Microprocessor Evolution: 4004 to Pentium-4

Joel Emer
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology

Based on the material prepared by
Krste Asanovic and Arvind

November 2, 2005

First Microprocessor Intel 4004, 1971

Image removed due to copyright restrictions.
To view image, visit
http://news.com.com/Images+Moores+Law+turns+40/2009-1041_3-5649019-5.html

- 4-bit accumulator architecture
- 8μm pMOS
- 2,300 transistors
- 3 x 4 mm^2
- 750kHz clock
- 8-16 cycles/inst.

Microprocessors in the Seventies

Initial target was embedded control

- First micro, 4-bit 4004 from Intel, designed for a desktop printing calculator

Constrained by what could fit on single chip

- Single accumulator architectures

8-bit micros used in hobbyist personal computers

- Micral, Altair, TRS-80, Apple-II

Little impact on conventional computer market until VISICALC spreadsheet for Apple-II (6502, 1MHz)

- First “killer” business application for personal computers

DRAM in the Seventies

Dramatic progress in MOSFET memory technology

1970, Intel introduces first DRAM (1Kbit 1103)

1979, Fujitsu introduces 64Kbit DRAM

=> By mid-Seventies, obvious that PCs would soon have > 64KBytes physical memory

Microprocessor Evolution

Rapid progress in size and speed through 70s
- Fueled by advances in MOSFET technology and expanding markets

Intel i432
- Most ambitious seventies' micro; started in 1975 - released 1981
- 32-bit capability-based object-oriented architecture
- Instructions variable number of *bits* long
- Severe performance, complexity, and usability problems

Motorola 68000 (1979, 8MHz, 68,000 transistors)
- Heavily microcoded (and nanocoded)
- 32-bit general purpose register architecture (24 address pins)
- 8 address registers, 8 data registers

Intel 8086 (1978, 8MHz, 29,000 transistors)
- "Stopgap" 16-bit processor, architected in 10 weeks
- Extended accumulator architecture, assembly-compatible with 8080
- 20-bit addressing through segmented addressing scheme

Intel 8086

Class	*Register*	*Purpose*
Data:	AX,BX	"general" purpose
	CX	string and loop ops only
	DX	mult/div and I/O only
Address:	SP	stack pointer
	BP	base pointer (can also use BX)
	SI,DI	index registers
Segment:	CS	code segment
	SS	stack segment
	DS	data segment
	ES	extra segment
Control:	IP	instruction pointer (low 16 bit of PC)
	FLAGS	C, Z, N, B, P, V and 3 control bits

- Typical format R <= R op M[X], many addressing modes
- *Not a GPR organization!*

IBM PC, 1981

Hardware

- Team from IBM building PC prototypes in 1979
- Motorola 68000 chosen initially, but 68000 was late
- IBM builds "stopgap" prototypes using 8088 boards from Display Writer word processor
- 8088 is 8-bit bus version of 8086 => allows cheaper system
- Estimated sales of 250,000
- 100,000,000s sold

Software

- Microsoft negotiates to provide OS for IBM. Later buys and modifies QDOS from Seattle Computer Products.

Open System

- Standard processor, Intel 8088
- Standard interfaces
- Standard OS, MS-DOS
- IBM permits cloning and third-party software

The Eighties: Personal Computer Revolution

Personal computer market emerges

- Huge business and consumer market for spreadsheets, word processing and games
- Based on inexpensive 8-bit and 16-bit micros: Zilog Z80, Mostek 6502, Intel 8088/86, …

Minicomputers replaced by workstations

- Distributed network computing and high-performance graphics for scientific and engineering applications (Sun, Apollo, HP,…)
- Based on powerful 32-bit microprocessors with virtual memory, caches, pipelined execution, hardware floating-point
- Commercial RISC processors developed for workstation market

Massively Parallel Processors (MPPs) appear

- Use many cheap micros to approach supercomputer performance (Sequent, Intel, Parsytec)

The Nineties

Advanced superscalar microprocessors appear
- first superscalar microprocessor is IBM POWER in 1990

MPPs have limited success in supercomputing market
- Highest-end mainframes and vector supercomputers survive “killer micro” onslaught

64-bit addressing becomes essential at high-end
- In 2004, 4GB DRAM costs <$1,000

Parallel microprocessor-based SMPs take over low-end server and supercomputer market

Workstation and PC markets merge
- By late ‘90s (except for Apple PowerPC-based systems) RISC vendors have tiny share of desktop market
- CISC x86 ISA thrives!

Intel Pentium 4 (2000)

This lecture contains figures and data taken from: "The microarchitecture of the Pentium 4 processor", Intel Technology Journal, Q1, 2001

Pentium 4 uOPs

- During L1 instruction cache refill, translates complex x86 instructions into RISC-like micro-operations (uops)
  - e.g., “R ← R op Mem” translates into

    load T, Mem # Load from Mem into temp reg
    R ← R op T # Operate using value in temp

- Execute uops using speculative out-of-order superscalar engine with register renaming
- uop translation introduced in Pentium Pro family architecture (P6 family) in 1995
  - also used on Pentium-II and Pentium-III processors, and new Pentium M (Centrino) processors

Instruction Set Translation:
Convert a target ISA into a host machine's ISA

- Pentium Pro (P6 family)
  - translation in hardware after instruction fetch
  - also used in AMD x86 processors
- Pentium-4 family
  - translation in hardware at level 1 instruction cache refill
- Transmeta Crusoe
  - translation in software using "Code Morphing" (see lecture 24)

