CS252
Graduate Computer Architecture

Lecture 18: ILP and Dynamic Execution #3: Examples (Pentium III, Pentium 4, IBM AS/400)

April 4, 2001

Prof. David A. Patterson

Computer Science 252

Spring 2001

Review: Dynamic Branch Prediction

- Prediction becoming important part of scalar execution
- Branch History Table: 2 bits for loop accuracy
- Correlation: Recently executed branches correlated with next branch.
  - Either different branches
  - Or different executions of same branches
- Tournament Predictor: more resources to competitive solutions and pick between them
- Branch Target Buffer: include branch address & prediction
- Predicated Execution can reduce number of branches, number of mispredicted branches
- Return address stack for prediction of indirect jump

Review: Limits of ILP

- 1985-2000: 1000X performance
  - Moore's Law transistors/chip => Moore's Law for Performance/MPU
- Hennessy: industry been following a roadmap of ideas known in 1985 to exploit Instruction Level Parallelism to get 1.55X/year
  - Caches, Pipelining, Superscalar, Branch Prediction, Out-of-order execution, …
- ILP limits: To make performance progress in future need to have explicit parallelism from programmer vs. implicit parallelism of ILP exploited by compiler, HW?
  - Otherwise drop to old rate of 1.3X per year?
  - Less because of processor-memory performance gap?
- Impact on you: if you care about performance, better think about explicitly parallel algorithms vs. rely on ILP?

Dynamic Scheduling in P6 (Pentium Pro, II, III)

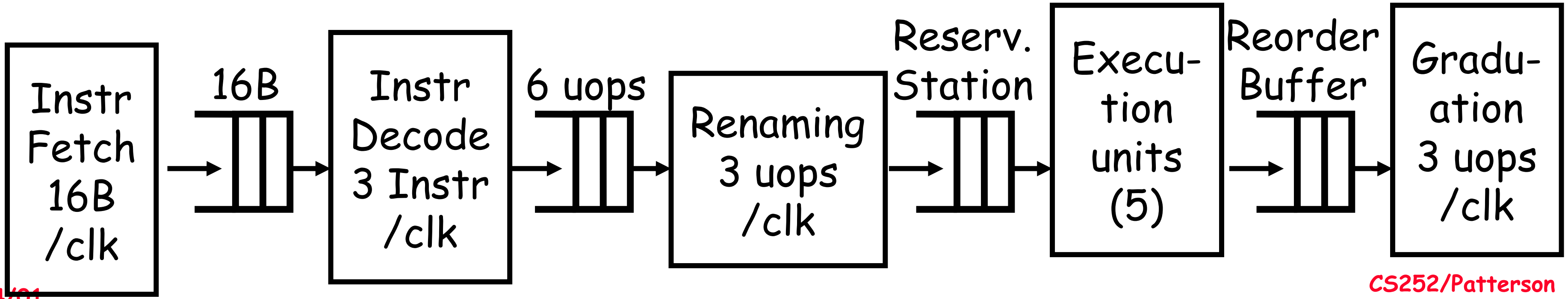
- Q: How pipeline 1 to 17 byte 80x86 instructions?
- P6 doesn't pipeline 80x86 instructions
- P6 decode unit translates the Intel instructions into 72-bit micro-operations (~ MIPS)
- Sends micro-operations to reorder buffer & reservation stations
- Many instructions translate to 1 to 4 micro-operations
- Complex 80x86 instructions are executed by a conventional microprogram (8K x 72 bits) that issues long sequences of micro-operations
- 14 clocks in total pipeline (~ 3 state machines)

Dynamic Scheduling in P6

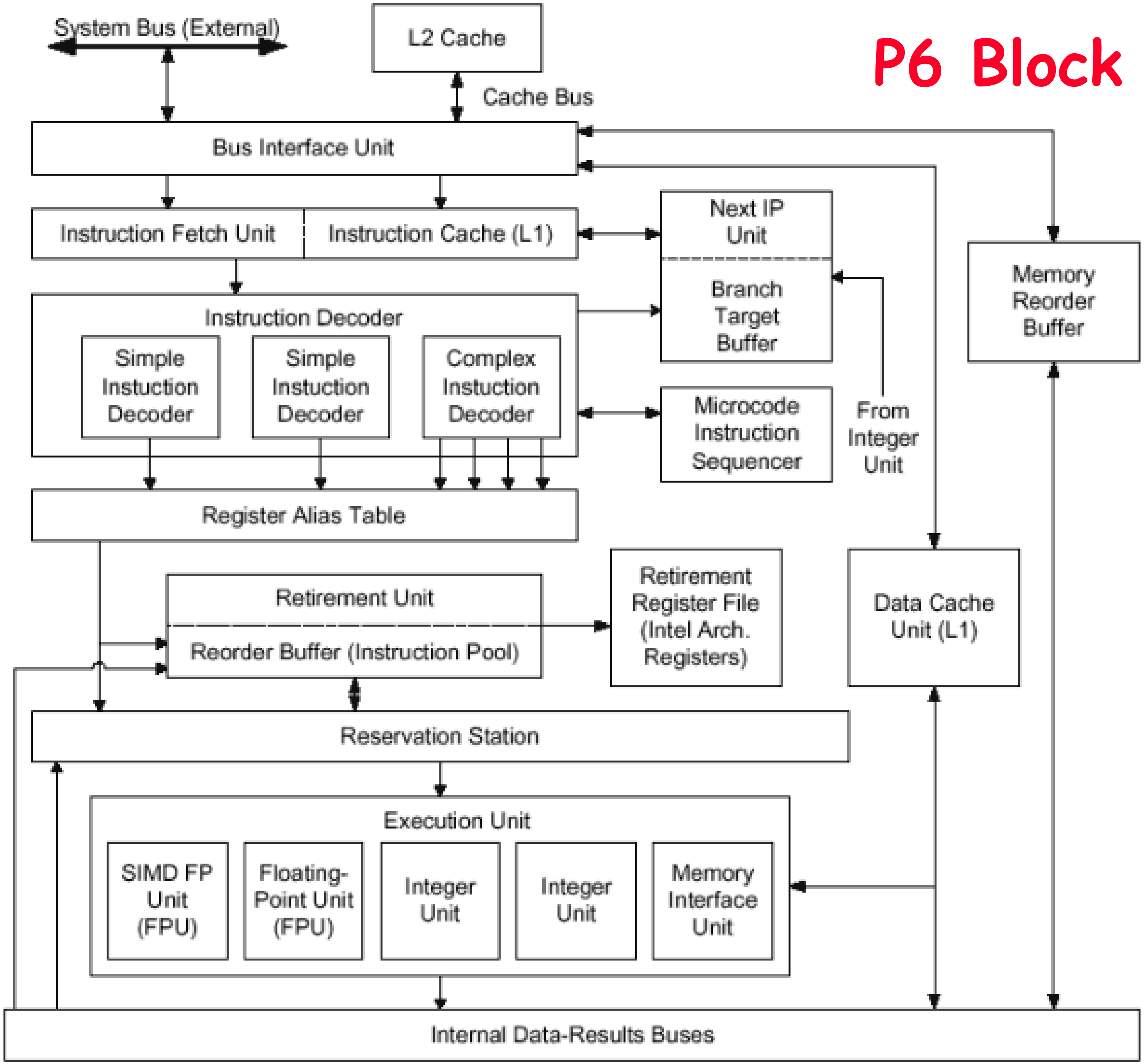
Parameter	80x86	microops
Max. instructions issued/clock	3	6
Max. instr. complete exec./clock		5
Max. instr. commited/clock		3
Window (Instrs in reorder buffer)		40
Number of reservations stations	20	
Number of rename registers	40	
No. integer functional units (FUs)	2	
No. floating point FUs	1	
No. SIMD Fl. Pt. FUs	1	
No. memory Fus	1 load + 1 store	

