CS698Y: Modern Memory Systems
Lecture-2 (Brushing-up Computer Architecture)

Biswabandan Panda

biswap@cse.iitk.ac.in

https://www.cse.iitk.ac.in/users/biswap/CS698Y.html

Assignment-0 (Highlights)

Why you want to enroll for CS698Y?
YES

A **link** to your web-page/linkedin that has your smiling face
None

Type the following: "I promise, I will ……."
YES

Any other comments/suggestions that you have for me or for CS698Y ?
Please give good grades ☺

What you want to do after your B.Tech/M.Tech/M.S./Ph.D. ?
Teaching (that too poor students of India)

Logistics

Contact: KD 203, biswap@cse.iitk.ac.in
Office Hours: Tues/Fri: 12 noon, by appointment

No participation in Piazza yet (oh wait, Partha had it in the last minute). There is chance of losing 10 points ☹

We have a T.A. now: Saurabh (Pursuing Ph.D. in Android security)

What about a 5min break at 11.15Hrs or so?

Stop me if you feel like not listening to me ☺

Shall I Freeze it?

Option-I:

30 = (3 × 10) = 3 programming assignments

40 = (2 × 20) = Quiz 1.0 and Quiz 2.0 (Optional Quiz 1.1 and Quiz 2.1) = max (Quiz 1.x) + max (Quiz 2.x)

10 = (2 × 10) = 2 paper reviews

10 = Classroom and Piazza participation

Option-II:

30 = (3 × 10) = 3 programming assignments

20 = (1 × 20) = max (Quiz 1.0, Quiz 1.1)

30 = (1 × 30) = 1 research project (weekly meetings)

10 = (2 × 5) = 2 paper reviews

10 = Classroom and Piazza participation

Let's Get Started

Wait!!!!

Latency, Bandwidth, Energy, and Power??

Latency vs Bandwidth

Latency vs Bandwidth, How they affect each other?

Latency helps bandwidth but not vice versa.

DRAM latency ↓ **More # Accesses ~DRAM Bandwidth** ↑

Bandwidth usually hurts latency

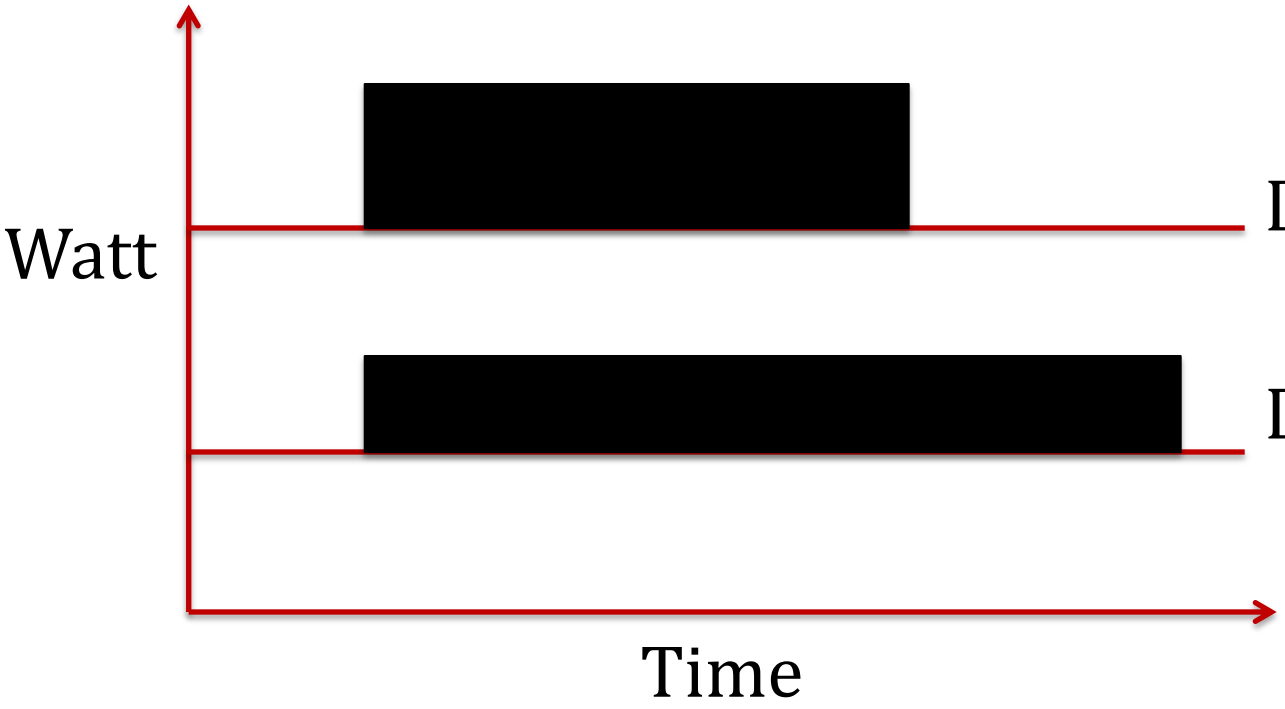
Queues - Bandwidth ↑ **Increases latency** ↓

