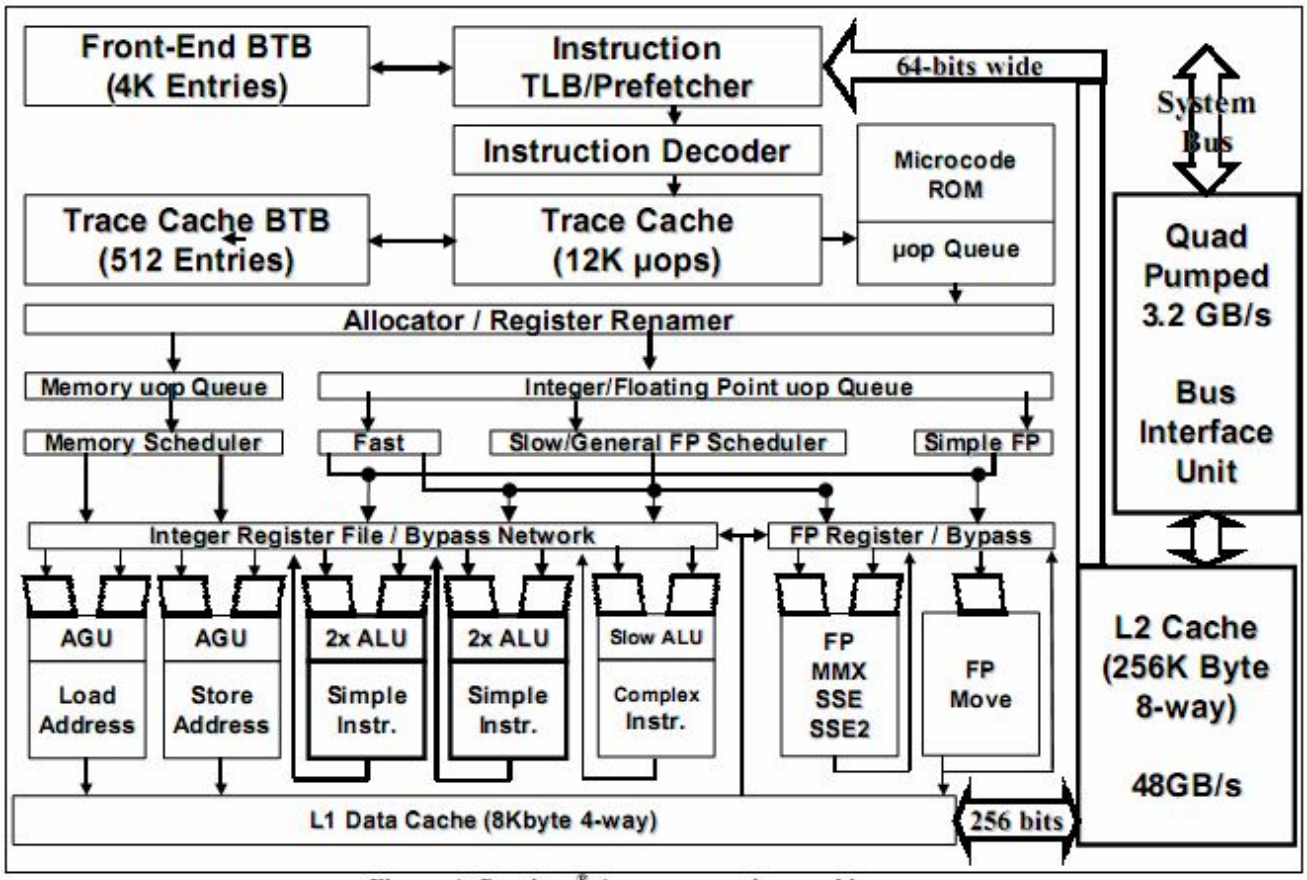


Figure 4: Pentium® 4 processor microarchitecture

Register Renaming

As shown in **Figure 5** the NetBurst microarchitecture allocates and renames the registers somewhat differently than the P6 microarchitecture. On the left of **Figure 5**, the P6 scheme is shown.

The NetBurst microarchitecture allocation scheme is shown on the right of **Figure 5**. It allocates the ROB entries and the result data Register File (RF) entries separately.

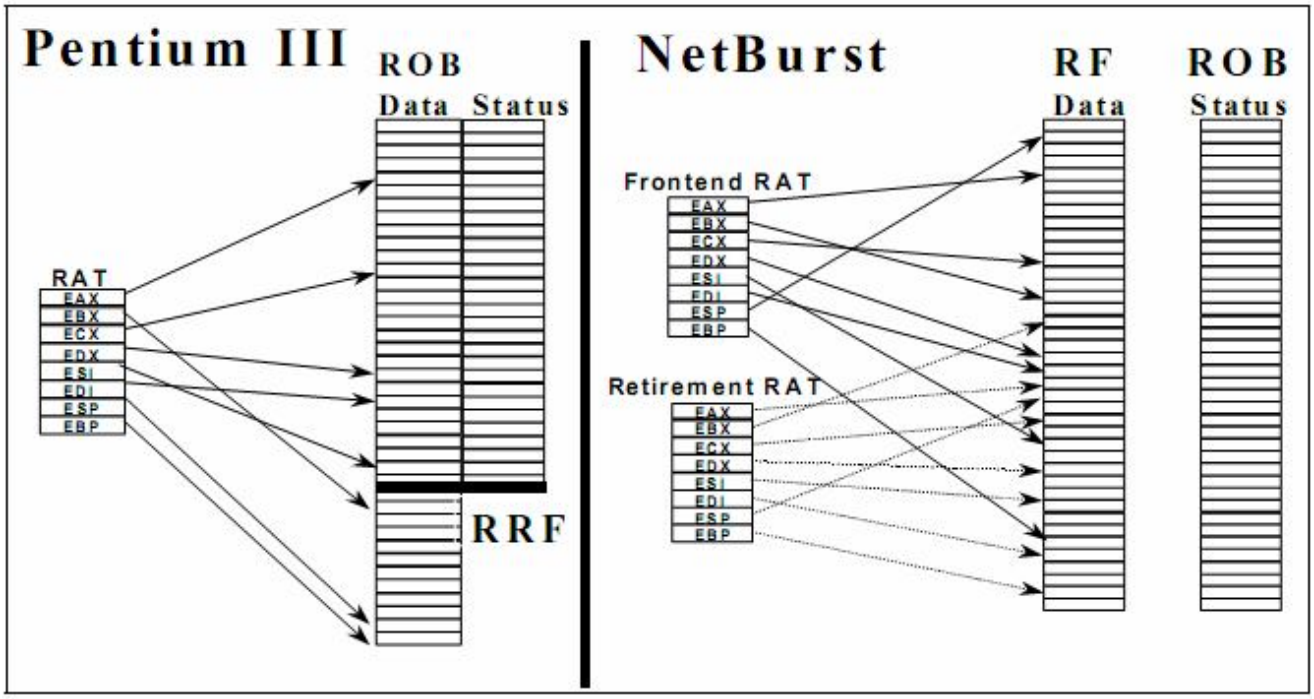


Figure 5: Pentium® III vs. Pentium® 4 processor register allocation

Uop Scheduling

The uop schedulers determine when a uop is ready to execute by tracking its input register operands. This is the heart of the out-of-order execution engine. There are two uop queues–one for memory operations (loads and stores) and one for non-memory operations. There are several individual uop schedulers that are used to schedule different types of uops for the various execution units on the **Pentium 4 processor** as shown in **Figure 6**.