P6 Pipeline

- 14 clocks in total (~3 state machines)
- 8 stages are used for in-order instruction fetch, decode, and issue
  - Takes 1 clock cycle to determine length of 80x86 instructions + 2 more to create the micro-operations (uops)
- 3 stages are used for out-of-order execution in one of 5 separate functional units
- 3 stages are used for instruction commit



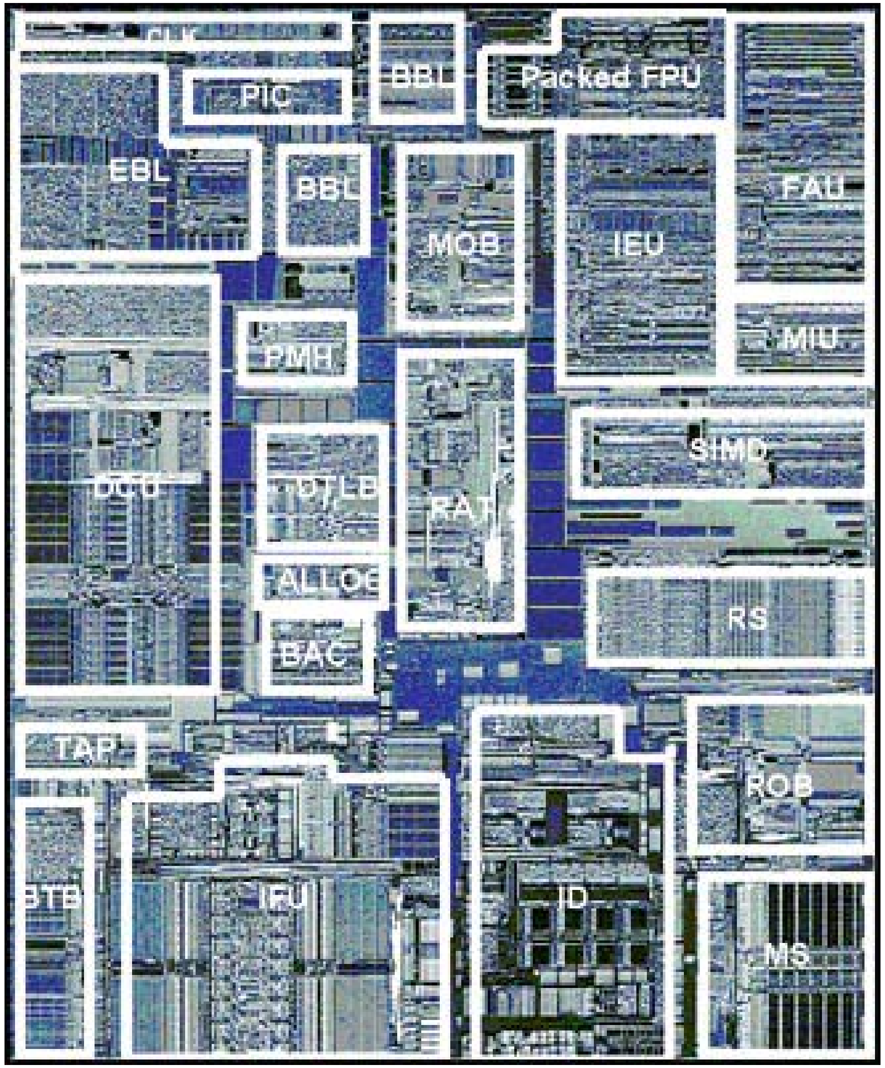
P6 Block Diagram



- IP = PC

From: http://www.digit-life.com/articles/pentium4/

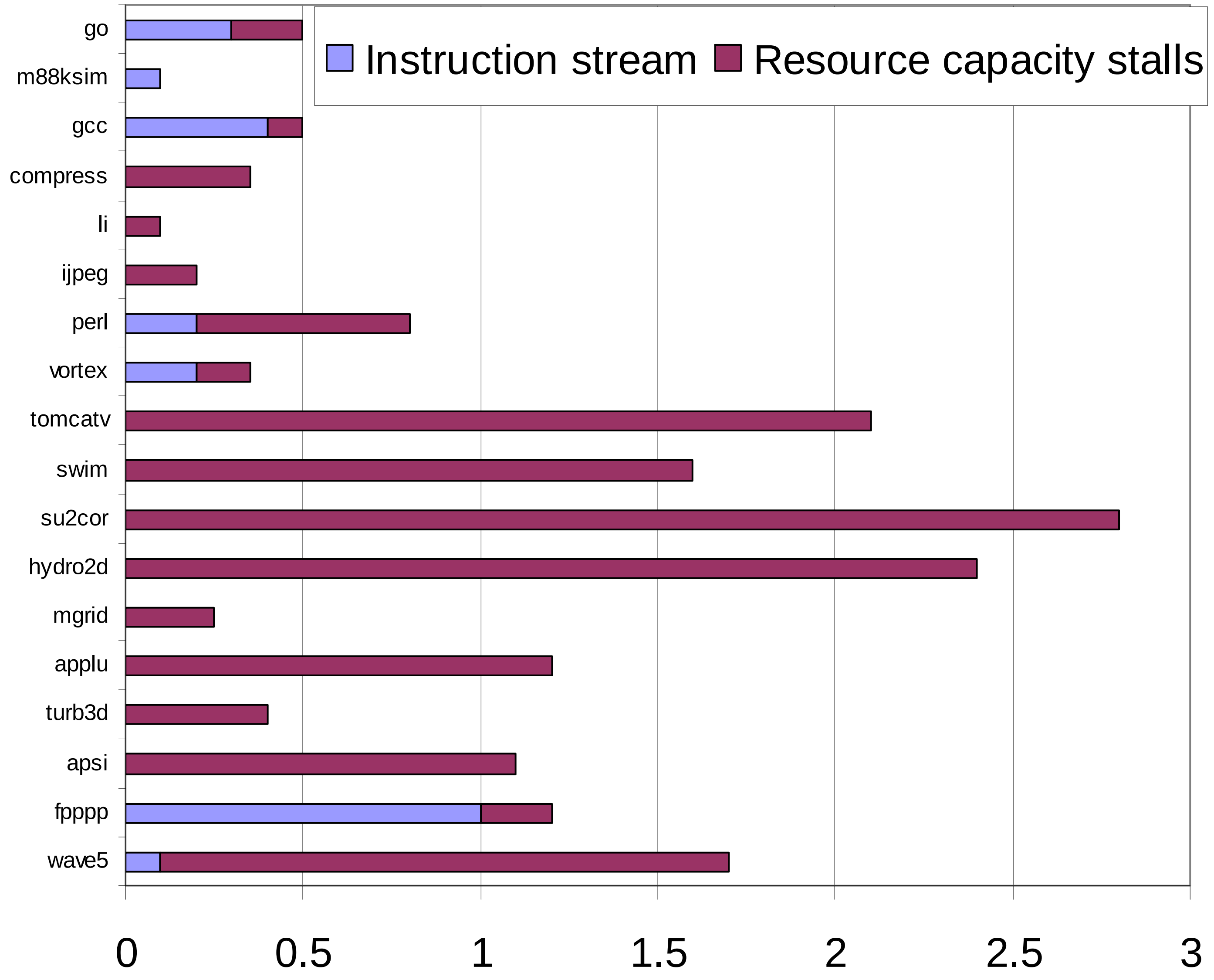
Pentium III Die Photo



- EBL/BBL - Bus logic, Front, Back
- MOB - Memory Order Buffer
- Packed FPU - MMX Fl. Pt. (SSE)
- IEU - Integer Execution Unit
- FAU - Fl. Pt. Arithmetic Unit
- MIU - Memory Interface Unit
- DCU - Data Cache Unit
- PMH - Page Miss Handler
- DTLB - Data TLB
- BAC - Branch Address Calculator
- RAT - Register Alias Table
- SIMD - Packed Fl. Pt.
- RS - Reservation Station
- BTB - Branch Target Buffer
- IFU - Instruction Fetch Unit (+I$)
- ID - Instruction Decode
- ROB - Reorder Buffer
- MS - Micro-instruction Sequencer