Bandwidth problems can be cured with money.
Latency problems are harder because the speed of light is fixed – you can't bribe God

Energy vs Power

Energy: Measure of using power for some time

Power: Instantaneous rate of energy transfer



Power: Height of the curve

Energy: Area under the curve

Let's Bring Latency ….

Source: Powerbar

Power efficiency = Performance/watt

Energy efficiency = Performance/Joule

BASICS

Execution time, Performance, CPI, IPC

Latency & Bandwidth, Amdahl's Law

Instruction Pipelining, Branch Prediction

LOAD/STORE(s), ROB, LQ/SQ

Superscalar, SMT Processor

Execution Time

$$\text{Execution Time} = \frac{\text{Time}}{\text{Program}}$$

Executed Instructions not static ones: Driven by: Algorithm, ISA, and Compilers

Instruction ??

$$= \frac{\text{Instructions}}{\text{Program}} \times$$

Driven by: ISA and Processor Organization

Cycle ??

$$= \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times$$

Driven by Technology

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

(Iron law)

Cycles Per Instruction (CPI)

Depends on

Depth of Processor pipeline, Instruction level Parallelism

Accuracy of branch predictor

Latency of Caches, TLBs, and DRAM

Many others ……

Performance Measurement

Performance of a machine – in terms of execution time, throughput ?

Latency is additive, throughput is not

6 hours at 30 km/hour + 2 hours at 90 km/hr
Total latency: 8 hrs
Total throughput: You find out ☺

Why you need to measure what you measure?

Benchmark Suites

Collection of *relevant* programs (binaries)

- SPEC CPU
- SPECWeb
- PARSEC and SPLASH
- Bigdatabench
- Mobile Workloads

Measuring Performance

Pick a *relevant* benchmark suite

Measure IPC of each program

Summarize the performance using:

- Arithmetic Mean (AM)
- Geometric Mean (GM)
- Harmonic Mean (HM)

Which one to choose?

An Example

	IMTEL	ABM	AND
App. one	10	20	30
App. two	20	30	40
App. three	30	40	10

Which machine is better over IMTEL and why?

An Example – Normalized Performance (Speedup)

	ABM	AND
App. one	2	3
App. two	1.5	2
App. three	1.3	0.3
A.M.	1.60	**1.76**
G.M.	**1.57**	1.21
H.M.	**1.54**	**0.72**

Source: Pinterest

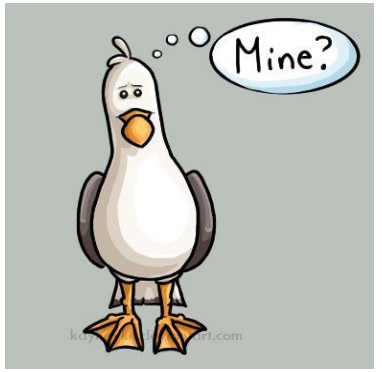
AM on Ratios

	X	Y
App. 1	1	100
App. 2	1000	10

Normalized to X	X	Y
App. 1	1	100
App. 2	1	0.01
AM	1	50.005

Normalized to Y	X	Y
App. 1	0.01	1
App. 2	100	1
AM	50.005	1

Y is 50 times faster than X



X is 50 times faster than Y

When to use What?