Pentium 4 Block Diagram

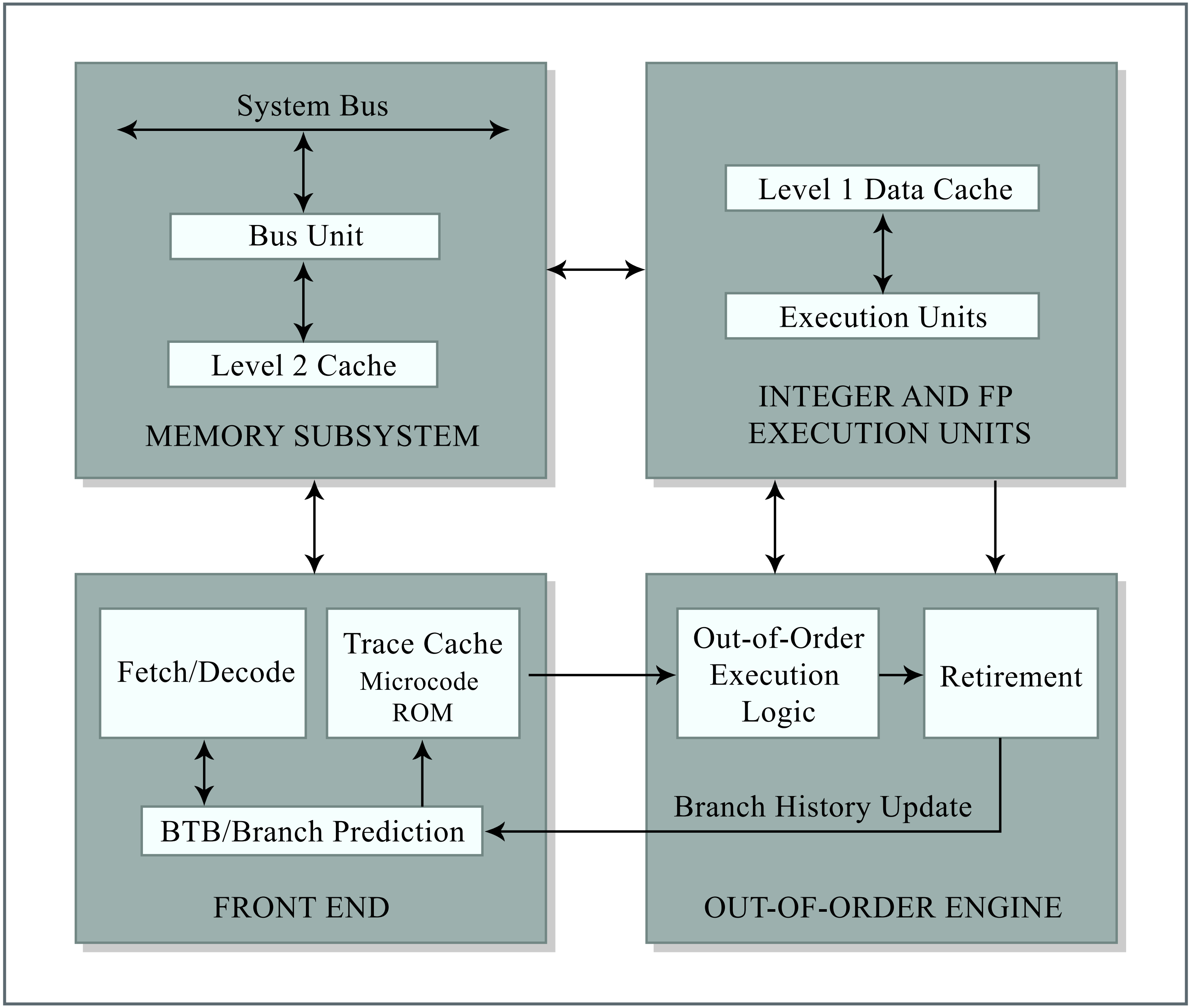
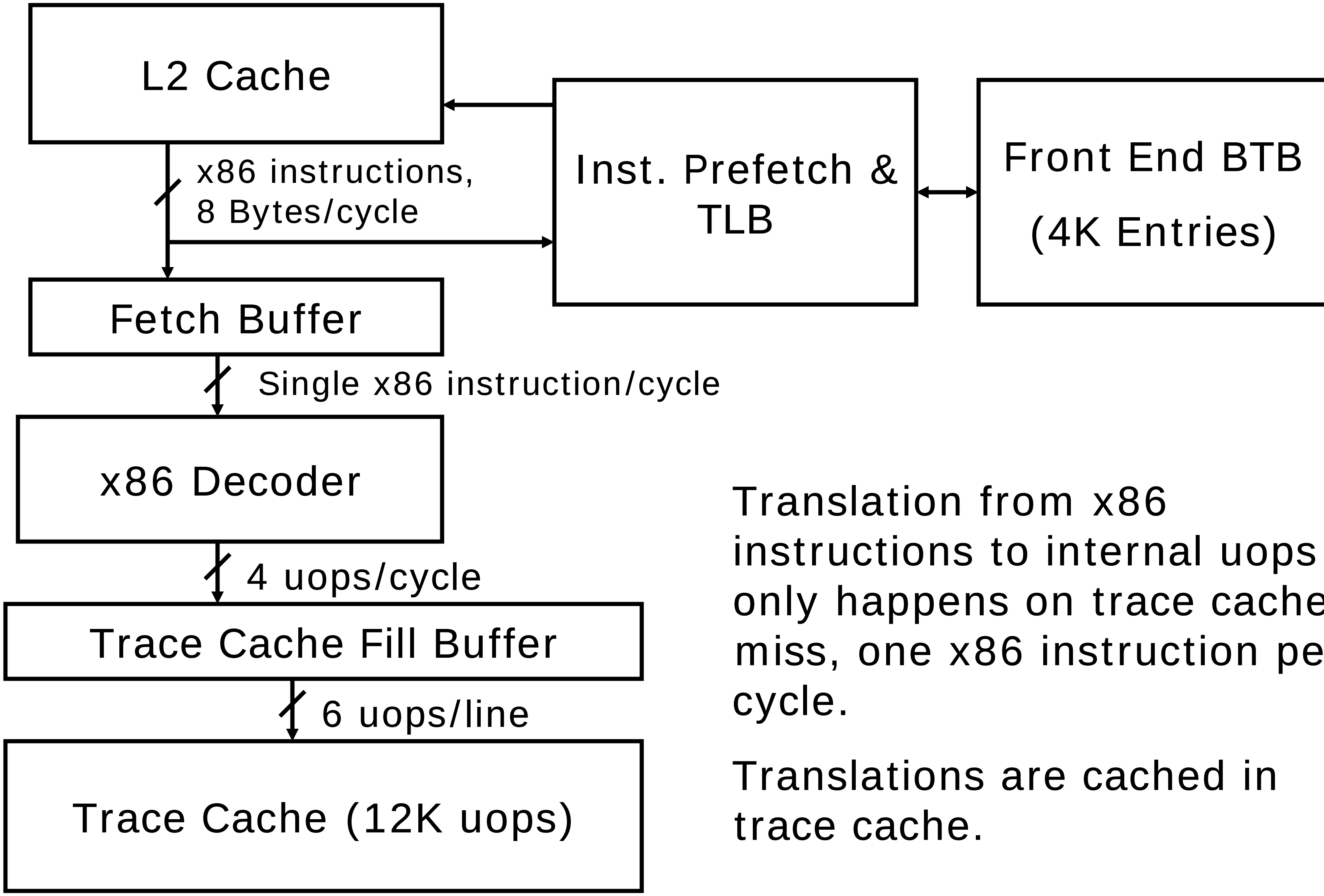


Figure by MIT OCW.

Pentium 4 Front End

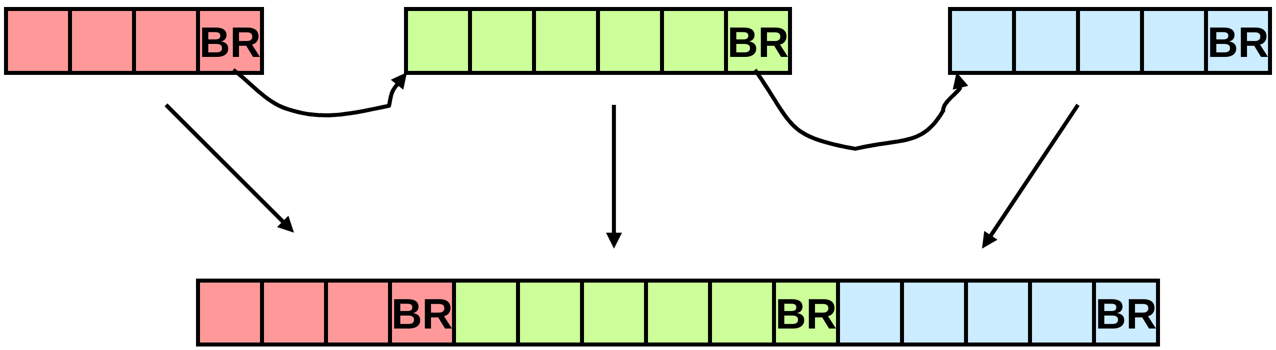


Translation from x86 instructions to internal uops only happens on trace cache miss, one x86 instruction per cycle.