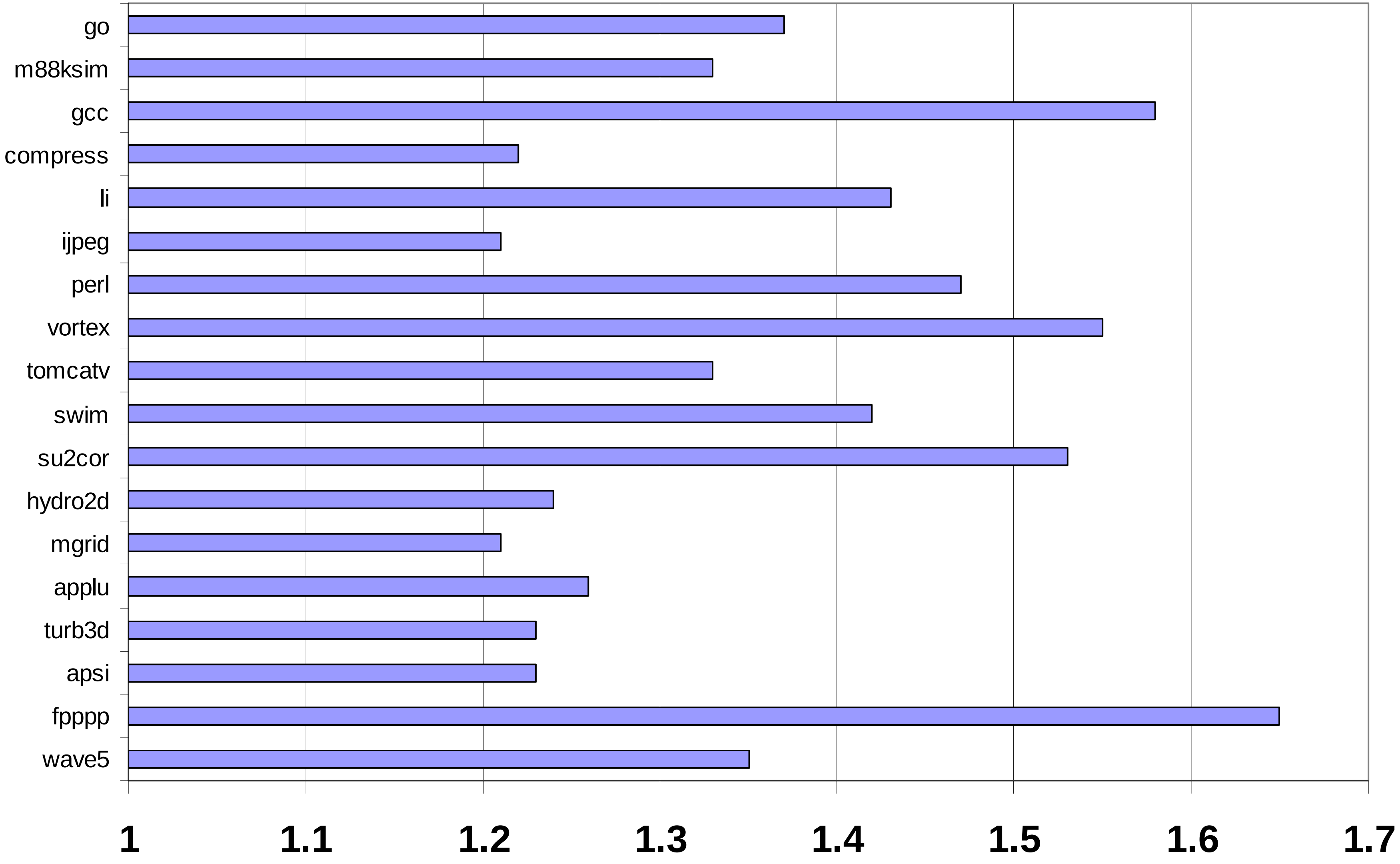
1st Pentium III, Katmai: 9.5 M transistors, 12.3 * 10.4 mm in 0.25-mi. with 5 layers of aluminum

P6 Performance: Stalls at decode stage I$ misses or lack of RS/Reorder buf. entry



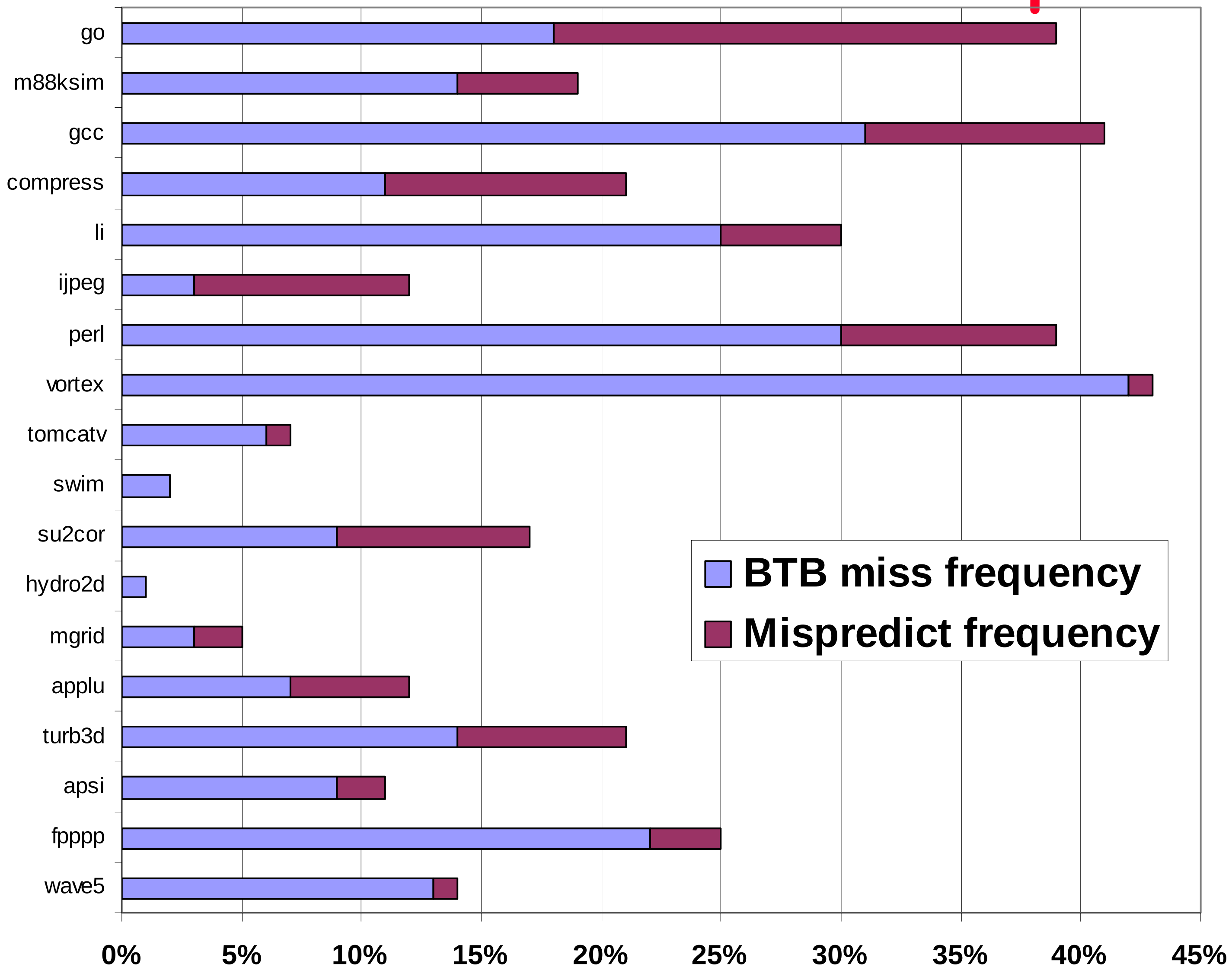
0.5 to 2.5 Stall cycles per instruction: 0.98 avg. (0.36 integer)