Edgar H. Sibley
Panel Editor

Using the arithmetic mean to summarize normalized benchmark results leads to mistaken conclusions that can be avoided by using the preferred method: the geometric mean.

Do not use A.M. on normalized numbers

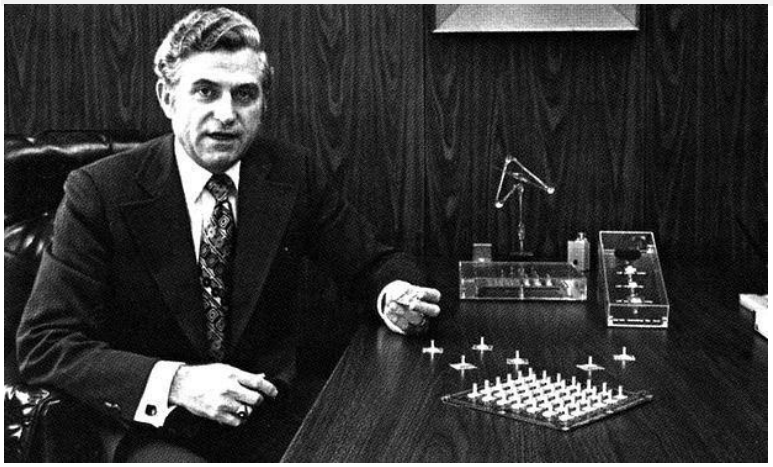
HOW NOT TO LIE WITH STATISTICS: THE CORRECT WAY TO SUMMARIZE BENCHMARK RESULTS

Use G.M. for normalized numbers

PHILIP J. FLEMING and JOHN J. WALLACE

Communications of the ACM, March 1986, pp. 218-221.

Amdahl's Law



Source: The Guardian

$$ExTime_{new} = ExTime_{old} \times \left[\left(1 - Fraction_{enhanced}\right) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}} \right]$$

$$Speedup_{overall} = \frac{ExTime_{old}}{ExTime_{new}} = \frac{1}{\left(1 - Fraction_{enhanced}\right) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}}}$$

Amdahl's Law

Which one will provide better overall speedup?

A. Small speedup on the large fraction of execution time.
B. Large speedup on the small fraction of execution time.
C. Does not matter.

Depends on the difference between small and large. Mostly it is A.