Translations are cached in trace cache.

Trace Cache

Key Idea: Pack multiple non-contiguous basic blocks into one contiguous trace cache line



- Single fetch brings in multiple basic blocks
- Trace cache indexed by start address *and* next *n* branch predictions

Pentium 4 Trace Cache

- Holds decoded uops in predicted program flow order, 6 uops per line

Code in memory

```
     cmp
     br T1
     ...
T1:  sub
     br T2
     ...
T2:  mov
     sub
     br T3
     ...
T3:  add
     sub
     mov
     br T4
     ...
T4:
```

Code packed in trace cache (6 uops/line)

cmp	br T1	sub
br T2	mov	sub

br T3	add	sub
mov	br T4	T4:...

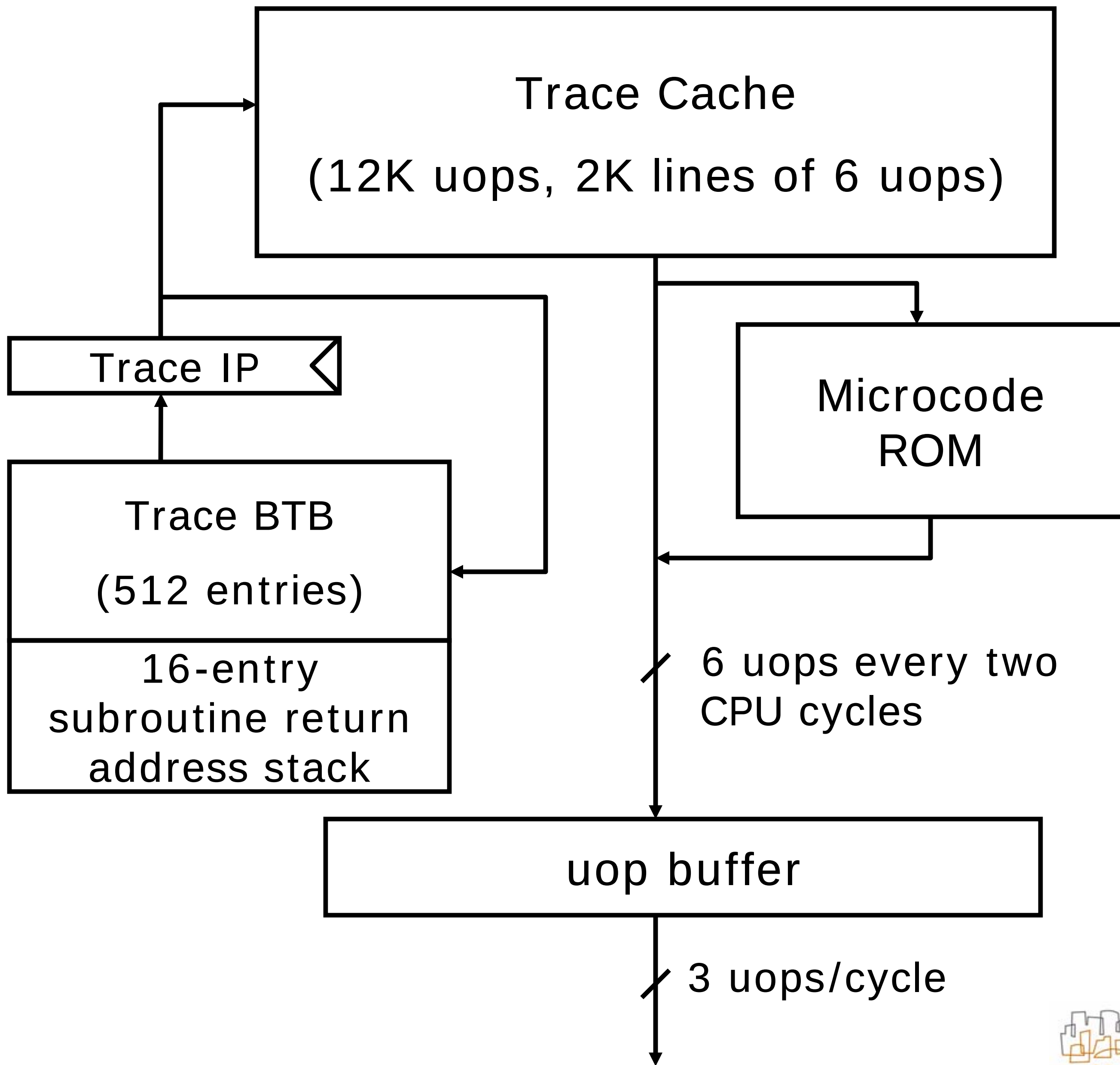
Trace cache fetches one 6 uop line every 2 CPU clock cycles (runs at 1/2 main CPU rate)

Trace Cache Advantages

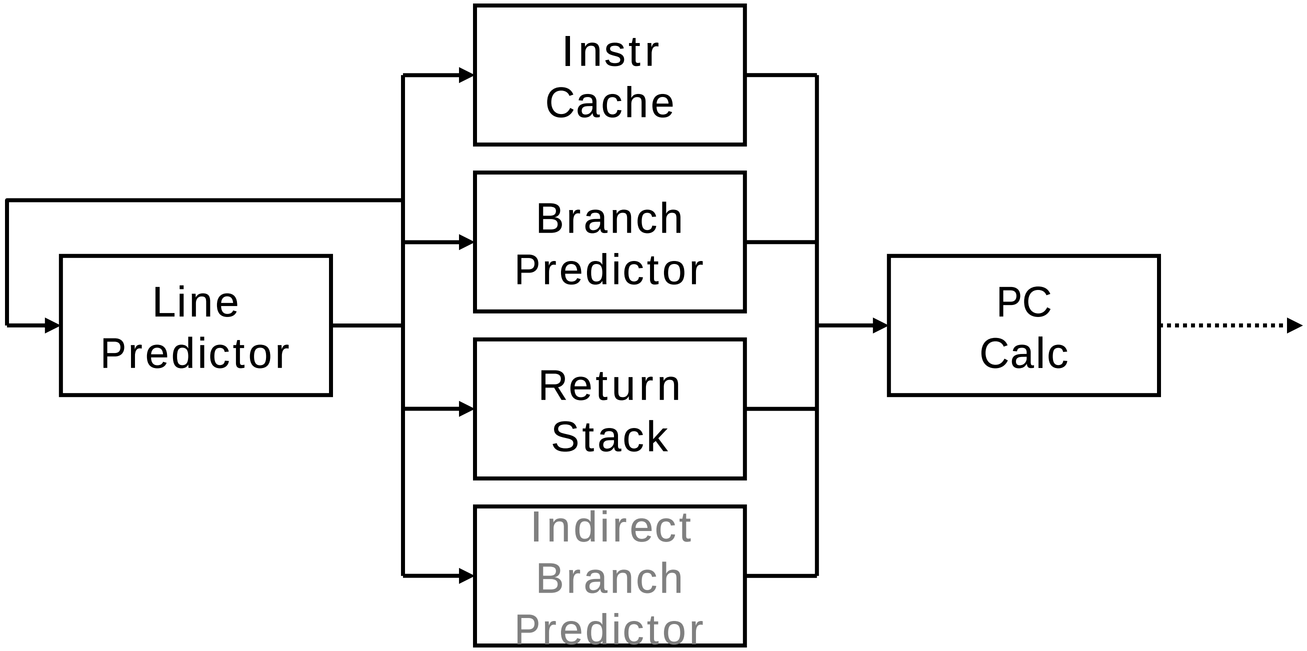
- Removes x86 decode from branch mispredict penalty
  - Parallel x86 decoder took 2.5 cycles in P6, would be 5 cycles in P-4 design
- Allows higher fetch bandwidth fetch for correctly predicted taken branches
  - P6 had one cycle bubble for correctly predicted taken branches
  - P-4 can fetch a branch and its target in same cycle
- Saves energy
  - x86 decoder only powered up on trace cache refill