P6 Performance: uops/x86 instr
200 MHz, 8KI$/8KD$/256KL2$, 66 MHz bus



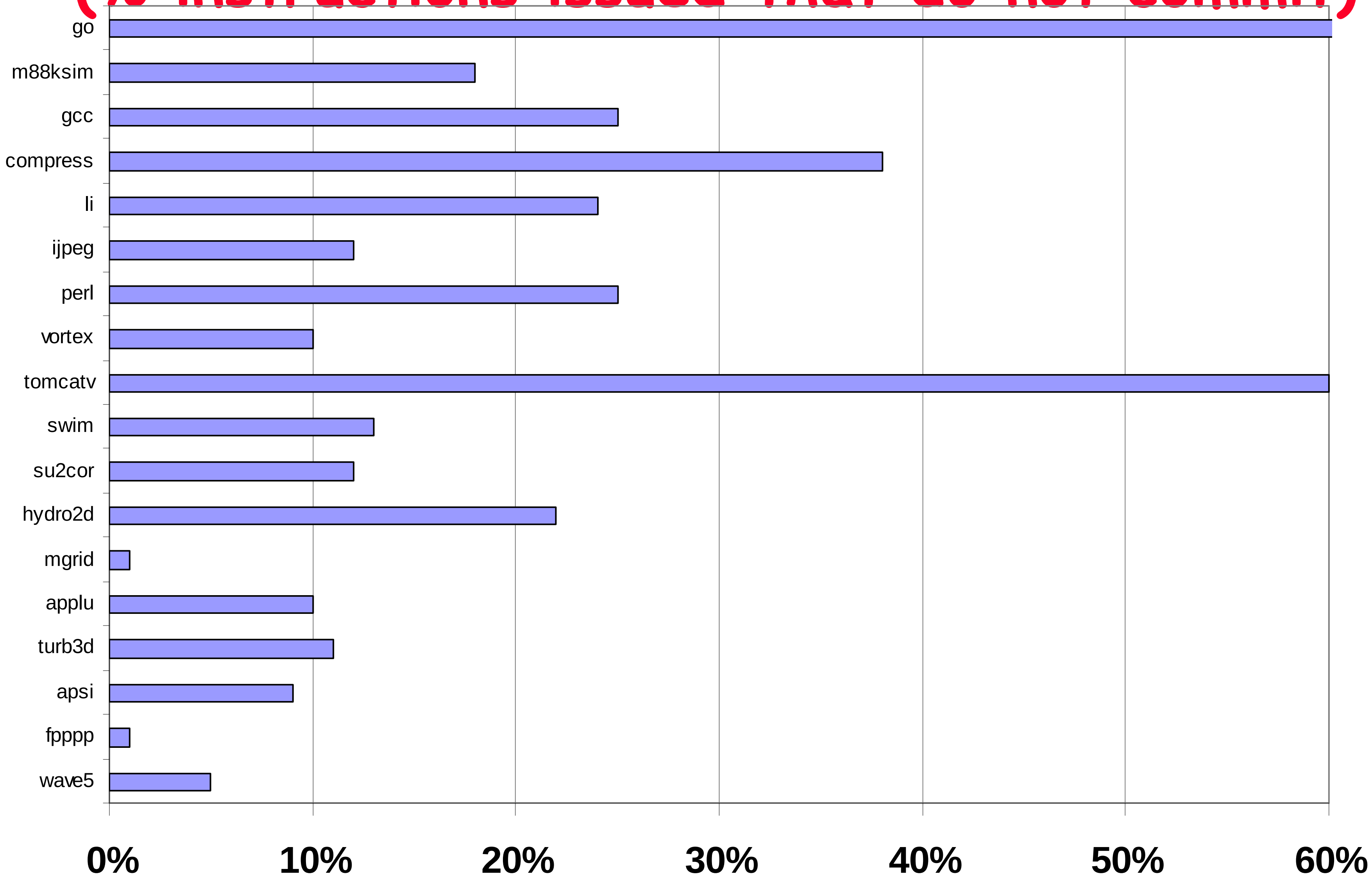
1.2 to 1.6 uops per IA-32 instruction: 1.36 avg. (1.37 integer)

P6 Performance: Branch Mispredict Rate



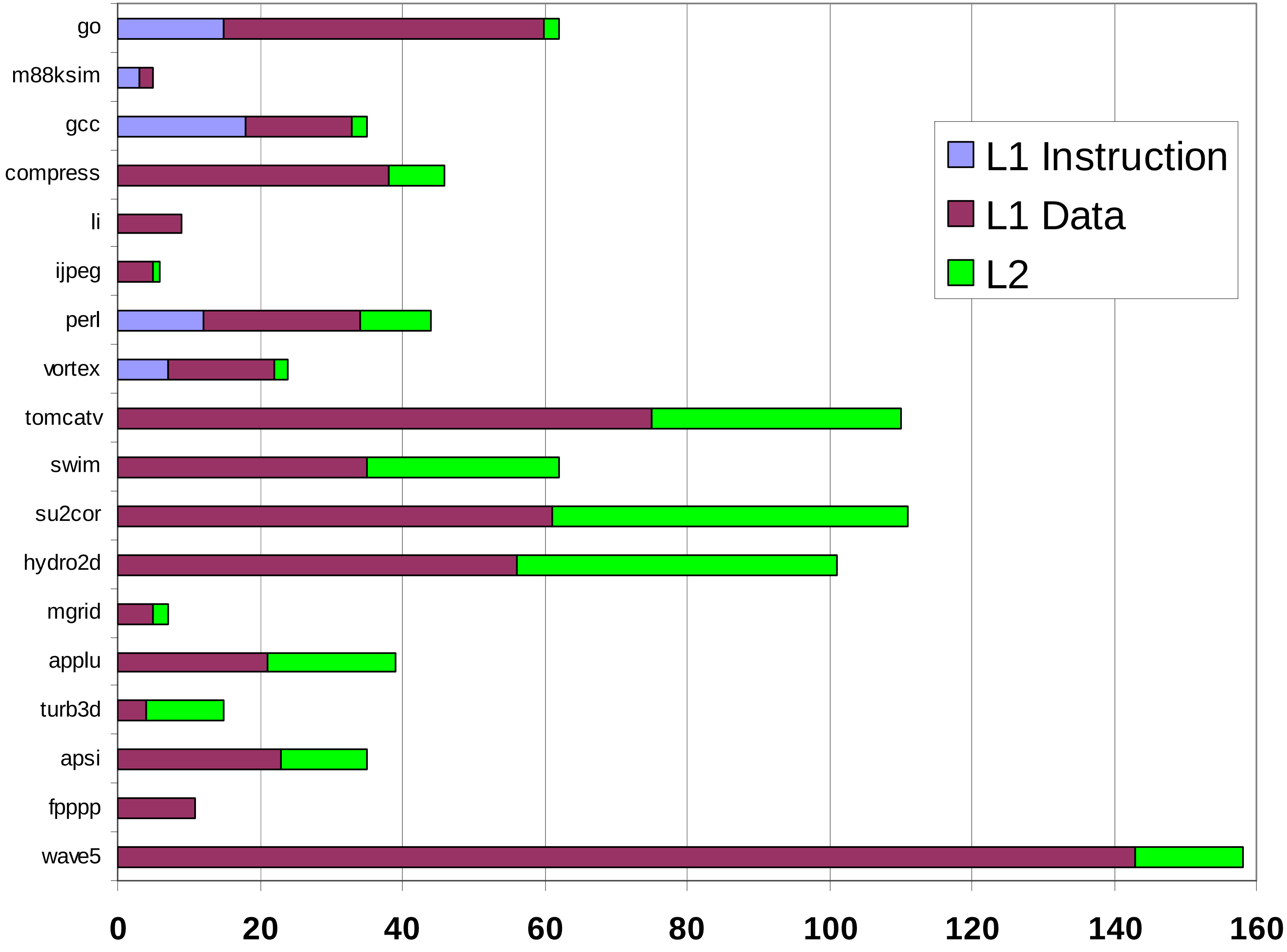
10% to 40% Miss/Mispredict ratio: 20% avg. (29% integer)