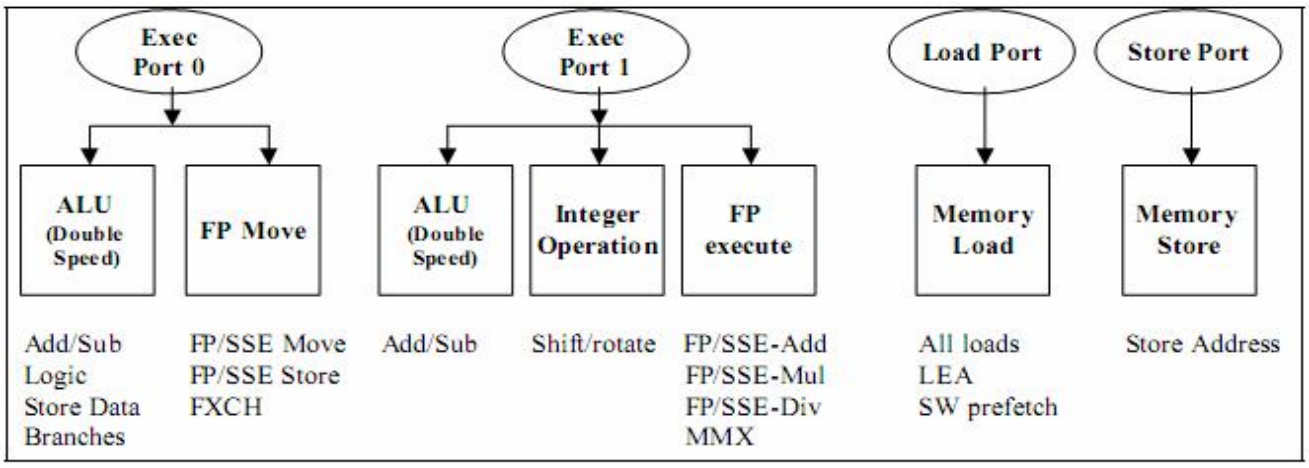


Figure 6: Dispatch ports in the Pentium® 4 processor

Low Latency Integer ALU

The **Pentium 4 processor** execution units are designed to optimize overall performance by handling the most common cases as fast as possible. The **Pentium 4 processor** can do fully dependent ALU operations at twice the main clock rate. Approximately **60-70%** of all uops in typical integer programs use this key integer ALU loop. Executing these operations at **½** the latency of the main clock helps speed up program execution for most programs.

The processor does ALU operations with an effective latency of one-half of a clock cycle. It does this operation in a sequence of three fast clock cycles (the fast clock runs at 2x the main clock rate) as shown in **Figure 7**.

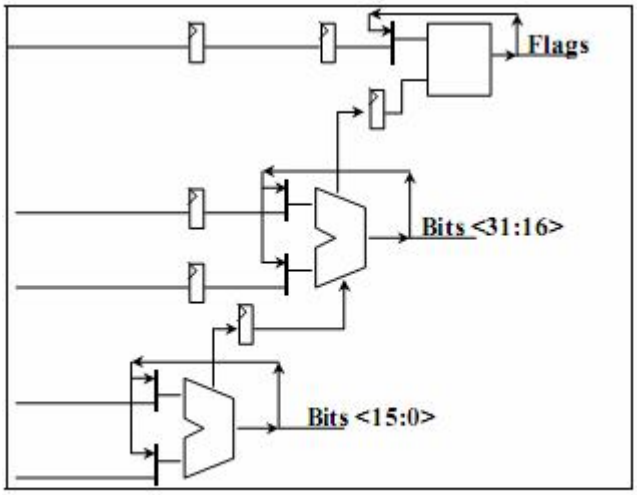


Figure 7: Staggered ALU add

FP/SSE Execution Units

The Floating-Point (FP) execution cluster of the **Pentium 4 processor** is where the floating-point, MMX, SSE, and SSE2 instructions are executed. Early in the development cycle of the **Pentium 4 processor**, we had two full FP/SSE execution units, but this cost a lot of hardware and did not buy very much performance for most FP/SSE applications. The high bandwidth system bus of the **Pentium 4 processor** allows this prefetching to help keep the execution engine well fed with streaming data.

Performance

The **Pentium 4 processor** delivers the highest SPECint_base performance of any processor in the world. It also delivers world-class SPECfp2000 performance. These are industry standard benchmarks that evaluate general integer and floating-point application performance.

Figure 8 shows the performance comparison of a **Pentium 4 processor** at 1.5GHz compared to a Pentium III processor at 1GHz for various applications. The integer applications are in the 15-20% performance gain while the FP and multi-media applications are in the 30-70% performance advantage range. For FSPEC 2000 the new SSE/SSE2 instructions buy about 5% performance gain compared to an x87-only version.

As the compiler improves over time the gain from these new instructions will increase. Also, as the relative frequency of the **Pentium 4 processor** increases over time (as its design matures), all these performance deltas will increase.

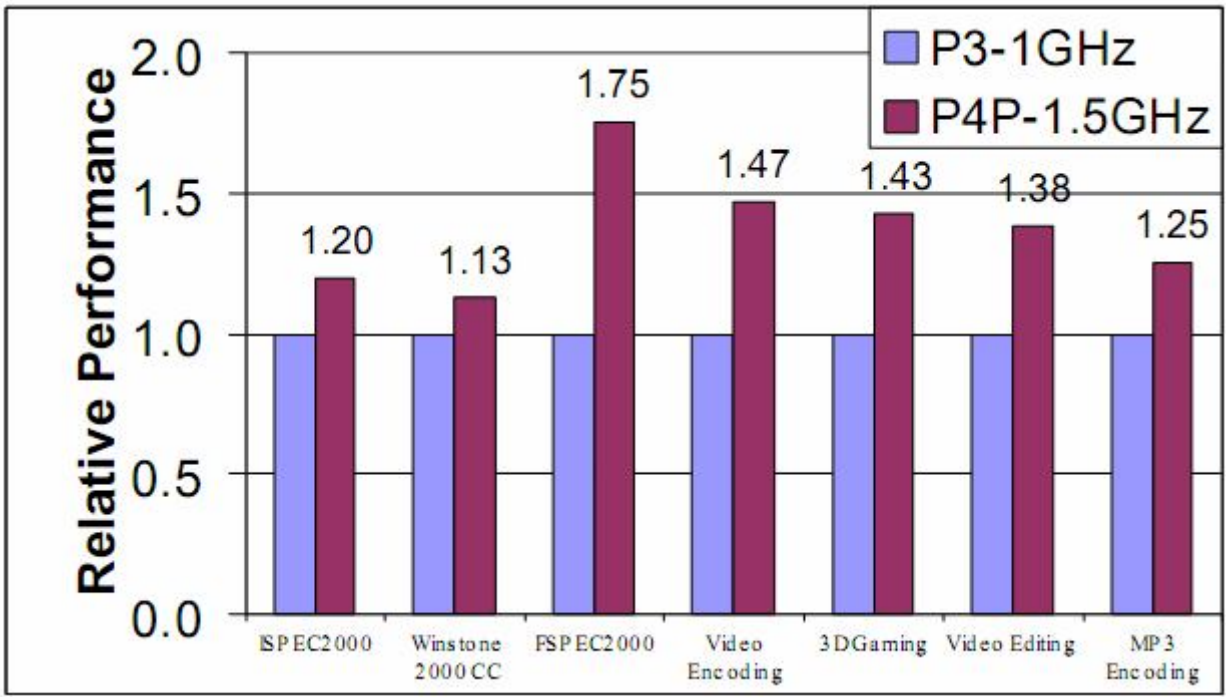


Figure 8: Performance comparison

Pre-Silicon Validation Challenges & Issues

The first thing that we had to do was build a validation team. Fortunately, we had a nucleus of people who had worked on the Pentium® Pro processor and who could do the initial planning for the **Pentium® 4 processor** while at the same time working with the architects and designers who were refining the basic microarchitectural concepts. However, this investment paid off handsomely over the next few years as the team matured into a highly effective bug-finding machine that found more than 60% of all the logic bugs that were filed prior to tapeout. In doing so, they developed an in-depth knowledge of the **Pentium 4 processor's** NetBurst™ microarchitecture that has proved to be invaluable in post-silicon logic and speedpath debug and also in fault grade test writing.