P-4 Trace Cache Fetch

1	TC Next IP (BTB)
2	
3	TC Fetch
4	
5	Drive
6	Alloc
7	Rename
8	
9	Queue
10	Schedule 1
11	Schedule 2
12	Schedule 3
13	Dispatch 1
14	Dispatch 2
15	Register File 1
16	Register File 2
17	Execute
18	Flags
19	Branch Check
20	Drive



Line Prediction
(*Alpha 21[234]64*)



- Line Predictor predicts line to fetch each cycle
  - 21464 was to predict 2 lines per cycle
- Icache fetches block, and predictors predict target
- PC Calc checks accuracy of line prediction(s)

P-III vs. P-4 Renaming

1	TC Next IP
2	(BTB)
3	TC Fetch
4	
5	Drive
	Alloc
	Rename
9	Queue
10	Schedule 1
11	Schedule 2
12	Schedule 3
13	Dispatch 1
14	Dispatch 2
15	Register File 1
16	Register File 2
17	Execute
18	Flags
19	Branch Check
20	Drive

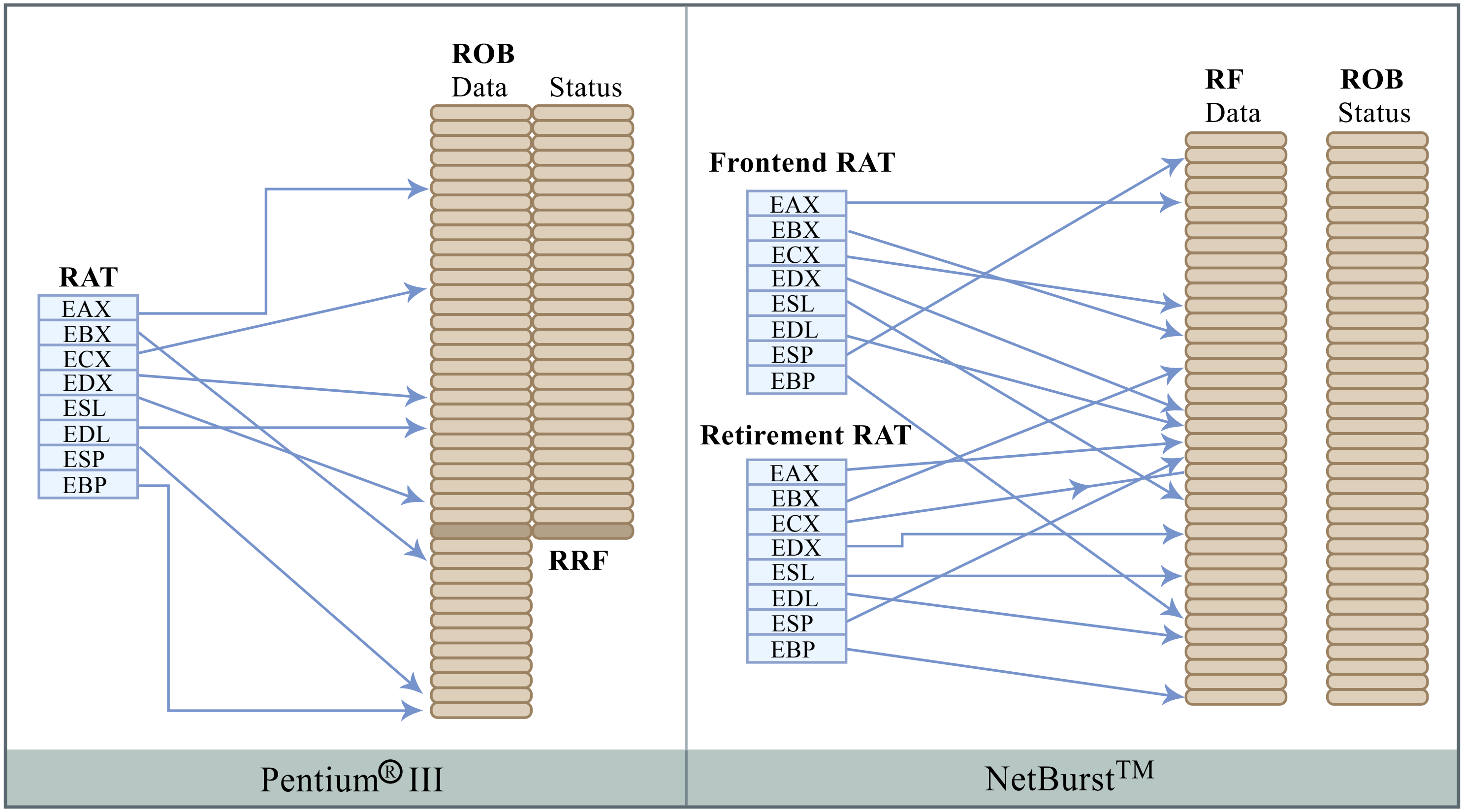
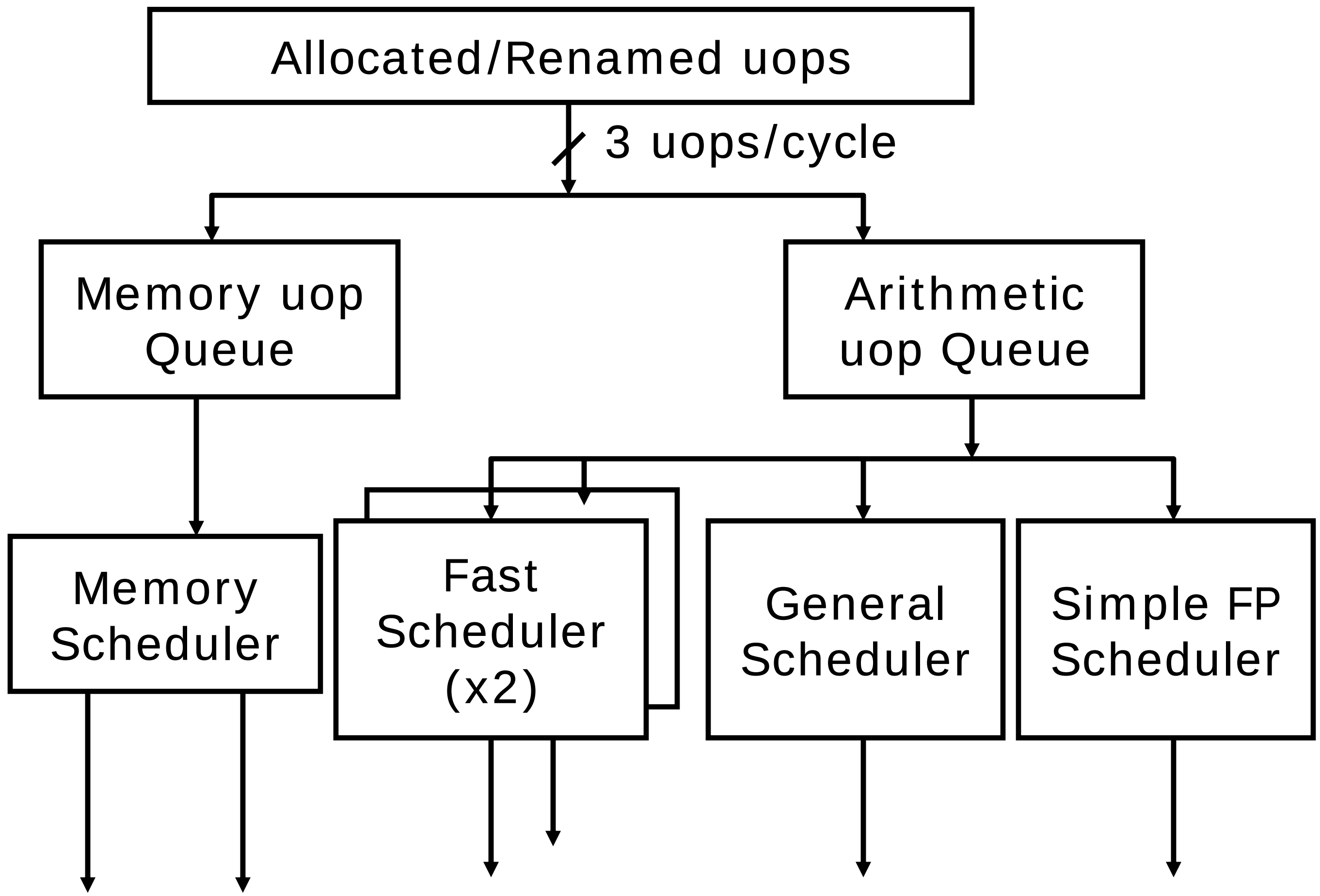