Amdahl's law for parallel processing

Cycles Per Instruction

Depends on

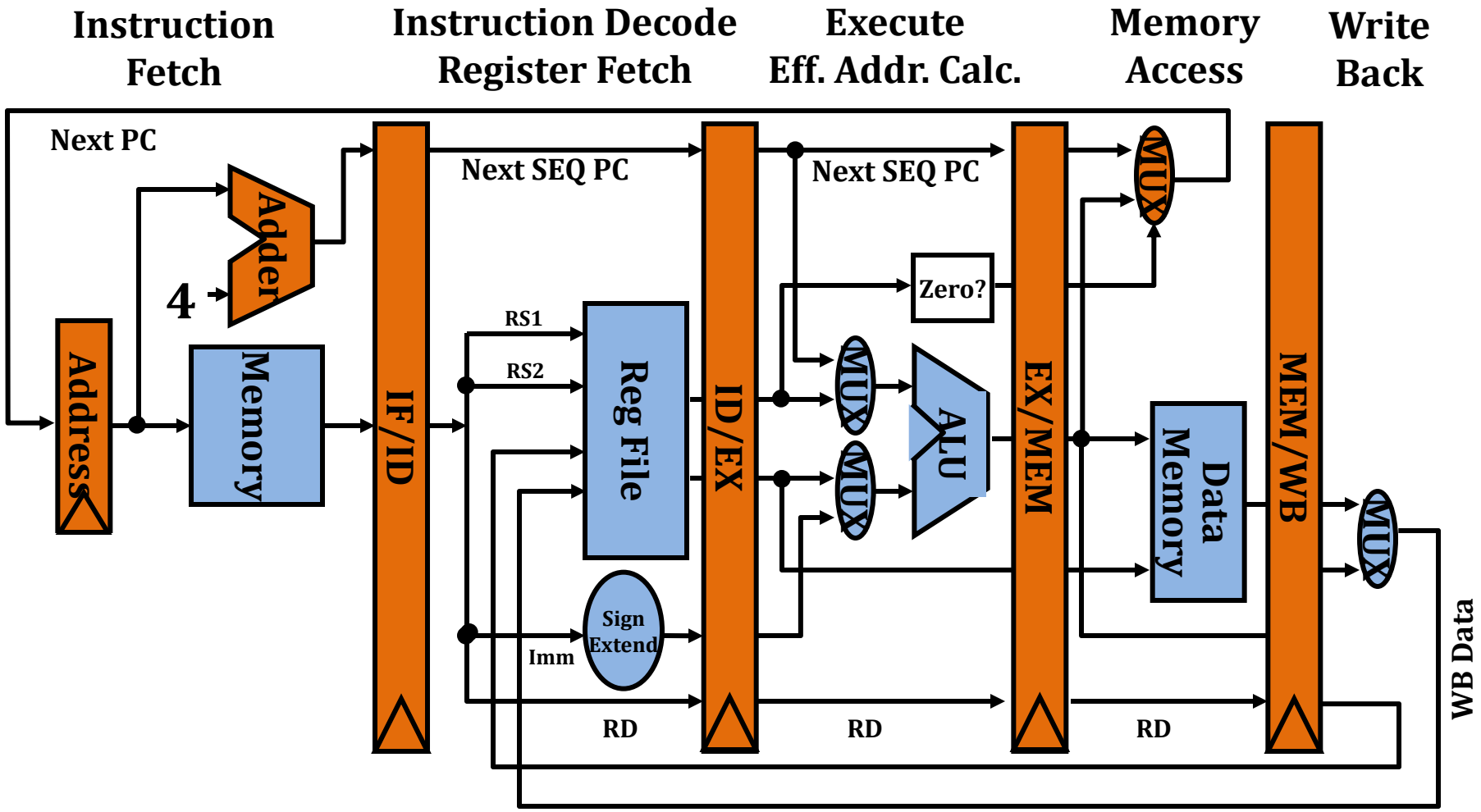
Depth of Processor pipeline, Instruction level Parallelism