Formal Verification

The **Pentium 4 processor** was the first project of its kind at Intel to apply FV on a large scale. They focused on the floating-point execution units and the instruction decodes logic. Because these areas had been sources of bugs in the past that escaped early detection, using FV allowed us to apply this technology to some real problems with real payback.

Two of these bugs were classic floating-point data space problems:

1. The FADD instruction had a bug where, for a specific combination of source operands, the 72-bit FP adder was setting the carryout bit to 1 when there was no actual carryout.

2. The FMUL instruction had a bug where, when the rounding mode was set to "round up", the sticky bit was not set correctly for certain combinations of source operand mantissa values, specifically:

$$src1[67:0] := X*2^{(i+15)} + 1*2^{i}$$

$$src2[67:0] := Y*2^{(j+15)} + 1*2^{j}$$

Where i+j = 54, and {X, Y} are any integers that fit in the 68-bit range.

Cluster-Level Testing

One of the fundamental decisions that they took early in the **Pentium 4 processor** development program was to develop Cluster Test Environments (CTEs) and maintain them for the life of the project. There is a CTE for each of the six clusters into which the **Pentium 4 processor** design is logically subdivided.

One measure of the success of the CTEs is that they caught almost 60% of the bugs found by dynamic testing at the SRTL level. Another is that, unlike the Pentium Pro processor and some other new microarchitecture developments, the **Pentium 4 processor** never needed an SRTL "get-well plan" at the full-chip level where new development is halted until the health of the full-chip model can be stabilized.

Power Reduction Validation

From the earliest days of the **Pentium 4 processor** design, power consumption was a concern. Even with the lower operating voltages offered by P858, it was clear that at the operating frequencies we were targeting we would have difficulty staying within the "thermal envelope" that was needed to prevent a desktop system from requiring exotic and expensive cooling technology. This led us to include two main mechanisms for active power reduction in the design: clock gating and thermal management. Each of these is discussed in other papers in this issue of the Intel Technology Journal. Each presented validation challenges—in particular, clock gating.

Full-chip Integration and Testing

With a design as complex as the **Pentium 4 processor**, integrating the pieces of SRTL code together to get a functioning full-chip model (let alone one capable of executing IA-32 code) is not a trivial task. The Architecture Validation (AV) team took the lead in developing tests that would exercise the new features as they became available in each phase, but did not depend upon any as-yet unimplemented IA-32 features. When a new feature was released to full-chip for the first time, a validator took responsibility for running his or her feature exercise tests, debugging the failures, and working with designers to rapidly drive fixes into graft (experimental) models, thereby bypassing the normal code turn-in procedure, until an acceptable level of stability was achieved.

System Validation

System Validation organization comprised a number of teams that targeted major CPU attributes:

• **Architecture** — including the Instruction Set Architecture (ISA), floating-point unit, data space, and virtual memory

• **Micro architecture** — focusing on boundary conditions between microarchitectural units

• **Multi-processor** — focusing on memory coherency, consistency, and synchronization

Bug Discussion

Comparing the development of the **Pentium® 4 processor** with the Pentium® Pro processor, there was a 350% increase in the number of bugs filed against SRTL prior to tapeout. The breakdown of bugs by cluster was also different: on the Pentium Pro processor microcode was the largest single source of bugs, accounting for over 30% of the total, whereas on the **Pentium 4 processor**, microcode accounted for less than 14% of the bugs.

The major categories were as follows:

• **RTL Coding (18.1%)** — these were things like typos, cut and paste errors, incorrect assertions (instrumentation) in the SRTL code, or the designer misunderstood what he/she was supposed to implement.

• **Microarchitecture (25.1%)** — this covered several categories: problems in the microarchitecture definition, architects not communicating their expectations clearly to designers, and incorrect documentation of algorithms, protocols, etc.

• **Logic/Microcode Changes (18.4%)** — these were bugs that occurred because: the design was changed, usually to fix bugs or timing problems, or state was not properly cleared or initialized at reset, or these were bugs related to clock gating.

• **Architecture (2.8%)** — certain features were not defined until late in the project. This led to shoehorning them into working functionality.

Interconnect and Noise Immunity Design for the Pentium® 4 Processor

The **Pentium® 4 processor** is Intel's fastest processor so far. It contains aggressive domino pipelines, pulsed circuits, and novel circuit families that attain very high speed at the cost of reduced-noise margins. Controlling interconnect RC delay is of paramount importance at such high frequencies. At the same time, the need for a high-volume ramp in the desktop segment necessitates high-density wiring constraints that prevent us from spacing or shielding all critical wires to manage coupling noise. All of these made the task of interconnect and noise design and verification quite challenging.

Interconnect Delay & Crosscapacitance Scaling

The problem has become significant enough to require entire architectural pipe stages in the **Pentium 4 processor** for interconnect communication. To avoid degrading interconnect resistance, the vertical dimension of metals has scaled very weakly compared to the horizontal dimension, leading to extremely high height/width aspect ratios (2-2.2). See **Figure 9.**

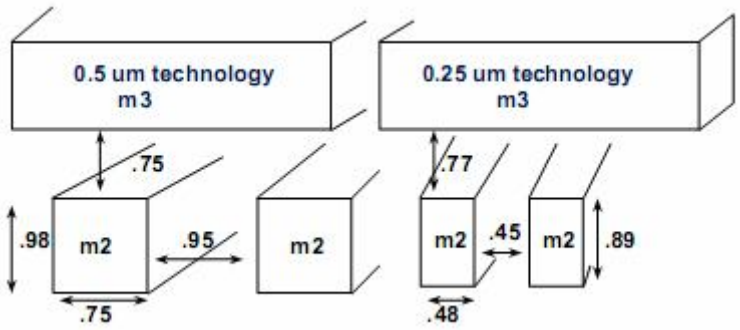


Figure 9: Wire aspect ratio scaling with technology

Nowadays, most of the wire capacitance is to parallel neighboring wires in the same layer (**Figure 10**), which can get routed together for long distances.