P6 Performance: Speculation rate (% instructions issued that do not commit)



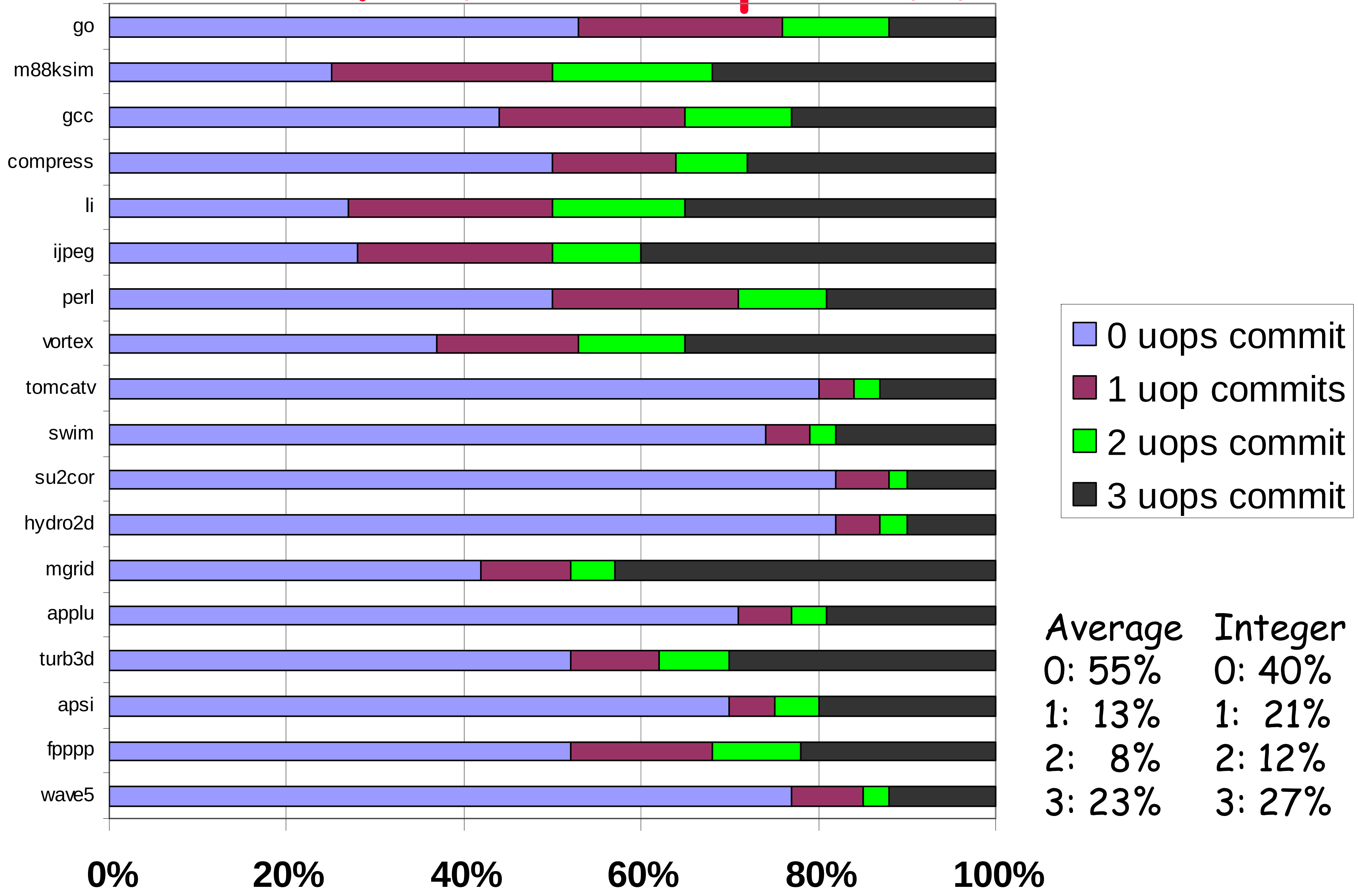
1% to 60% instructions do not commit: 20% avg (30% integer)

P6 Performance: Cache Misses/1k instr



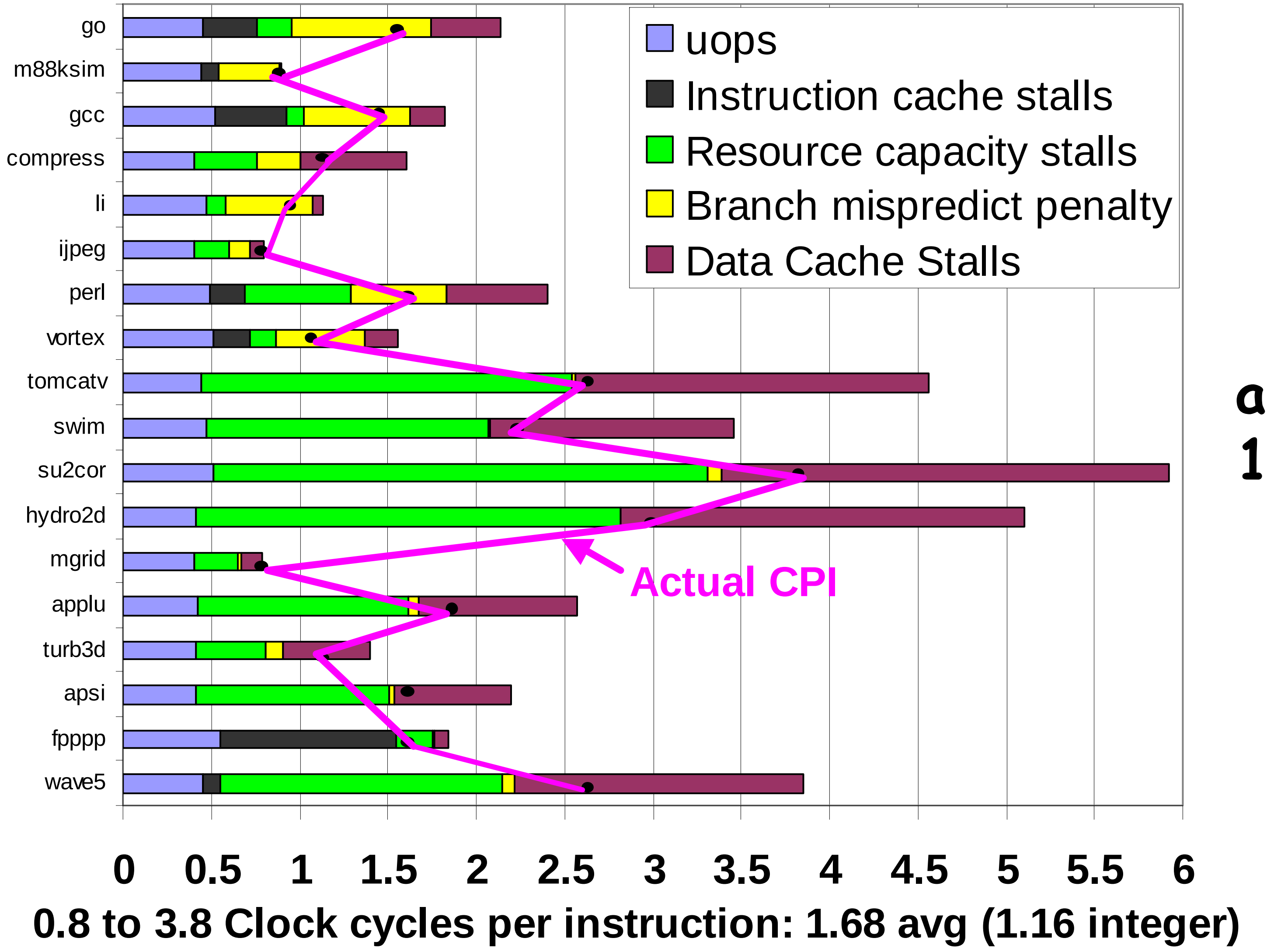
10 to 160 Misses per Thousand Instructions: 49 avg (30 integer)