Accuracy of branch predictor

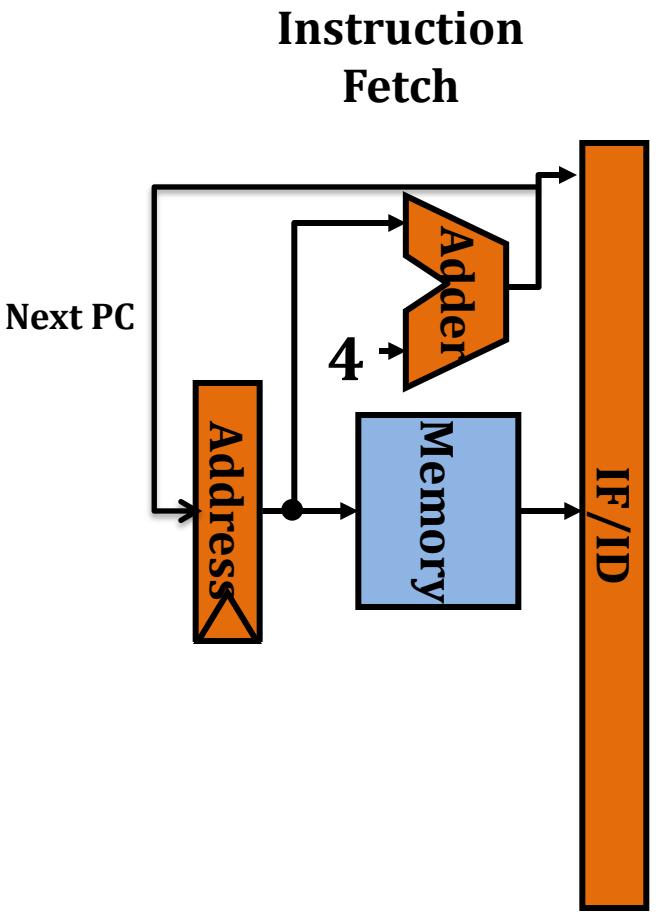
Latency of Caches, TLBs, and DRAM

Many others ……

Instruction Pipelining



Fetch Stage

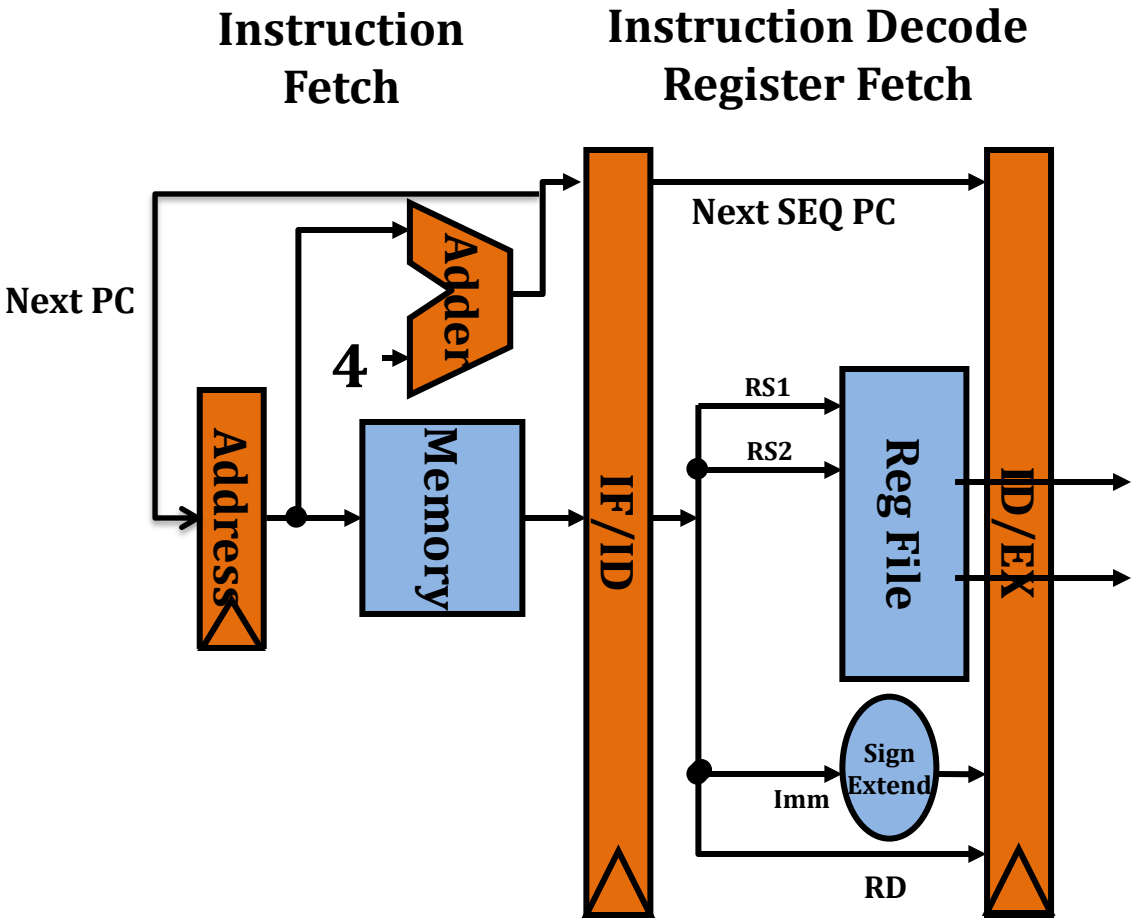


IR <= mem[PC]

PC <= PC + 4

What is the first instruction that is fetched when a system is power-up?