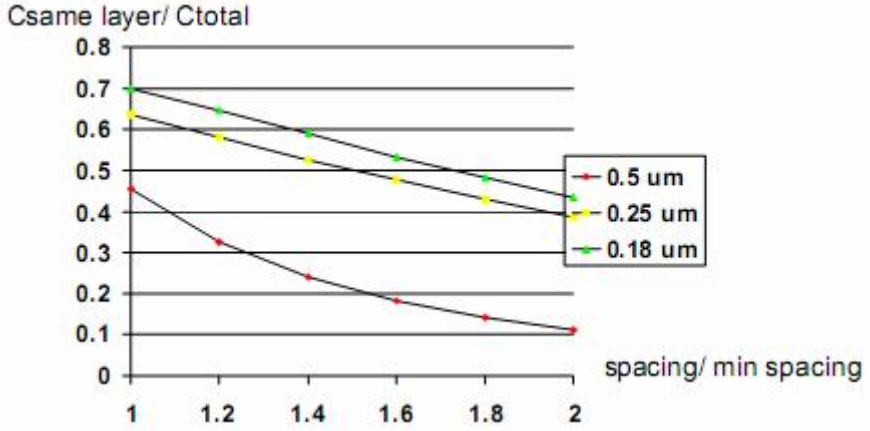


Figure 10: Coupling capacitance scaling with technology

Wire & Repeater Design Methodology for the Pentium 4 Processor

The extensive use of dedicated repeater blocks is evident in the **Pentium 4 processor** floor plan shown in **Figure 11**. Further, the net length comparison in **figure 11** shows that although the **Pentium 4 processor** is a much larger chip, there are very few long nets in it compared to previous-generation chips such as the Pentium III processor. This is even more notable given that the **Pentium 4 processor** has more than twice as many full-chip nets as the Pentium III processor and has architecturally bigger blocks. If we compare the M5 wire segments of the Pentium III, and **Pentium 4 processors**, we note that 90% of the M5 wire segments of the **Pentium 4 processor** are shorter than 2000 microns while the same percentage of Pentium III processor wires are 3500 microns long. These short wires are a key to enabling high-frequency operation.

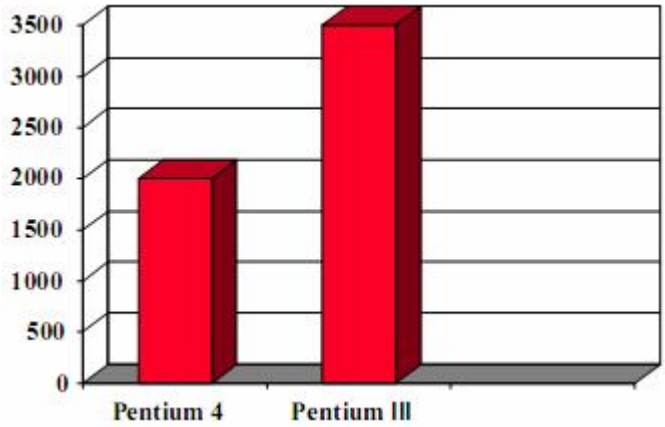


Figure 11: M5 length comparison of global wires for different processors using the same 0.18 um technology

Cross capacitance & Density Comparative Results of the Pentium 4 Processor Interconnect

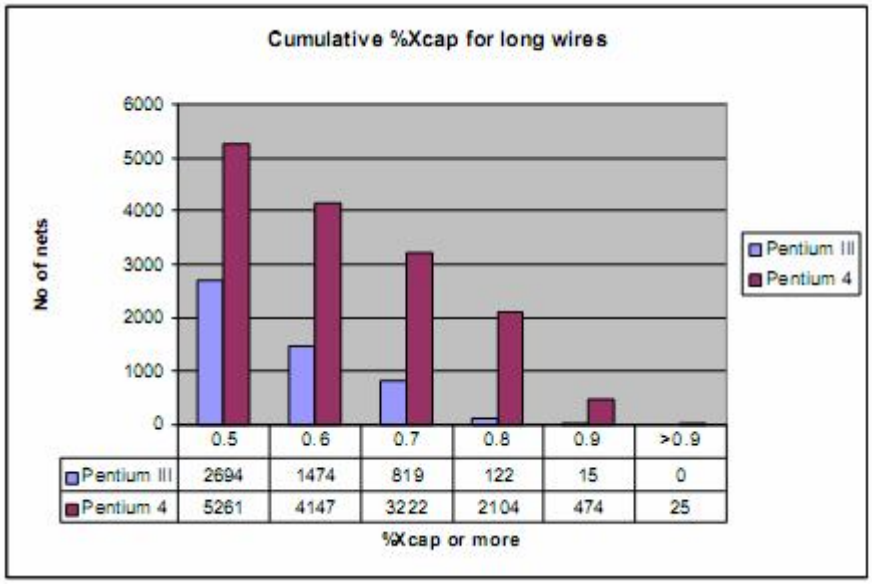


Figure 12: Coupling comparison of Pentium 4 processor/Pentium III processor wires

Figure 12 clearly shows that the **Pentium 4 processor** has significantly more wires with high crosscapacitance than does the Pentium III processor. This aggressive wiring makes additional accuracy in noise CAD tools (discussed later) even more important.

Noise Challenges on the Pentium 4 Processor

The performance goals of the Intel **Pentium 4 processor** compared to the Pentium III processor were 1.5X–2X higher frequency on the normal (medium) part of the chip and 3X–4X the frequency on the fast (rapid execution engine) part of the chip.

Accurate noise analysis using NoisePad and circuit styles such as pseudo-CMOS logic shown in **Figure 13** were employed.

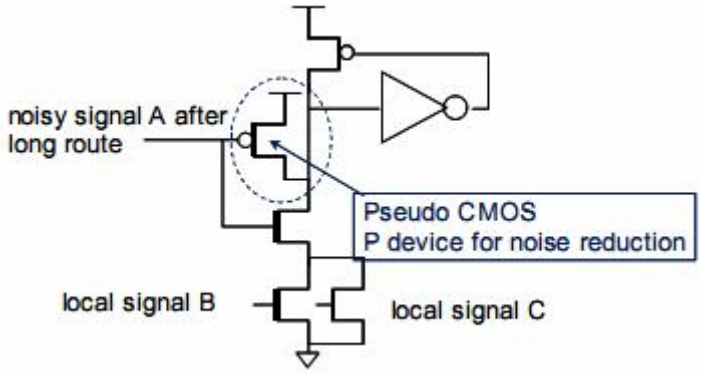


Figure 13: Pseudo CMOS circuit for input noise protection

Process Optimization Consideration for Noise and Leakage

Most design rules and circuit decisions for the **Pentium 4 processor**, were based on early 0.18 um process specs. As shown in **Figure 14** by the process trend over time, this was indeed a wise choice: the **Pentium 4 processor** has scaled well in frequency and still has considerable frequency headroom speed.

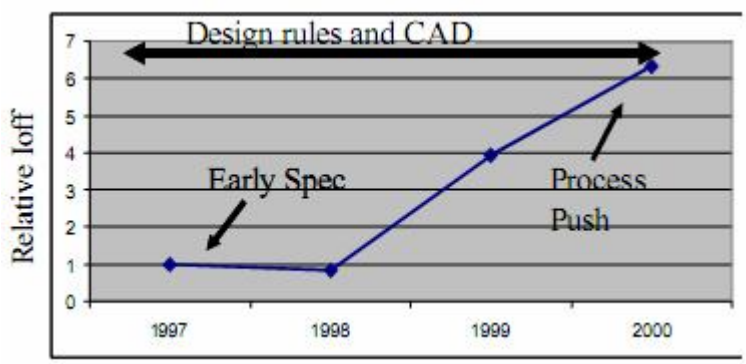


Figure 14: Impact of process push on subthreshold leakage

Small Signal Unity Gain

Prior to our work on the **Pentium 4 processor**, traditional analysis of noise margins relied on the small signal unity gain failure criteria.

