Inside the Pentium® 4 Processor Micro-architecture

Next Generation IA-32 Micro-architecture

Doug Carmean
Principal Architect
Intel Architecture Group

August 24, 2000
Agenda

- IA-32 Processor Roadmap
- Design Goals
- Frequency
- Instructions Per Cycle
- Summary
Intel® Pentium® 4 Processor

Performance

Time

486 Micro-architecture

P5 Micro-Architecture

P6 Micro-Architecture

Intel® NetBurst™ Micro-Architecture

Today

Copyright © 2000 Intel Corporation.
Intel® Pentium® 4 Processor Design Goals

- Deliver world class performance across both existing and emerging applications
- Deliver performance headroom and scalability for the future

Micro-architecture that will Drive Performance Leadership for the Next Several Years
Delivered Performance = \text{Frequency} \times \text{Instructions Per Cycle}
Frequency

- What limits frequency?
  - Process technology
  - Microarchitecture

- On a given process technology
  - Fewer gates per pipeline stage will deliver higher frequency

Frequency is driven by Micro-architecture
Netburst™ Micro-architecture
Pipeline vs P6

Basic P6 Pipeline

1. Fetch
2. Fetch
3. Decode
4. Decode
5. Decode
6. Rename
7. ROB Rd
8. Rdy/Sch
9. Exec

Intro at 733MHz .18μ

Basic Pentium® 4 Processor Pipeline

1. TC Nxt IP
2. TC Fetch
3. Drive
4. Alloc
5. Rename
6. Que
7. Sch
8. Sch
9. Exec
10. Disp
11. Disp
12. Disp
13. Disp
14. Disp
15. Disp
16. Exec

Intro at ≥ 1.4GHz .18μ

Hyper pipelined Technology enables industry leading performance and clock rate
Hyper Pipelined Technology

Introduction

Time

Frequency

P5 Micro-Architecture

60MHz

166MHz

233MHz

P6 Micro-Architecture

1.13GHz

1.4GHz

Netburst™ Micro-Architecture

Today

≥1.4GHz

Copyright © 2000 Intel Corporation.
Hyper pipelined Technology

TC Nxt IP: Trace cache next instruction pointer
Pointer from the BTB, indicating location of next instruction.
Hyper Pipelined Technology

TC Fetch: Trace cache fetch
Read the decoded instructions (uOPs) out of the Execution Trace Cache
Hyper Pipelined Technology

<table>
<thead>
<tr>
<th></th>
<th>TC Nxt IP</th>
<th>TC Fetch</th>
<th>Allo</th>
<th>Rd</th>
<th>Qe</th>
<th>Sch</th>
<th>Sch</th>
<th>Disp</th>
<th>RF</th>
<th>Ex</th>
<th>Flgs</th>
<th>Br Ck</th>
<th>Drive</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Drive</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Drive:** Wire delay
Drive the uOPs to the allocator

![Diagram of system interface and pipeline stages](image-url)
Hyper Pipelined Technology

Alloc: Allocate
Allocate resources required for execution. The resources include Load buffers, Store buffers, etc..

Copyright © 2000 Intel Corporation.
**Hyper Pipelined Technology**

|   |   |   |   |   | 7 | 8 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 1 | TC Nxt IP | 2 | TC Fetch | 3 | Drive | 4 | Alloc | 5 | Rename | 6 |   | 9 | Que | 10 | Sch | 11 | Sch | 12 | Disp | 13 | Disp | 14 | RF | 15 | RF | 16 | Ex | 17 |   | 18 | Figs | 19 | Br Ck | 20 | Drive |

**Rename: Register renaming**

Rename the logical registers (EAX) to the physical register space (128 are implemented).
Que: Write into the uOP Queue

uOPs are placed into the queues, where they are held until there is room in the schedulers.
Hyper Pipelined Technology

Sch: Schedule
Write into the schedulers and compute dependencies. Watch for dependency to resolve.
Disp: Dispatch
Send the uOPs to the appropriate execution unit.
RF: Register File

Read the register file. These are the source(s) for the pending operation (ALU or other).
Hyper Pipelined Technology

Ex: Execute

Execute the uOPs on the appropriate execution port.