Figure by MIT OCW.

P-4 physical register file separated from ROB status.
ROB entries allocated sequentially as in P6 family.
One of 128 physical registers allocated from free list.
No data movement on retire, only Retirement RAT updated.

P-4 mOp Queues and Schedulers

1	TC Next IP (BTB)
2	
3	TC Fetch
4	
5	Drive
6	Alloc
7	Rename
8	
	Queue
	Schedule 1
	Schedule 2
	Schedule 3
	Dispatch 1
	Dispatch 2
15	Register File 1
16	Register File 2
17	Execute
18	Flags
19	Branch Check
20	Drive



Ready uops compete for dispatch ports

(Fast schedulers can each dispatch 2 ALU operations per cycle)

P-4 Execution Ports

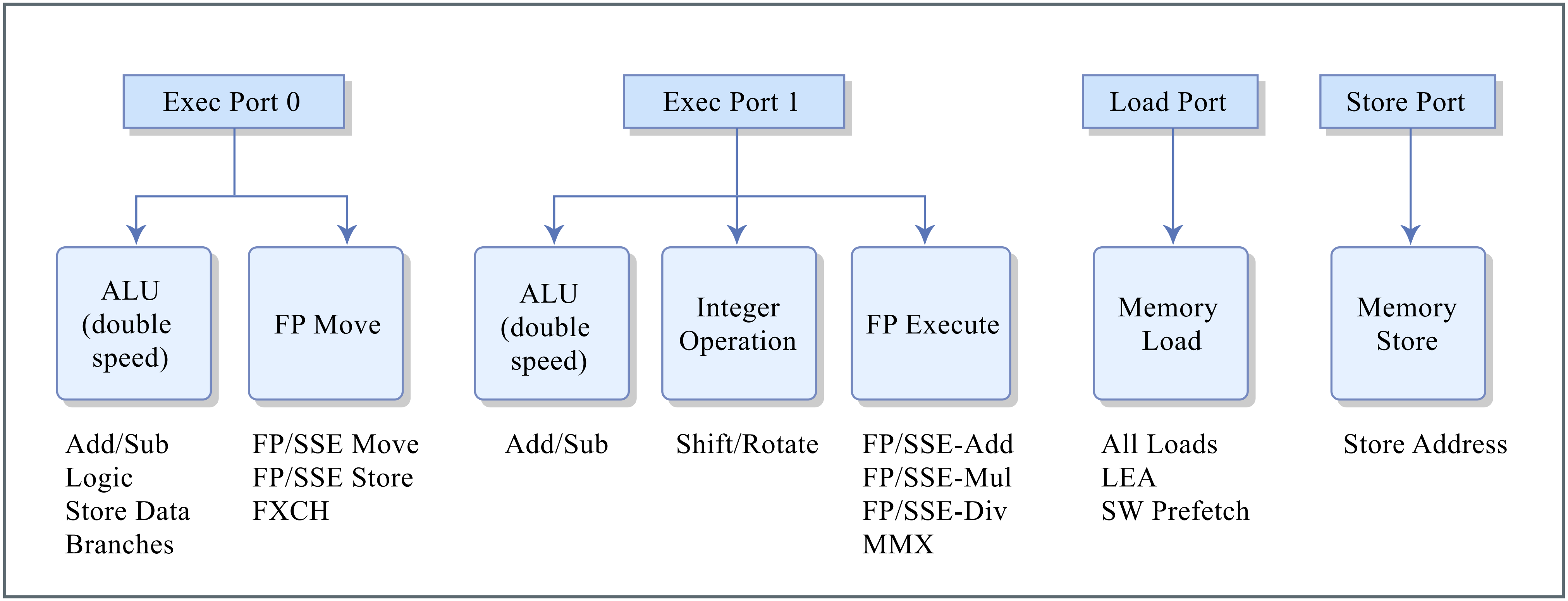
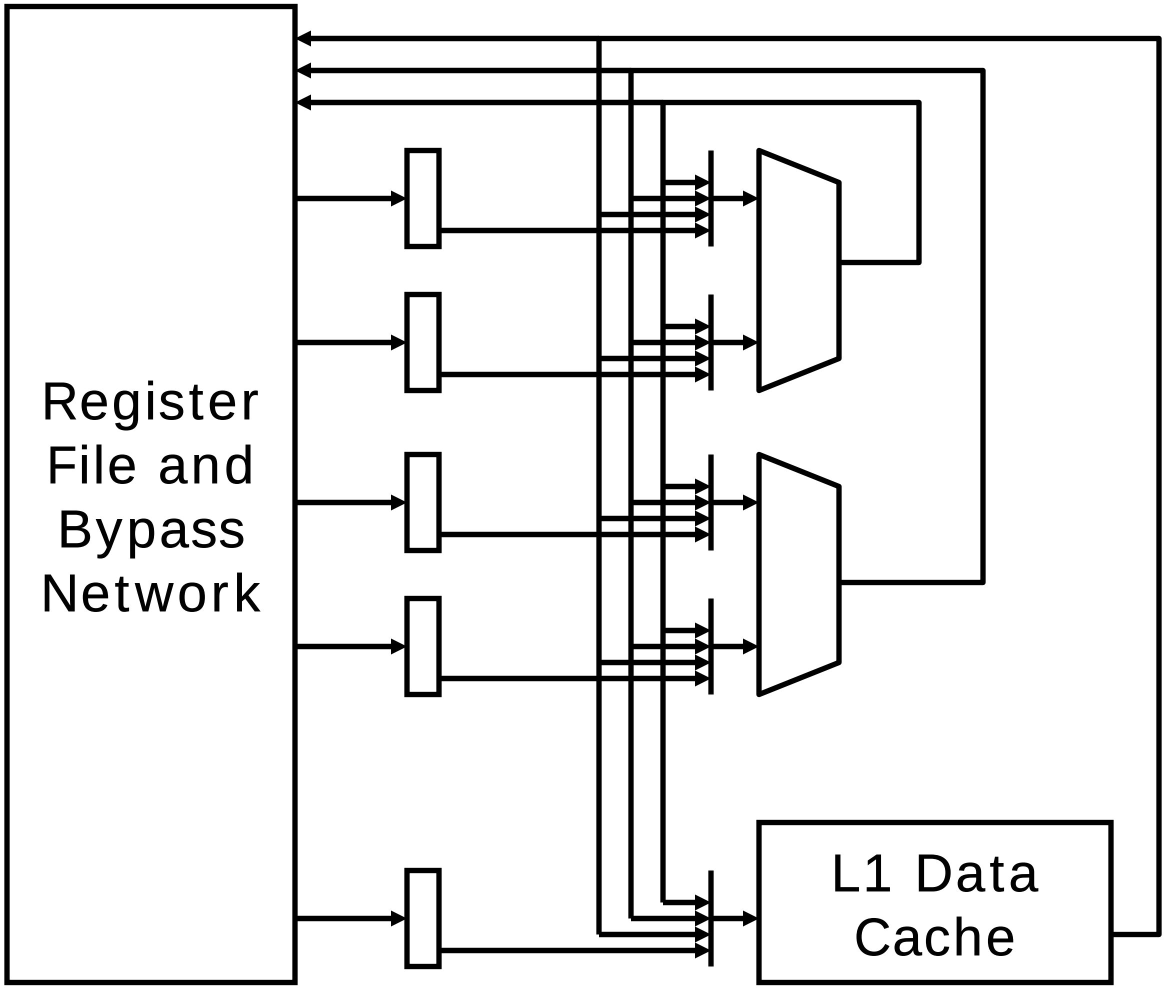


Figure by MIT OCW.

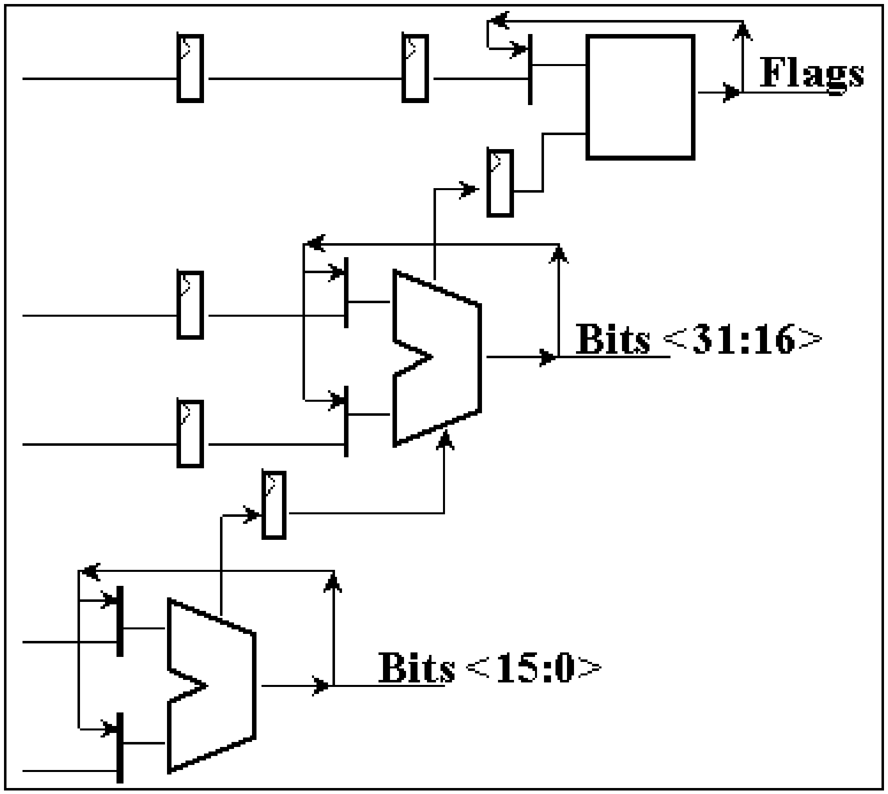
- **Schedulers compete for access to execution ports**
- **Loads and stores have dedicated ports**
- **ALUs can execute two operations per cycle**
- **Peak bandwidth of 6 uops per cycle**
  - **load, store, plus four double-pumped ALU operations**

P-4 Fast ALUs and Bypass Path



- Fast ALUs and bypass network runs at twice global clock speed
- All "non-essential" circuit paths handled out of loop to reduce circuit loading (shifts, mults/divs, branches, flag/ops)
- Other bypassing takes multiple clock cycles

P-4 Staggered ALU Design



- Staggers 32-bit add and flag compare into three 1/2 cycle phases
  - low 16 bits
  - high 16 bits
  - flag checks

- Bypass 16 bits around every ½ cycle
  - back-to-back dependent 32-bit adds at 3GHz in 0.18mm (7.2GHz in 90nm)

- L1 Data Cache access starts with bottom 16 bits as index, top 16 bits used as tag check later

P-4 Load Schedule Speculation

1	TC Next IP
2	
3	TC Fetch
4	
5	Drive
6	Alloc
7	Rename
8	
9	Queue
	Schedule 1
	Schedule 2
	Schedule 3
	Dispatch 1
	Dispatch 2
	Register File 1
	Register File 2
	Load Execute 1
	Load Execute 2
19	Branch Check
20	**Drive**