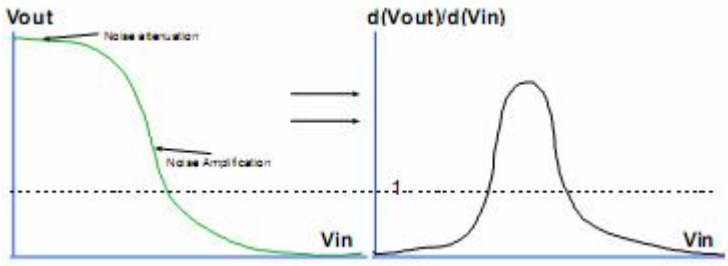


Figure 15: DC transfer function of an inverter illustrating small signal unity gain

As illustrated in **Figure 15**, for a small change in input noise to a circuit biased at an operating point, the resultant change in output noise is measured. **If |d (Output)/ d (Input)| > 1 then the circuit is considered unstable.**

Full-Chip Wire Noise Verification

The key idea behind the **Pentium 4 processor** full-chip noise verification is "strobed signaling." A non-restoring node for noise is defined as a node, which if falsely tripped due to noise, will not recover with the passage of time (e.g., domino node or off pass gate latch). A signal is called "strobed," if its logic cone leading to a non-restoring noise node is controlled with a clock (e.g., D1k domino). In this case, the effect of noise on this node may be dependent on clock frequency.

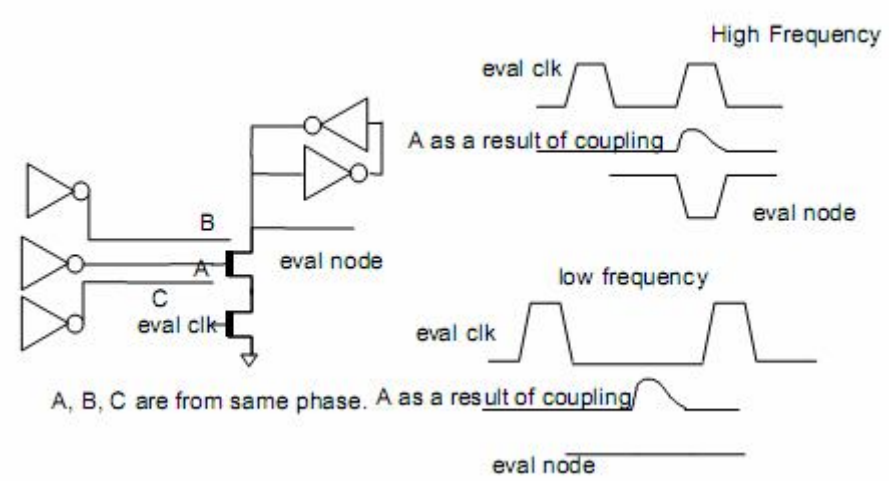


Figure 16: Impact of frequency on noise failure

As shown with the D1-k example in **Figure 16**, at a lower frequency, the noise will settle down before the signal is sampled and as such will not fail at the lower frequency.

Full-Chip Noise Convergence

For a lead processor like the **Pentium 4 processor**, "clean" data for all nets are available only very close to tapeout. Further, this detailed model is too slow to turn and, moreover, it is serial in nature. After finding a violation, one has to backtrack through numerous files, models, and schematics to verify if a real problem exists (needle in a haystack scenario). With these incomplete data, trending and schedule predictions are difficult. The dramatic decrease in noise violations seen in **Figure 17** involved no work from the design team!

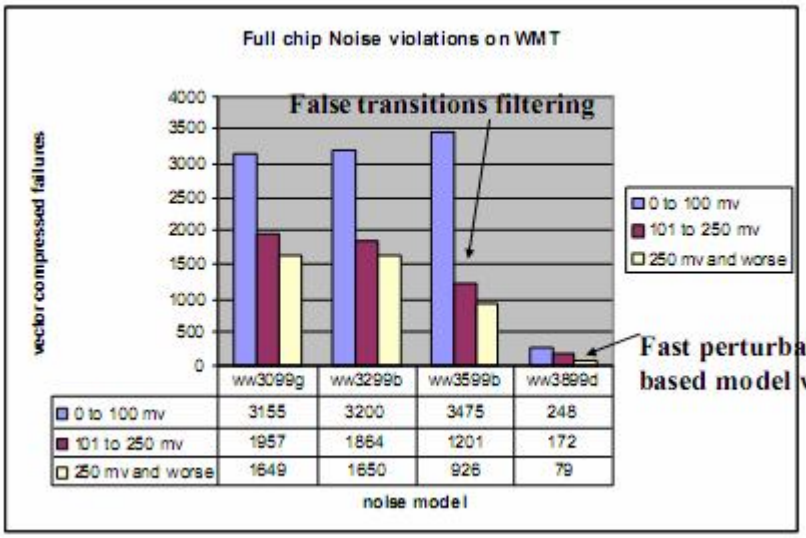


Figure 17: Road to noise convergence on the Pentium 4 processor

Automatic Vectorization

Figure 18 shows the speedup (serial vs. vector execution time) obtained on a 1.5GHz. **Pentium 4 processor** by automatic vectorization of a single-precision dot-product kernel (SDOT) and a double-precision dot-product kernel (DDOT) for array lengths ranging from 1 to 64K.

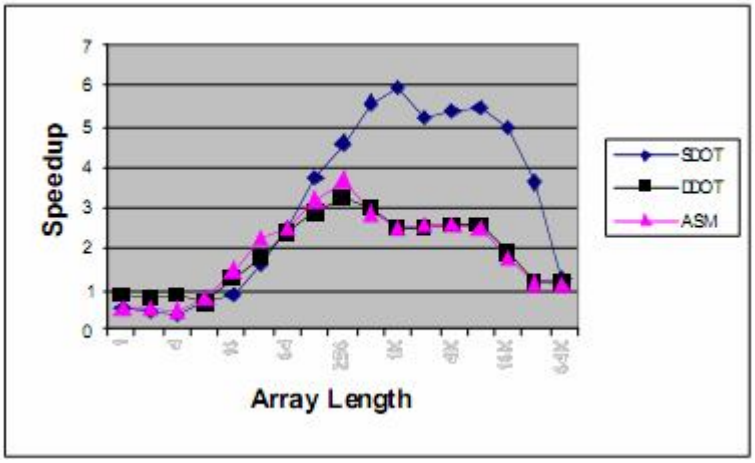


Figure 18: Speedup for dot-product on a Pentium 4 processor

In the last graph shown in **Figure 19**, we show the speedup obtained on a 1.5GHz. **Pentium 4 processor** by automatic vectorization of kernels of the form “x[i] = F(y[i])”.

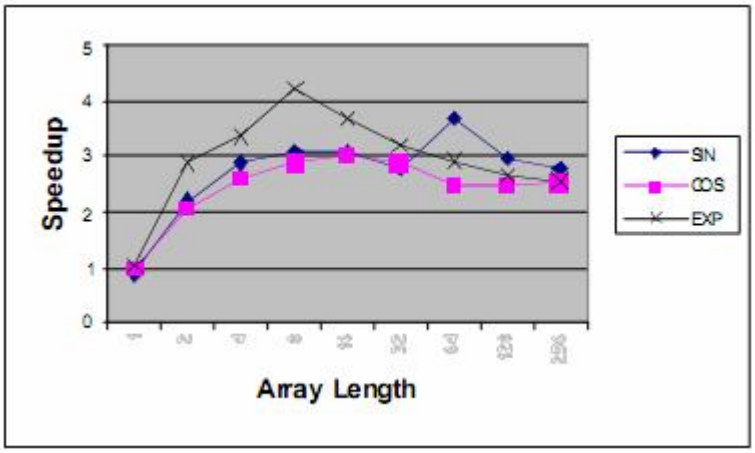


Figure 19: Speedup for math functions on a Pentium® 4 processor

Platform Improvements Deliver Performance

System bus bandwidth is a limiter for the performance of the platform when high bandwidth is required by the application. This was foremost in the minds of the designers of the Intel **Pentium 4 processor** when developing the system bus. The system bus used in the **Pentium 4 processor** delivers unprecedented bandwidth for the PC platform, as can be seen in **Figure 21**.