P6 Performance: uops commit/clock



Average	Integer
0: 55%	0: 40%
1: 13%	1: 21%
2: 8%	2: 12%
3: 23%	3: 27%

P6 Dynamic Benefit?
Sum of parts CPI vs. Actual CPI



0.8 to 3.8 Clock cycles per instruction: 1.68 avg (1.16 integer)

Ratio of sum of parts vs. actual CPI: 1.38X avg. (1.29X integer)

Administratrivia

- 3rd (last) Homework on Ch 3 due Saturday
- 3rd project meetings 4/11
- Quiz #2 4/18 310 Soda at 5:30

AMD Althon

- Similar to P6 microarchitecture (Pentium III), but more resources
- Transistors: PIII 24M v. Althon 37M
- Die Size: 106 mm^2 v. 117 mm^2
- Power: 30W v. 76W
- Cache: 16K/16K/256K v. 64K/64K/256K
- Window size: 40 vs. 72 uops
- Rename registers: 40 v. 36 int +36 Fl. Pt.
- BTB: 512 x 2 v. 4096 x 2
- Pipeline: 10-12 stages v. 9-11 stages
- Clock rate: 1.0 GHz v. 1.2 GHz
- Memory bandwidth: 1.06 GB/s v. 2.12 GB/s

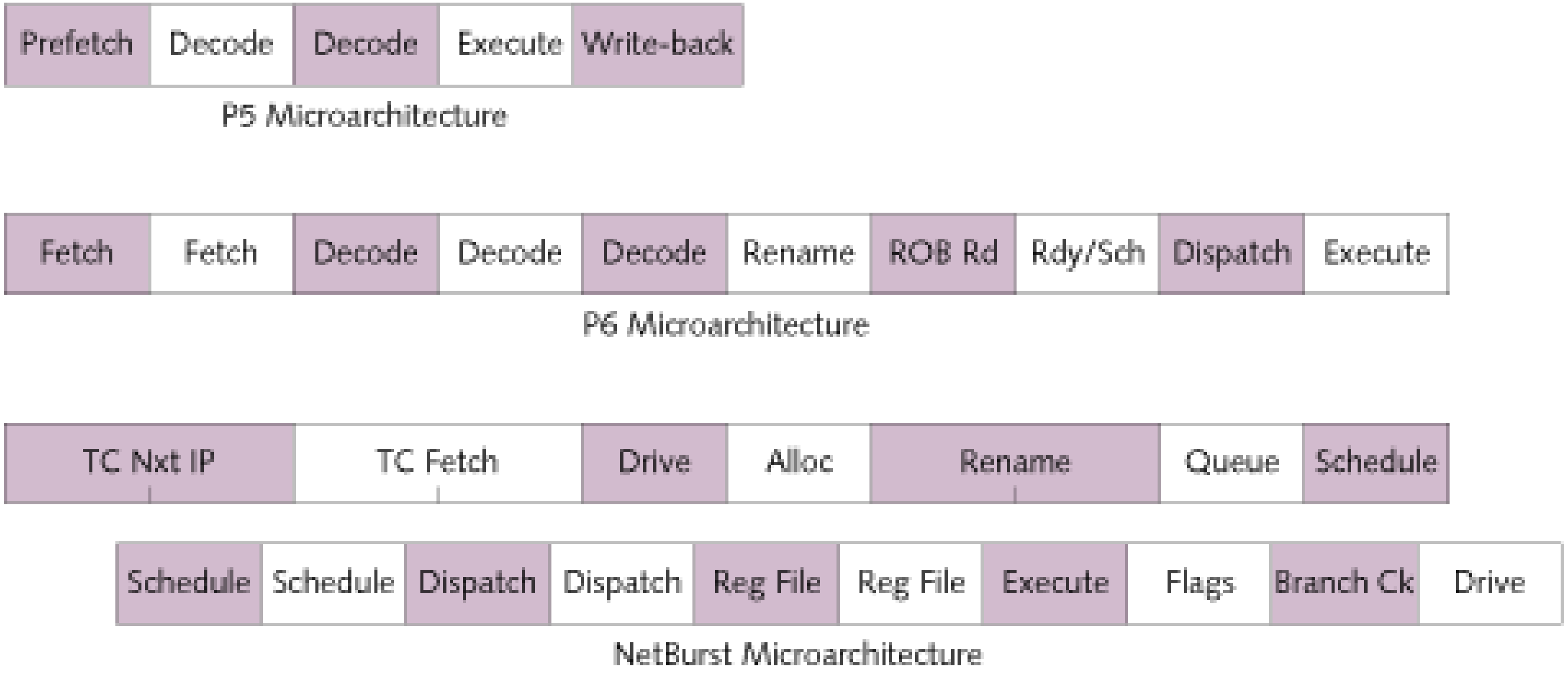
Pentium 4

- Still translate from 80x86 to micro-ops
- P4 has better branch predictor, more FUs
- Instruction Cache holds micro-operations vs. 80x86 instructions
  - no decode stages of 80x86 on cache hit
  - called "trace cache" (TC)
- Faster memory bus: 400 MHz v. 133 MHz
- Caches
  - Pentium III: L1I 16KB, L1D 16KB, L2 256 KB
  - Pentium 4: L1I 12K uops, L1D 8 KB, L2 256 KB
  - Block size: PIII 32B v. P4 128B; 128 v. 256 bits/clock
- Clock rates:
  - Pentium III 1 GHz v. Pentium IV 1.5 GHz
  - 14 stage pipeline vs. 24 stage pipeline

Pentium 4 features

- Multimedia instructions 128 bits wide vs. 64 bits wide => 144 new instructions
  - When used by programs??
  - Faster Floating Point: execute 2 64-bit Fl. Pt. Per clock
  - Memory FU: 1 128-bit load, 1 128-store /clock to MMX regs
- Using RAMBUS DRAM
  - Bandwidth faster, latency same as SDRAM
  - Cost 2X-3X vs. SDRAM
- ALUs operate at 2X clock rate for many ops
- Pipeline doesn't stall at this clock rate: uops replay
- Rename registers: 40 vs. 128; Window: 40 v. 126
- BTB: 512 vs. 4096 entries (Intel: 1/3 improvement)