Long delay from schedulers to load hit/miss

- P-4 guesses that load will hit in L1 and schedules dependent operations to use value
- If load misses, only dependent operations are replayed

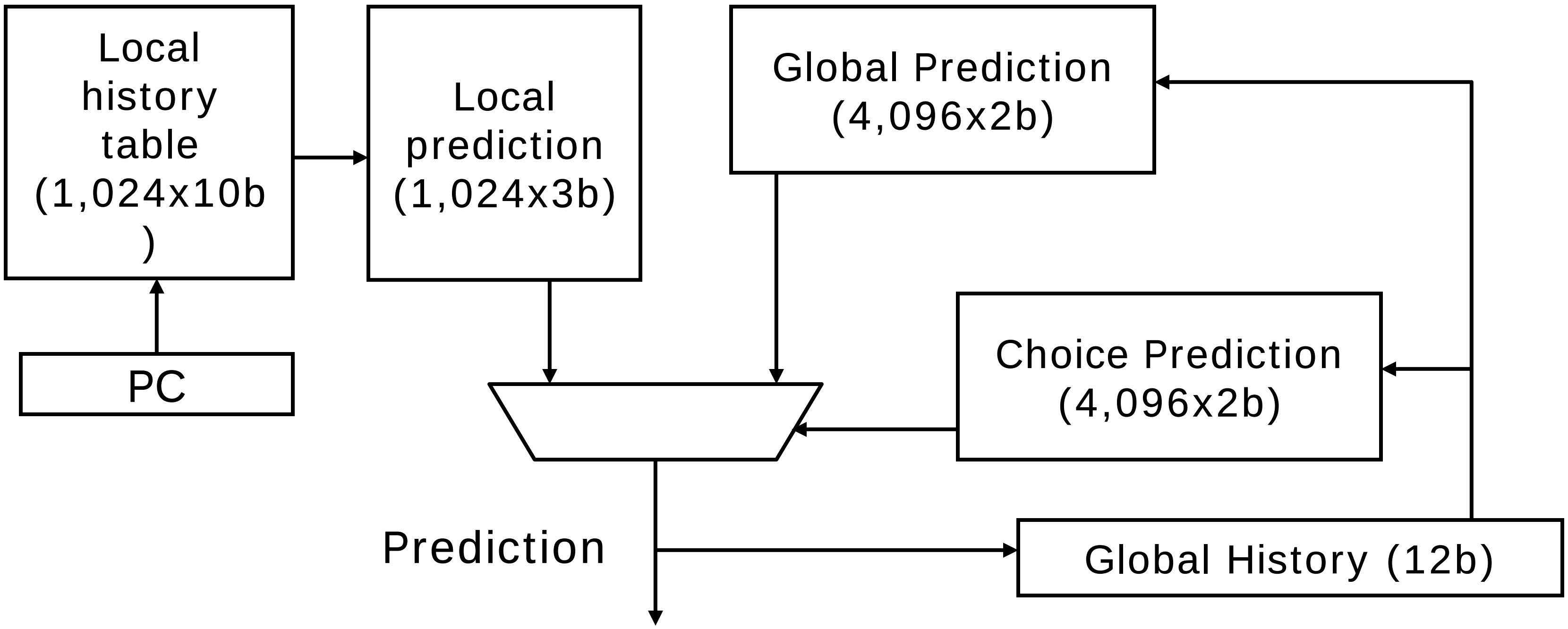
P-4 Branch Penalty

1	TC Next IP
2	
3	TC Fetch
4	
5	Drive
6	Alloc
7	Rename
8	
9	Queue
10	Schedule 1
11	Schedule 2
12	Schedule 3
13	Dispatch 1
14	Dispatch 2
15	Register File 1
16	Register File 2
17	Execute
18	Flags
19	Branch Check
20	Drive

20 cycle branch mispredict penalty

- P-4 uses new “trade secret” branch prediction algorithm
- Intel claims 1/3 fewer mispredicts than P6 algorithm

Tournament Branch Predictor
(Alpha 21264)



- Choice predictor learns whether best to use local or global branch history in predicting next branch
- Global history is speculatively updated but restored on mispredict
- Claim 90-100% success on range of applications

P-III vs. P-4 Pipelines

Basic Pentium® III Processor Misprediction Pipeline

1	2	3	4	5	6	7	8	9	10
Fetch	Fetch	Decode	Decode	Decode	Rename	ROB Rd	Rdy/Sch	Dispatch	Exec

Basic Pentium® 4 Processor Misprediction Pipeline

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

Figure by MIT OCW.

- In same process technology, ~1.5x clock frequency
- Performance Equation:

$$\frac{\text{Time}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} * \frac{\text{Cycles}}{\text{Instruction}} * \frac{\text{Time}}{\text{Cycle}}$$

Deep Pipeline Design

Greater potential throughput but:

- Clock uncertainty and latch delays eat into cycle time budget
  - doubling pipeline depth gives less than twice frequency improvement
- Clock load and power increases
  - more latches running at higher frequencies
- More complicated microarchitecture needed to cover long branch mispredict penalties and cache miss penalties
  - from Little's Law, need more instructions in flight to cover longer latencies ➔ larger reorder buffers
- P-4 has three major clock domains
  - Double pumped ALU (3 GHz), small critical area at highest speed
  - Main CPU pipeline (1.5 GHz in 0.18µm)
  - Trace cache (0.75 GHz), save power

Scaling of Wire Delay

- Over time, transistors are getting relatively faster than long wires
  - wire resistance growing dramatically with shrinking width and height
  - capacitance roughly fixed for constant length wire
  - RC delays of fixed length wire rising

- Chips are getting bigger
  - P-4 >2x size of P-III

- Clock frequency rising faster than transistor speed
  - deeper pipelines, fewer logic gates per cycle
  - more advanced circuit designs (each gate goes faster)

$\Rightarrow$ Takes multiple cycles for signal to cross chip

Visible Wire Delay in P-4 Design

1	TC Next IP
2	
3	TC Fetch
4	
5	Drive
6	Alloc
7	Rename
8	
9	Queue
10	Schedule 1
11	Schedule 2
12	Schedule 3
13	Dispatch 1
14	Dispatch 2
15	Register File 1
16	Register File 2
17	Execute
18	Flags
19	Branch Check
20	Drive

Pipeline stages dedicated to just driving signals across chip!