Copyright © 2000 Intel Corporation.
Hyper Pipelined Technology

Flgs: Flags

Compute flags (zero, negative, etc..). These are typically the input to a branch instruction.
**Hyper Pipelined Technology**

**Br Ck: Branch Check**

The branch operation compares the result of the actual branch direction with the prediction.
Hyper Pipelined Technology

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC Nxt IP</td>
<td>TC Fetch</td>
<td>Drive</td>
<td>Alloc</td>
<td>Rename</td>
<td>Que</td>
<td>Sch</td>
<td>Sch</td>
<td>Disp</td>
<td>Disp</td>
<td>RF</td>
<td>RF</td>
<td>Ex</td>
<td>Flgs</td>
<td>Br Ck</td>
<td>Drive</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Drive: Wire delay**

Drive the result of the branch check to the front end of the machine.
Delivered Performance = Frequency * Instructions Per Cycle
Improving Instructions Per Cycle

- Improve efficiency
  - Do more things in a clock
  - Branch prediction
- Reduce time it takes to do something
  - Reducing latency
Improving Instructions Per Cycle

- Improve efficiency
  - Branch prediction
  - Do more things in a clock

- Reduce time it takes to do something
  - Reducing latency
Branch Prediction

- Accurate branch prediction is key to enabling longer pipelines
- Dramatic improvement over P6 branch predictor:
  - 8x the size (4K)
  - Eliminated 1/3 of the mispredictions
- Proven to be better than all other publicly disclosed predictors
  - (g-share, hybrid, etc)
The Execution Trace Cache
Execution Trace Cache

- Advanced L1 instruction cache
  - Caches “decoded” IA-32 instructions (uops)
- Removes decoder pipeline latency
- Capacity is ~12K uOps
- Integrates branches into single line
  - Follows predicted path of program execution

Execution Trace Cache feeds fast engine
Execution Trace Cache

1 cmp 2 br \rightarrow T1
   \ldots (unused code)
T1: 3 sub
   4 br \rightarrow T2
   \ldots (unused code)
T2: 5 mov 6 sub
   7 br \rightarrow T3
   \ldots (unused code)
T3: 8 add 9 sub
   10 mul 11 cmp 12 br \rightarrow T4

Trace Cache Delivery

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>cmp</td>
<td>2</td>
<td>br T1</td>
</tr>
<tr>
<td>3</td>
<td>T1:</td>
<td>sub</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>br T2</td>
<td>5</td>
<td>mov</td>
</tr>
<tr>
<td>6</td>
<td>sub</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>br T3</td>
<td>8</td>
<td>T3:add</td>
</tr>
<tr>
<td>9</td>
<td>sub</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>mul</td>
<td>11</td>
<td>cmp</td>
</tr>
<tr>
<td>12</td>
<td>br T4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Advanced Dynamic Execution

- Extends basic features found in P6 core
- Very deep speculative execution
  - 126 instructions in flight (3x P6)
  - 48 loads (3x P6) and 24 stores (2x P6)
- Provides larger window of visibility
  - Better use of execution resources

Deep Speculation Improves Parallelism
Improving Instructions Per Cycle

- Improve efficiency
  - Do more things in a clock
  - Branch prediction

- Reduce time it takes to do something
  - Reducing latency
Rapid Execution Engine

- Dramatically lower ALU latency
- P6:
  - 1 clock @ 1GHz
- Intel® NetBurst™ micro-architecture:
  - ½ clock @ >1.4GHz
  - <0.35ns
High Performance L1 Data Cache

- 8KB, 4-way set associative, 64-byte lines
- Very high bandwidth
  - 1 Ld and 1 St per clock
- New access algorithms
- Very low latency
  - 2 clock read