Pentium, Pentium Pro, Pentium 4 Pipeline



Prefetch	Decode	Decode	Execute	Write-back

P5 Microarchitecture

Fetch	Fetch	Decode	Decode	Decode	Rename	ROB Rd	Rdy/Sch	Dispatch	Execute

P6 Microarchitecture

TC Nxt IP	TC Fetch	Drive	Alloc	Rename	Queue	Schedule

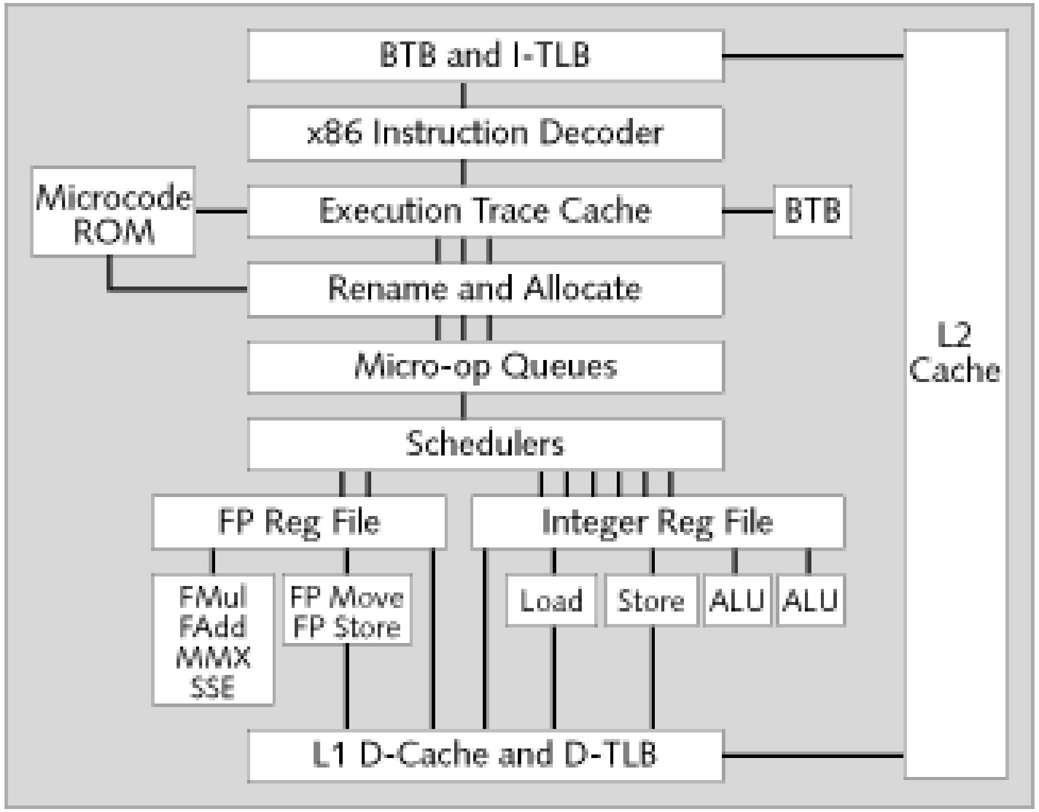
Schedule	Schedule	Dispatch	Dispatch	Reg File	Reg File	Execute	Flags	Branch Ck	Drive

NetBurst Microarchitecture

- Pentium (P5) = 5 stages
  Pentium Pro, II, III (P6) = 10 stages (1 cycle ex)
  Pentium 4 (NetBurst) = 20 stages (no decode)

From "Pentium 4 (Partially) Previewed," Microprocessor Report, 8/28/00

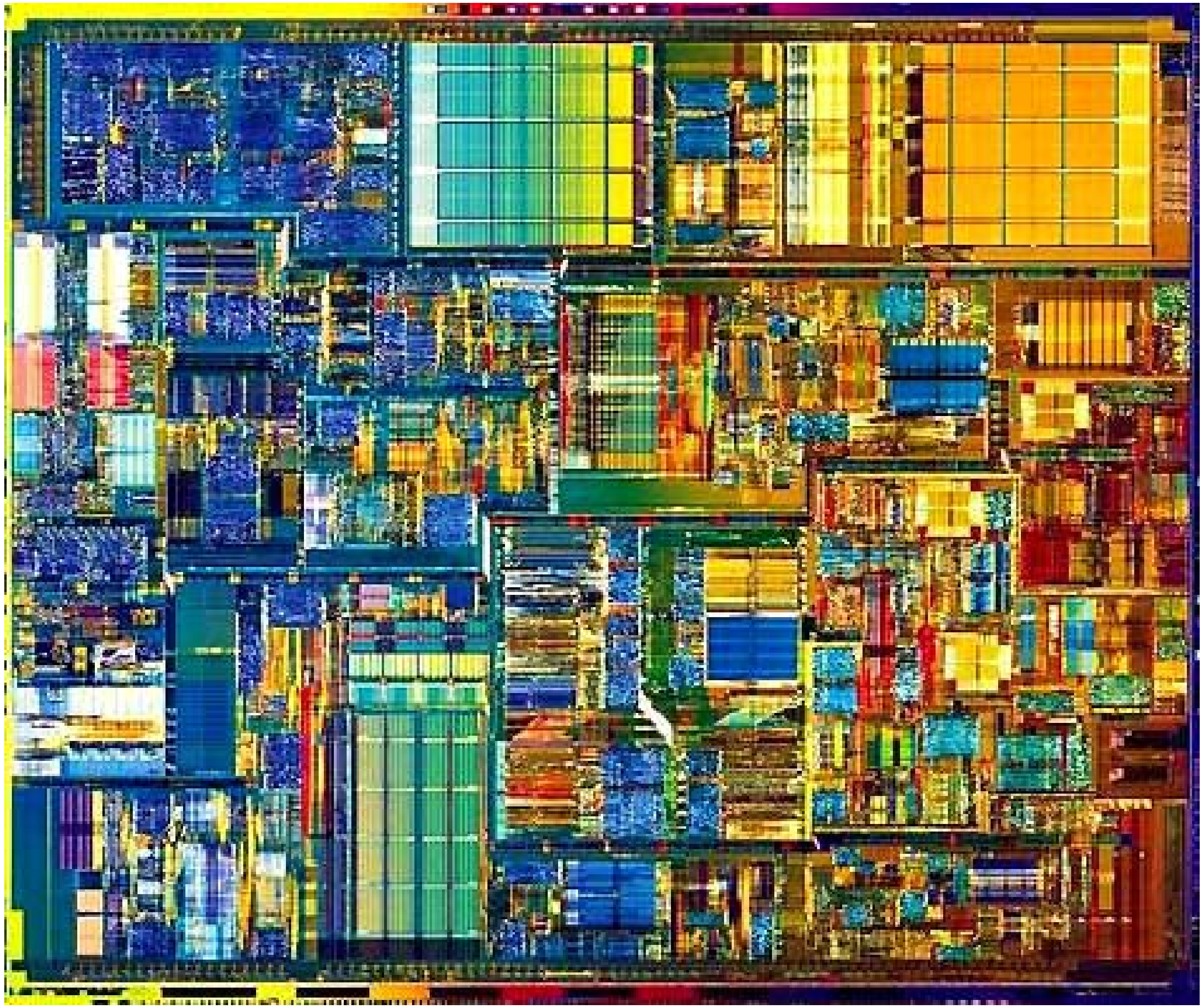
Block Diagram of Pentium 4 Microarchitecture



- BTB = Branch Target Buffer (branch predictor)
- I-TLB = Instruction TLB, Trace Cache = Instruction cache
- RF = Register File; AGU = Address Generation Unit
- "Double pumped ALU" means ALU clock rate 2X => 2X ALU F.U.s

From "Pentium 4 (Partially) Previewed," Microprocessor Report, 8/28/00

Pentium 4 Die Photo



- 42M Xtors
  - PIII: 26M
- 217 mm^2
  - PIII: 106 mm^2
- L1 Execution Cache
  - Buffer 12,000 Micro-Ops
- 8KB data cache
- 256KB L2$

Benchmarks: Pentium 4 v. PIII v. Althon

- SPECbase2000
  - Int, P4@1.5 GHz: 524, PIII□@1GHz: 454, AMD Althon@1.2Ghz:?
  - FP, P4@1.5 GHz: 549, PIII□@1GHz: 329, AMD Althon@1.2Ghz:304
- WorldBench 2000 benchmark (business) PC World magazine, Nov. 20, 2000 (bigger is better)
  - P4 : 164, PIII : 167, AMD Althon: 180
- Quake 3 Arena: P4 172, Althon 151
- SYSmark 2000 composite: P4 209, Althon 221
- Office productivity: P4 197, Althon 209
- S.F. Chronicle 11/20/00: "… the challenge for AMD now will be to argue that frequency is not the most important thing-- precisely the position Intel has argued while its Pentium III lagged behind the Athlon in clock speed."