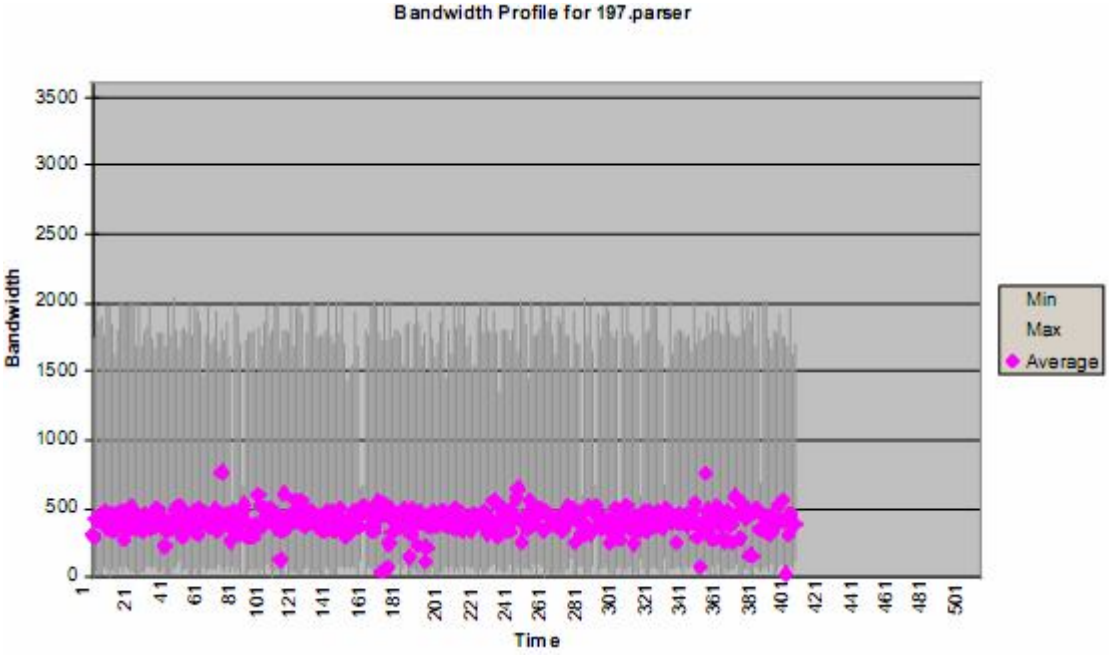


Figure 21: Pentium® 4 processor 197. Parser bandwidth profile

The corresponding results on a **Pentium 4** system are shown in **Figure 22**. The bandwidth demanded by the **Pentium 4 processor** is nearly doubled over that of the Pentium® II processor. Since the **Pentium 4 processor** platform is able to satisfy the higher demand, the execution time of 171.swim is greatly reduced.

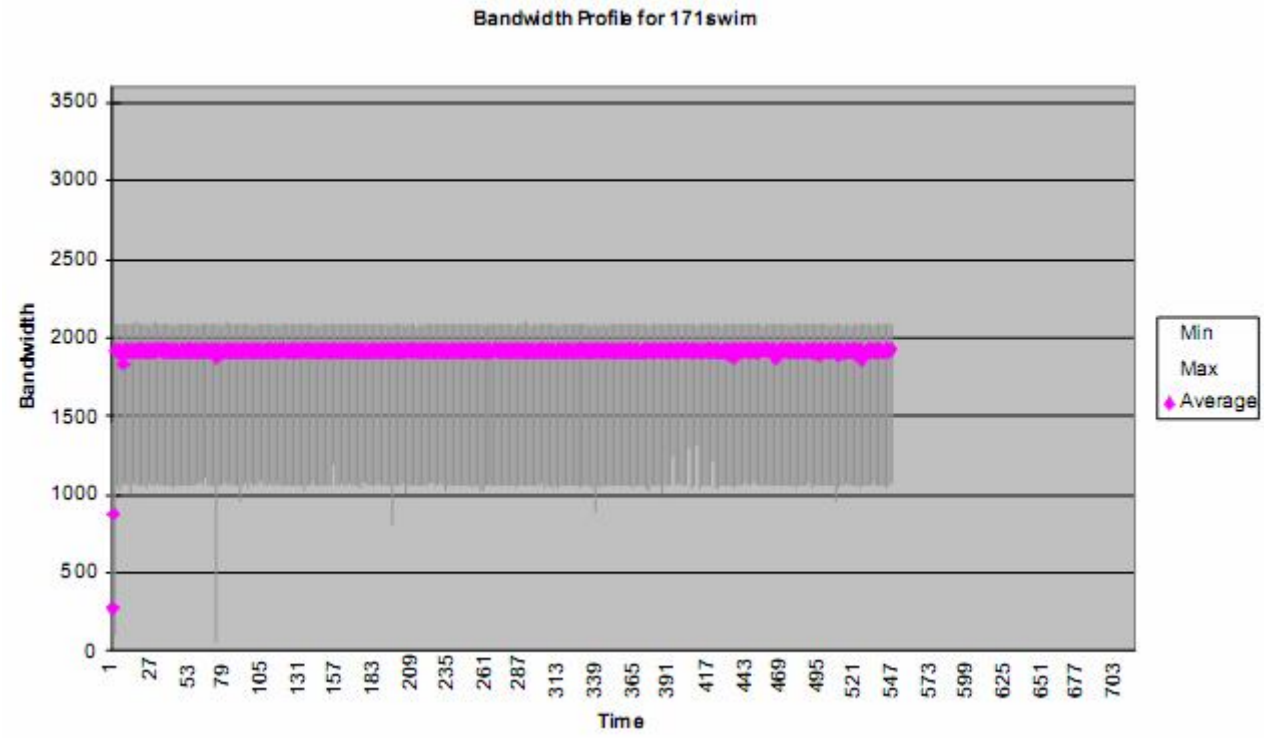


Figure 22: Pentium® 4 processor 171.swim profile

Challenges to Conventional Approach

Server density has grown dramatically over the past decade to keep pace with escalating performance requirements for enterprise applications. Ongoing progress in processor designs has enabled servers to continue delivering increased performance, which in turn helps fuel the powerful applications that support

rapid business growth. However, increased performance incurs a corresponding increase in processor power consumption—and heat is a consequence of power use.

As a result, administrators must determine not only how to supply large amounts of power to systems, but also how to contend with the large amounts of heat that these systems generate in the data center. As more applications move from proprietary to standards based systems, the performance demands on industry standard servers are spiraling upward. Today, in place of midrange and large mainframe systems, tightly packed racks of stand-alone servers and blade servers can be clustered to handle the same types of business-critical application loads that once required large proprietary systems. Organizations are using databases such as Microsoft SQL Server, Oracle Database 0 g, g and MySQL to enhance business decision making along with enterprise-wide messaging applications such as Microsoft Exchange. Meanwhile, network infrastructure, Internet connectivity, and e-commerce are growing at tremendous rates. Altogether, the result is a steady increase in performance demands as user loads and processing loads grow, driving a steady increase in the density of systems in the data center, which is intensified by ever-faster processors—and in turn this can create power and cooling challenges for many IT organizations.