P-4 Microarchitecture

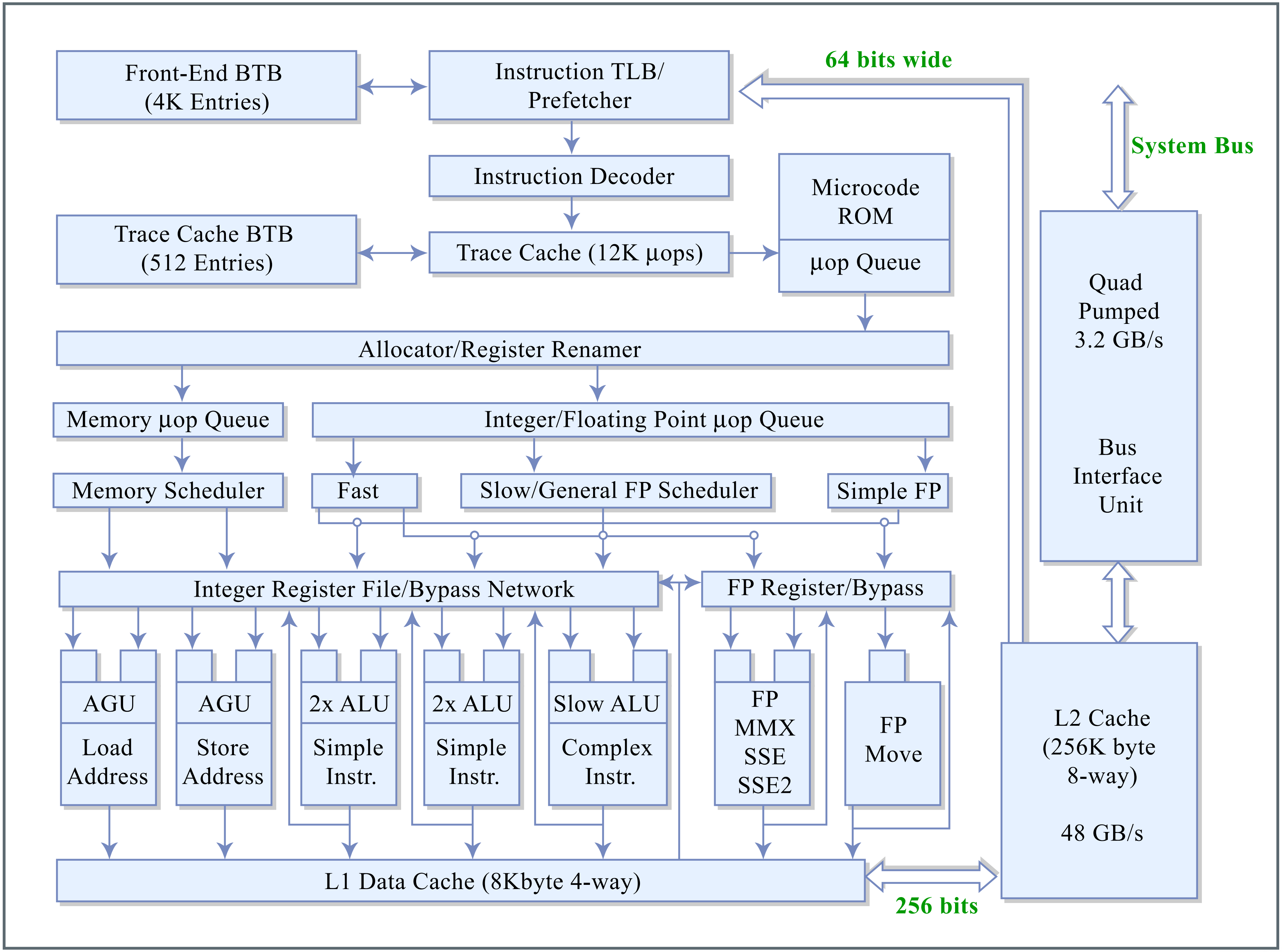
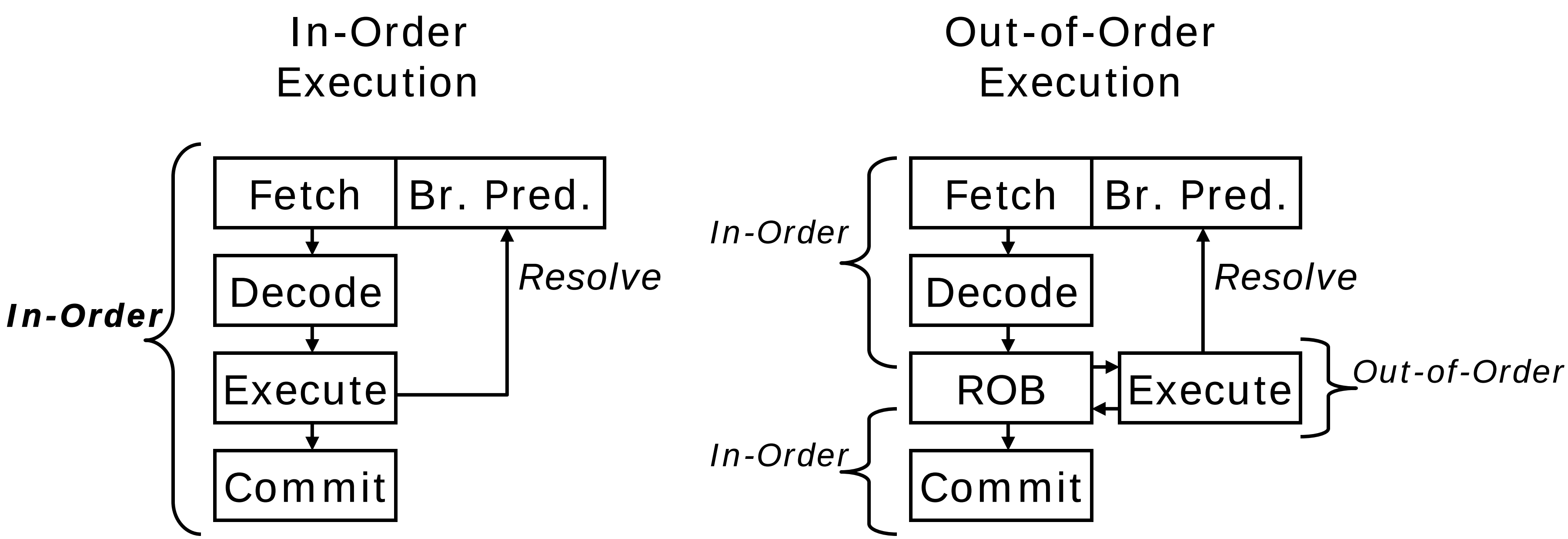


Figure by MIT OCW.

Microarchitecture Comparison



- **Speculative fetch but not speculative execution - branch resolves before later instructions complete**
- **Completed values held in bypass network until commit**

- **Speculative execution, with branches resolved after later instructions complete**
- **Completed values held in rename registers in ROB or unified physical register file until commit**

- Both styles of machine can use same branch predictors in front-end fetch pipeline, and both can execute multiple instructions per cycle
- Common to have 10-30 pipeline stages in either style of design

MIPS R10000 (1995)

Image removed due to copyright restrictions.
To view the image, visit http://www-vlsi.stanford.edu/group/chips_micropro_body.html

- 0.35µm CMOS, 4 metal layers
- Four instructions per cycle
- Out-of-order execution
- Register renaming
- Speculative execution past 4 branches
- On-chip 32KB/32KB split I/D cache, 2-way set-associative
- Off-chip L2 cache
- Non-blocking caches

Compare with simple 5-stage pipeline (R5K series)

- ~1.6x performance SPECint95
- ~5x CPU logic area
- ~10x design effort