New algorithm enables faster cache
**Data Speculation**

- **Observation:** Almost all memory accesses hit in the cache
- **Optimize for the common case**
  - Assume that the access will hit the cache
  - Use a low cost mechanism to fix the rare cases that miss
- **Benefit:**
  - Reduces latency
  - Significantly higher performance
Replay

- Repairs incorrect speculation
  - Re-execute until correct
- Replay is uOP specific
  - Replay the uOP that mis-speculated
  - Replay dependent uOPs
  - Independent uOPs are not replayed

Efficient mechanism to reduce latency
L1 Cache is >2x Faster

- **P6:**
  - 3 clocks @ 1GHz

- **Intel® NetBurst™ micro-architecture:**
  - 2 clocks @ ≥1.4GHz

Lower Latency is Higher Performance
Example with higher IPC and Faster Clock!

Code Sequence

P6 @1GHz
Ld
Add
Add
Ld
Add
Add

10 clocks
10ns
IPC = 0.6

Intel® NetBurst™ Micro-architecture @1.4GHz

6 clocks
4.3ns
IPC = 1.0

Copyright © 2000 Intel Corporation.
L2 ATC Organization

- 256KB, 8-way set associative
  - 128-byte lines
  - Two 64-byte pieces per line
- Holds both data and instructions
- High bandwidth: 45 GB/Sec @ 1.4GHz
  - 2.8x P6 @1GHz
Aggregate Cache Latency

- Function of all caches in a processor
- Overall Effective Latency
  
  \[ L_1 \text{ latency } + L_1 \text{ Miss Rate } \times L_2 \text{ latency } + L_2 \text{ Miss Rate } \times \text{DRAM Latency} \]

Average cache speed is >1.8x better than the Pentium® III Processor

Average on desktop applications,
Intel® Pentium® III processor @ 1GHz, Intel® Pentium® 4 processor @ 1.4GHz
## Comparison

<table>
<thead>
<tr>
<th></th>
<th>Pentium® III Processor</th>
<th>Pentium 4 Processor</th>
<th>Relative Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>1 GHz</td>
<td>≥ 1.4 Ghz</td>
<td>≥ 1.4</td>
</tr>
<tr>
<td>Adder Speed</td>
<td>1 ns</td>
<td>&lt; .36 ns</td>
<td>≥ 2.8</td>
</tr>
<tr>
<td>Adder Bandwidth</td>
<td>2 billion/sec</td>
<td>≥ 5.6 billion/sec</td>
<td>&gt; 2.8</td>
</tr>
<tr>
<td>L1 Cache Speed</td>
<td>3 ns</td>
<td>&lt; 1.42 ns</td>
<td>≥ 2.1</td>
</tr>
<tr>
<td>L1 Cache Size</td>
<td>16 KB</td>
<td>8 KB</td>
<td>0.5</td>
</tr>
<tr>
<td>L1 Cache Bandwidth</td>
<td>16 GB/sec</td>
<td>≥ 44.8 GB/sec</td>
<td>≥ 2.8</td>
</tr>
<tr>
<td>L2 Cache Bandwidth</td>
<td>16 GB/sec</td>
<td>≥ 44.8 GB/sec</td>
<td>≥ 2.8</td>
</tr>
<tr>
<td>Instructions In flight</td>
<td>40</td>
<td>126</td>
<td>3.15</td>
</tr>
<tr>
<td>Loads in flight</td>
<td>16</td>
<td>48</td>
<td>3</td>
</tr>
<tr>
<td>Stores in flight</td>
<td>12</td>
<td>24</td>
<td>2</td>
</tr>
<tr>
<td>Branch targets</td>
<td>512</td>
<td>4092</td>
<td>8</td>
</tr>
<tr>
<td>Uop Fetch Bandwidth</td>
<td>3 billion/sec</td>
<td>≥ 4.2 billion/sec</td>
<td>≥ 1.4</td>
</tr>
</tbody>
</table>
Intel® Pentium® 4 Processor

Summary

- Revolutionary, new micro-architecture from Intel designed for the evolving Internet
- Design features for balanced, high performance platform scalability and headroom
- The world’s highest performance desktop processor