Optimizing Software Applications

Software optimization can be an efficient way to enable incremental performance gains without increasing power consumption and heat. Many of today's leading software tools, along with Intel compilers, can enable significant performance improvements over applications that have not been compiled or tuned using such optimization tools. Actual performance gains will depend on the specific system configuration and application environment. To get the most performance from existing data center components, administrators must not overlook potential gains from optimizing software applications during the infrastructure planning processes.

Power & Cooling Advantages of Multicore Processors

A multicore architecture can help alleviate the environmental challenges created by high-clock-speed, single-core processors. Heat is a function of several factors, two of which are processor density and clock speed. Other drivers include cache size and the size of the core itself. In traditional architectures, heat generated by each new generation of processors has increased at a greater rate than clock speed. Multicore processors may help administrators minimize heat while maintaining high overall performance.

This capability may help make future multicore processors attractive for IT deployments in which density is a key factor, such as high-performance computing (HPC) clusters, Web farms, and large clustered applications. Currently, technologies such as demand-based switching (DBS) are beginning to enter the mainstream, helping organizations reduce the utility power and cooling costs of computing. DBS allows a processor to reduce power consumption (by lowering frequency and voltage) during periods of low computing demand. DBS is available in single-core processors today, and its inclusion in multicore processors may add capabilities for managing power consumption and, ultimately, heat output.

Significance of Sockets in a Multicore Architecture

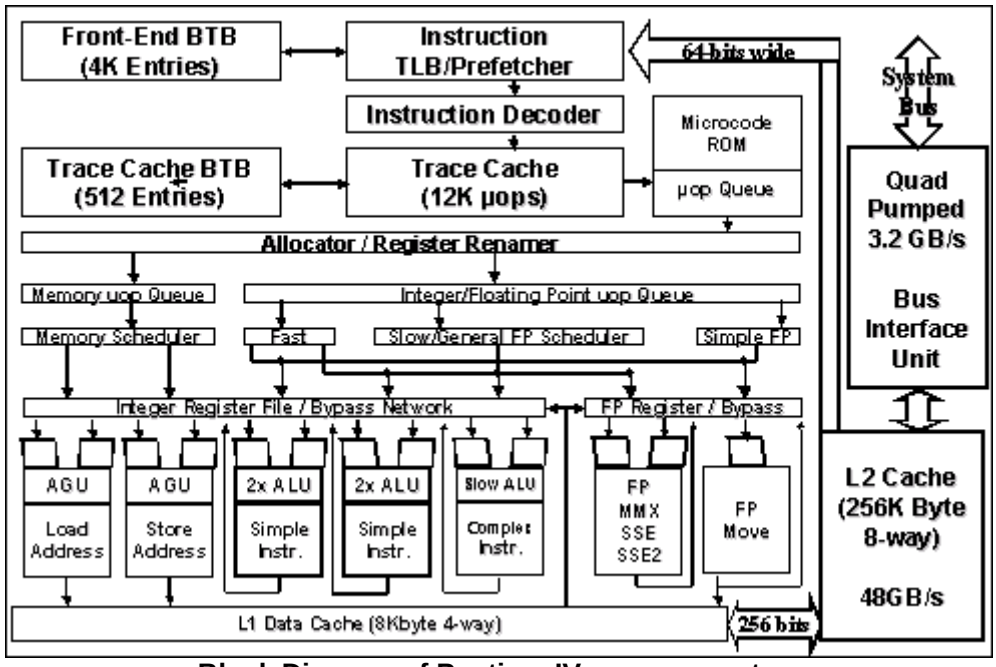
However, multicore processors will call for a new mind-set that considers processor cores as well as sockets.

Single-threaded applications that perform best today in a single-processor environment will likely continue to be deployed on single-processor, single-core system architectures. For single-threaded applications, which cannot make use of multiple processors in a system, moving to a multiprocessor, multicore architecture may not necessarily enhance performance. Most of today's leading operating systems, including Microsoft Windows Server System and Linux variants, are multithreaded, so multiple single-threaded applications can run on a multicore architecture even though they are not inherently multithreaded.

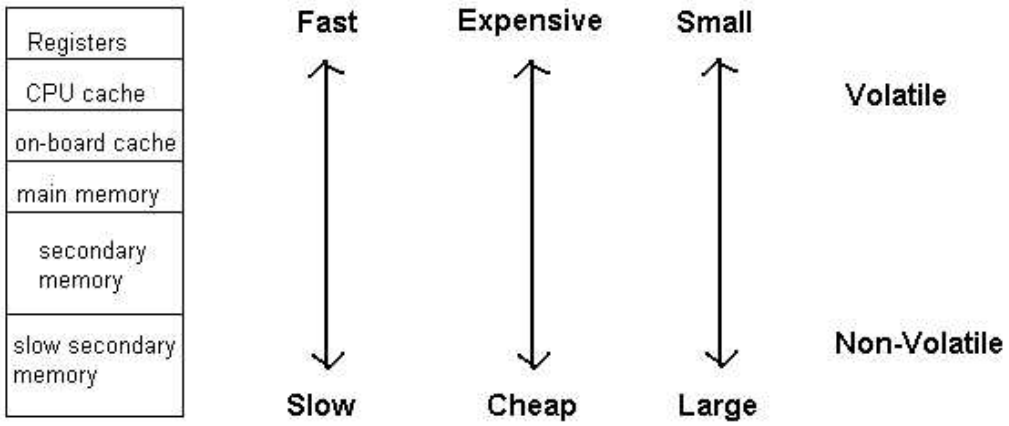
However, for multithreaded applications that are currently deployed on single-processor architectures because of cost constraints, moving to a single-processor, dual-core architecture has the potential to offer performance benefits while helping to keep costs low. Because higher-powered, dual-socket systems typically run applications that are more mission-critical than those running on less-powerful, single-processor systems, organizations may continue to expect more availability, scalability, and performance features to be designed for dual-socket systems relative to single-socket systems—just as they do today.

For applications running today on high-performing quad processor systems, a transition to multicore technology is not necessarily an opportunity to move from four-socket, four-core systems to dual-socket, four-core systems.

Complex Modern Pentium IV Memory System



Block Diagram of Pentium IV memory system



Storage Hierarchy

- **CPU cache** - memory located on the processor chip (VOLATILE)
- **on-board cache** - located on circuit board; fastest external memory available (VOLATILE)
- **main memory** - software managed (VOLATILE)
- **secondary memory** - hard drive (NON-VOLATILE)

- **slow secondary memory** - tapes, diskettes (NON-VOLATILE)

❖ Pentium 4's Cache Organization

Cache Organization in the Memory Hierarchy

There is usually a trade-off between cache size and speed. This is mostly because of the extra capacitive loading on the signals that drive the larger SRAM arrays. Refer to the block diagram of the Pentium 4 memory system. Intel has chosen to keep **the L1 caches rather small so that they can reduce the latency of cache accesses**. Even a data cache hit will take 2 cycles to complete (6 cycles for floating-point data). We'll talk about the L1 caches in a moment, but further down the hierarchy we find that the **L2 cache is an 8-way, unified (includes both instruction and data)**, 256KB cache with a 128B line size.