Why?

- Instruction count is the same for x86
- Clock rates: P4 > Althon > PIII
- How can P4 be slower?
- Time =
  Instruction count x CPI x 1/Clock rate
- Average Clocks Per Instruction (CPI) of P4 must be worse than Althon, PIII
- Will CPI ever get < 1.0 for real programs?

Another Approach: Mulithreaded Execution for Servers

- Thread: process with own instructions and data
  - thread may be a process part of a parallel program of multiple processes, or it may be an independent program
  - Each thread has all the state (instructions, data, PC, register state, and so on) necessary to allow it to execute
- Multithreading: multiple threads to share the functional units of 1 processor via overlapping
  - processor must duplicate indepedent state of each thread e.g., a separate copy of register file and a separate PC
  - memory shared through the virtual memory mechanisms
- Threads execute overlapped, often interleaved
  - When a thread is stalled, perhaps for a cache miss, another thread can be executed, improving throughput

Multithreaded Example: IBM AS/400

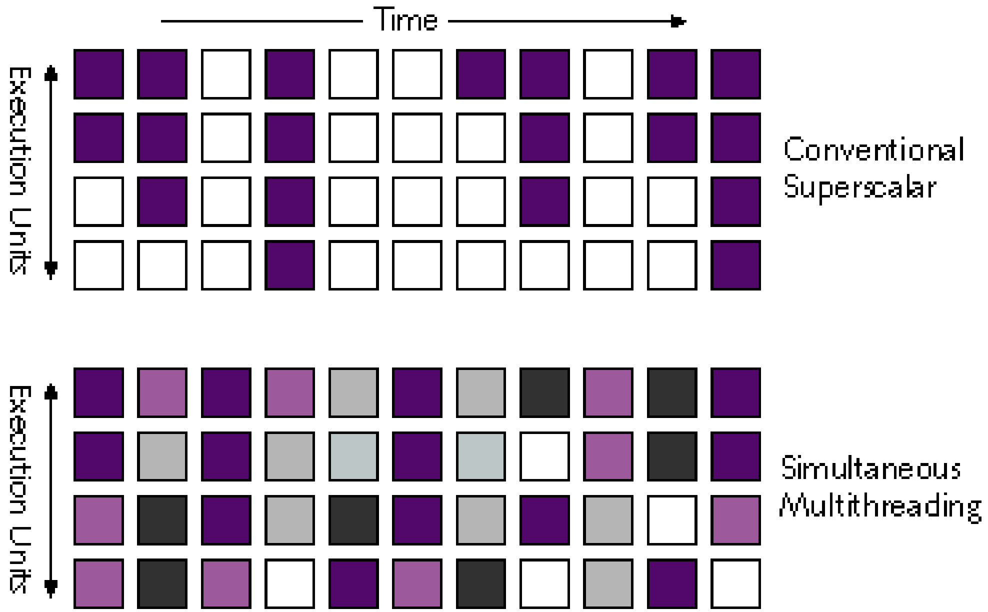
- IBM Power III processor, " Pulsar"
  - PowerPC microprocessor that supports 2 IBM product lines: the RS/6000 series and the AS/400 series
  - Both aimed at commercial servers and focus on throughput in common commercial applications
  - such applications encounter high cache and TLB miss rates and thus degraded CPI
- include a multithreading capability to enhance throughput and make use of the processor during long TLB or cache-miss stall
- Pulsar supports 2 threads: little clock rate, silicon impact
- Thread switched only on long latency stall

Multithreaded Example: IBM AS/400

- Pulsar: 2 copies of register files & PC
- < 10% impact on die size
- Added special register for max no. clock cycles between thread switches:
  - Avoid starvation of other thread

Simultaneous Multithreading (SMT)

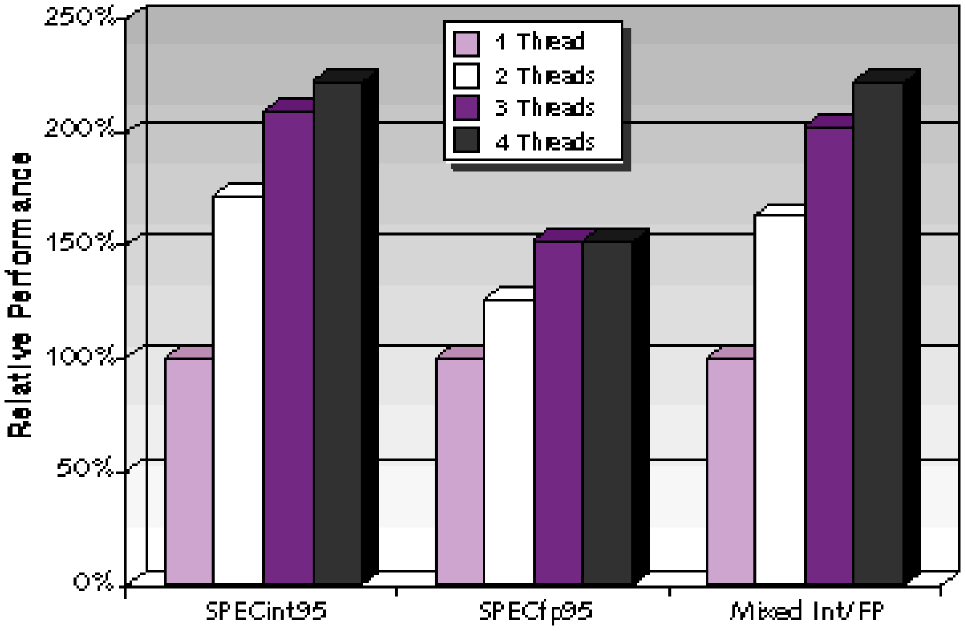
- Simultaneous multithreading (SMT): insight that dynamically scheduled processor already has many HW mechanisms to support multithreading
  - large set of virtual registers that can be used to hold the register sets of independent threads (assuming separate renaming tables are kept for each thread)
  - out-of-order completion allows the threads to execute out of order, and get better utilization of the HW



Source: Micrprocessor Report, December 6, 1999
“Compaq Chooses SMT for Alpha”

SMT is coming

- Just adding a per thread renaming table and keeping separate PCs
  - Independent commitment can be supported by logically keeping a separate reorder buffer for each thread
- Compaq has announced it for future Alpha microprocessor: 21464 in 2003; others likely



On a multiprogramming workload comprising a mixture of SPECint95 and SPECfp95 benchmarks, Compaq claims the SMT it simulated achieves a 2.25X higher throughput with 4 simultaneous threads than with just 1 thread. For parallel programs, 4 threads 1.75X v. 1

Source: Micrprocessor Report, December 6, 1999
"Compaq Chooses SMT for Alpha"