The 8-way structure means it has 8 sets of tags, providing about the same cache miss rate as a "fully-associative" cache (as good as it gets). This makes the 256KB cache more effective than its size indicates, since the **miss rate of this cache is approximately 60% of the miss rate for a direct-mapped (1-way) cache of the same size**.

The downside is that **an 8-way cache will be slower to access**. Intel states that the load latency is 7 cycles (this reflects the time it takes an L2 cache line to be fully retrieved to either the L1 data cache or the x86 instruction pre-fetch/decode buffers), but the cache is able to transfer new data every 2 cycles (which is the effective throughput assuming multiple concurrent cache transfers are initiated). Again, notice that the **L2 cache is shared between instruction fetches and data accesses (unified).**

System Bus Architecture is matched to Memory Hierarchy Organization

One interesting change for the L2 cache is to make the line size 128 bytes, instead of the familiar 32 bytes. The **larger line size can slightly improve the hit rate (in some cases), but requires a longer latency for cache line refills from the system bus**. This is where the new Pentium 4 bus comes into play. Using a 100MHz clock and transferring data four times on each bus clock (which Intel calls a 400MHz data rate), the 64-bit system bus can bring in 32 bytes each cycle. This translates to a bandwidth of 3.2 GB/sec.

The longer line size still causes a longer latency before getting all the burst data from main memory. In fact, some analysts note that **P4 systems** have about 19% more memory latency than Pentium III systems (measured in nanoseconds for the demand word of a cache refill). Smart pre-fetching is critical or else the **P4** will end up with less performance on many applications.

Pre-Fetching Hardware Can Help if Data Accesses Follow a Regular Pattern

The **L2 cache has pre-fetch hardware to request the next 2 cache lines (256 bytes)** beyond the current access location. This pre-fetch logic has some intelligence to allow it to monitor the history of cache misses and try to avoid unnecessary pre-fetches (that waste bandwidth and cache space The hardware pre-fetch logic should easily notice the pattern of cache misses and then pre-load data, leading to much better performance on applications like streaming media types (like video).

Designing for Data Cache Hits

Intel boasts of "new algorithms" to allow faster access to the 8KB, four-way, L1 data cache. They are most likely referring to the fact that the **Pentium 4** speculatively processes load instructions as if they always hit in the L1 data cache (and data TLB). By optimizing for this case, **there aren't any extra cycles burned while cache tags are checked for a miss**. The load instruction is sent on its merry way down the pipeline; if a cache miss delays the load, the processor passes temporarily incorrect data to dependent instructions that assumed the data arrived in 2 cycles.

The Pentium® 4 Processor, Advanced Technology for the Internet and Beyond

The launch of a brand new micro architecture and supporting platform, such as the **Pentium® 4 processor** platform, is an especially proud and exciting moment for Intel's engineers and technologists. Not only is the product launch the pinnacle of a long and intense development cycle, it is also the moment when the innovations underlying the product begin to alter the computing landscape. This allows the innovator to witness the effects of his or her ideas on the world at large.

The **Pentium 4 processor** platform is the beginning of a whole new family of products from Intel. The range of new technologies and innovations inherent in this platform is breathtaking and constitutes the foundation upon which Intel will be able to build for years to come. I am confident that the **Pentium 4 processor** will have a profound effect on the computing industry, taking performance to dizzying new heights and enabling new uses for end users. In particular, applications such as speech, natural language processing, and video are quite likely to become pervasive with the arrival of the **Pentium 4 processor** platform.

Several of the key innovations and technologies underlying the **Pentium 4 processor-based** platform are described in this issue of the Intel® Technology Journal by the engineers who first had the ideas and then worked long and hard to turn those ideas into reality. A highly optimized and balanced system design that uses a novel chipset, a quad-pumped processor system bus and high-performance RDRAM memory, and last but not least, compiler methods to leverage new instructions introduced with the **Pentium 4 processor**.

At the center is the new **Pentium 4 processor** with great performance today and enormous frequency and performance headroom for the future. At its launch frequency of 1.5GHz, the **Pentium 4 processor** is already in a class by itself for multimedia performance, floating-point performance, and the world's highest integer performance.

Conclusion

The **Pentium 4 processor** is a new, state-of-the-art processor microarchitecture and design. It is the beginning of a new family of processors that utilize the new Intel NetBurst microarchitecture. Its deeply pipelined design delivers world-leading frequencies and performance. It uses many novel microarchitectural ideas including a Trace Cache, double-clocked ALU, new low-latency L1 data cache algorithms, and a new high bandwidth system bus. It delivers world-class performance in the areas where added performance makes a difference including media rich environments (video, sound, and speech), 3D applications, workstation applications, and content creation.

The **Pentium® 4 processor** was highly functional on A-0 silicon and received production qualification in only ten months from tapeout. The work described here is a major reason why we were able to maintain such a tight schedule and enable Intel to realize early revenue from the **Pentium 4 processor** in today’s highly competitive marketplace.

Reference

1. D. Sager, G. Hinton, M. Upton, T. Chappell, T. Fletcher, S. Samaan, and R. Murray, “A 0.18um CMOS IA32 Microprocessor with a 4GHz Integer Execution Unit,” International Solid State Circuits Conference, Feb 2001.

2. Doug Carmean, “Inside the High-Performance Intel ® Pentium® 4 Processor Micro-architecture” Intel Developer Forum, fall 2000 at ftp://download.intel.com/design/idf/fall2000/presenta tions/pda/pda_s01_cd.pdf

3. IA-32 Intel Architecture Software Developer’s Manual Volume 1: Basic Architecture at http://developer.intel.com/design/pentium4/manuals/245470.htm.

4. Intel ® Pentium® 4 Processor Optimization Reference Manual at http://developer.intel.com/design/pentium4/manuals/248966.htm.

5. Clark, D, “Bugs Are Good: A Problem-Oriented Approach To The Management Of Design Engineering,” Research-Technology Management,33(3), May 1990, pp. 23-27.

6. Bentley, R., and Milburn, B., “Analysis of Pentium® Pro Processor Bugs,” Intel Design and Test Technology Conference, June 1996. Intel Internal Document.

7. Zucker, R., “Bug Root Cause Analysis for Willamette,” Intel Design and Test Technology Conference, August 2000. Intel Internal Document.

8. Rajesh Kumar, Eitan Zahavi, Desmond Kirkpatrick, “Accurate design and analysis of Noise Immunity for high-performance circuit design,” Design and Test Technology Conference (DTTC) 1997. Intel internal document.

9. Eitan Zahavi, Rajesh Kumar et. al., “Novel Methodology and Tools for Noise Immunity Design and Verification,” DTTC 1998. Intel internal document.

10. Madhu Swarna et. al., “Integrated timing and noise characterization of sequentials for accuracy and increased design space,” DTTC 2000. Intel internal document.

11. John Randal Allen and Ken Kennedy, “Automatic Translation of Fortran Programs into Vector Form,” ACM Transactions on Programming Languages and Systems: 9:491-542, 1987.

12. Michael J. Wolfe, High Performance Compilers for Parallel Computer, Addison-Wesley Publishing Company, Redwood City, California, 1996.

13. Hans Zima, Supercompilers for Parallel and Vector Computers, ACM Press, New York, NY, 1990.

14. McCalpin, John D., “Sustainable Memory Bandwidth in Current High-Performance Computers,” October 12, 1995.

15. http://www.intel.com/

16. http://www.intel.com/sites/corporate/tradmarx.htm