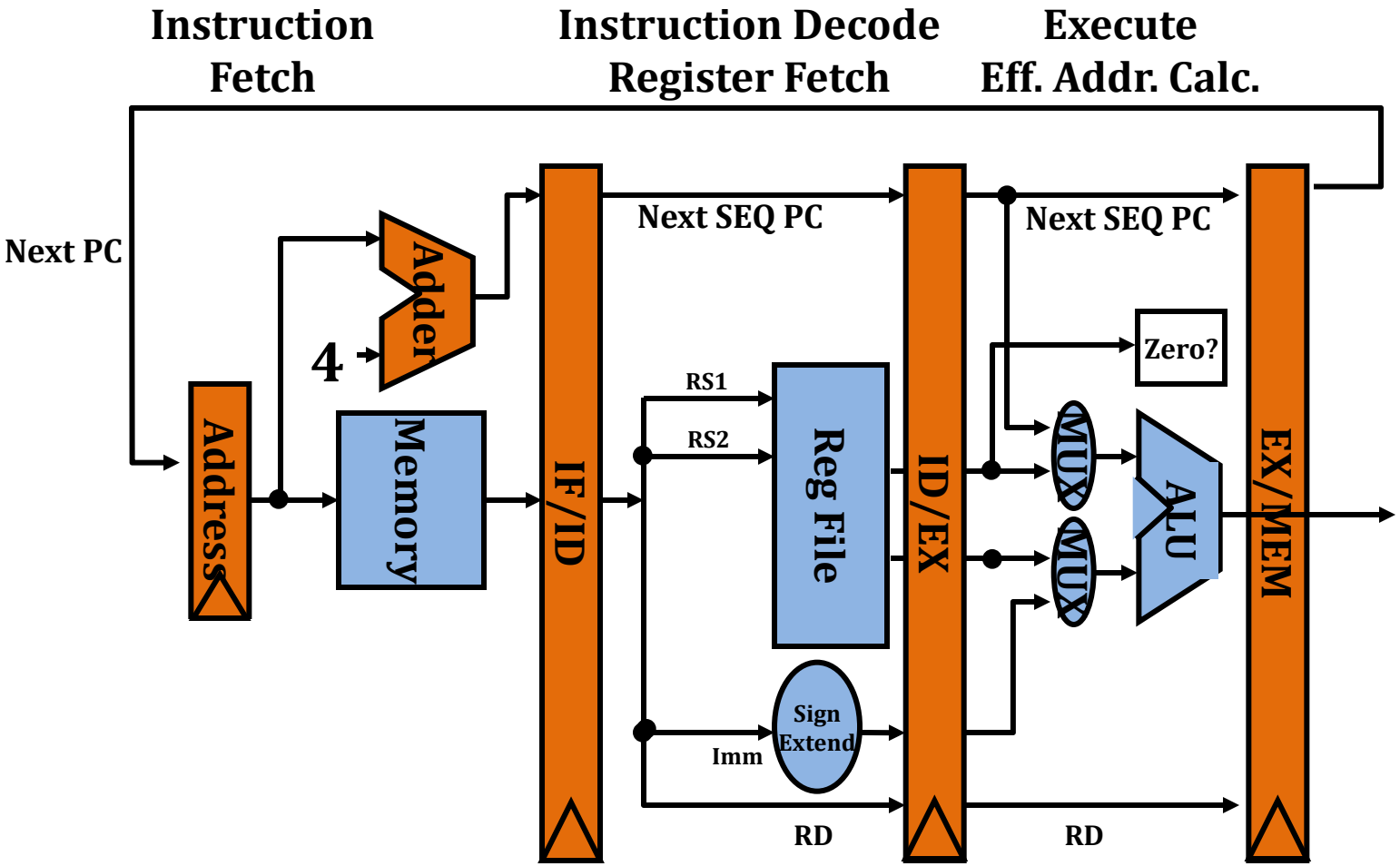
Decode Stage



$A <= Reg[IR_{rs}]$

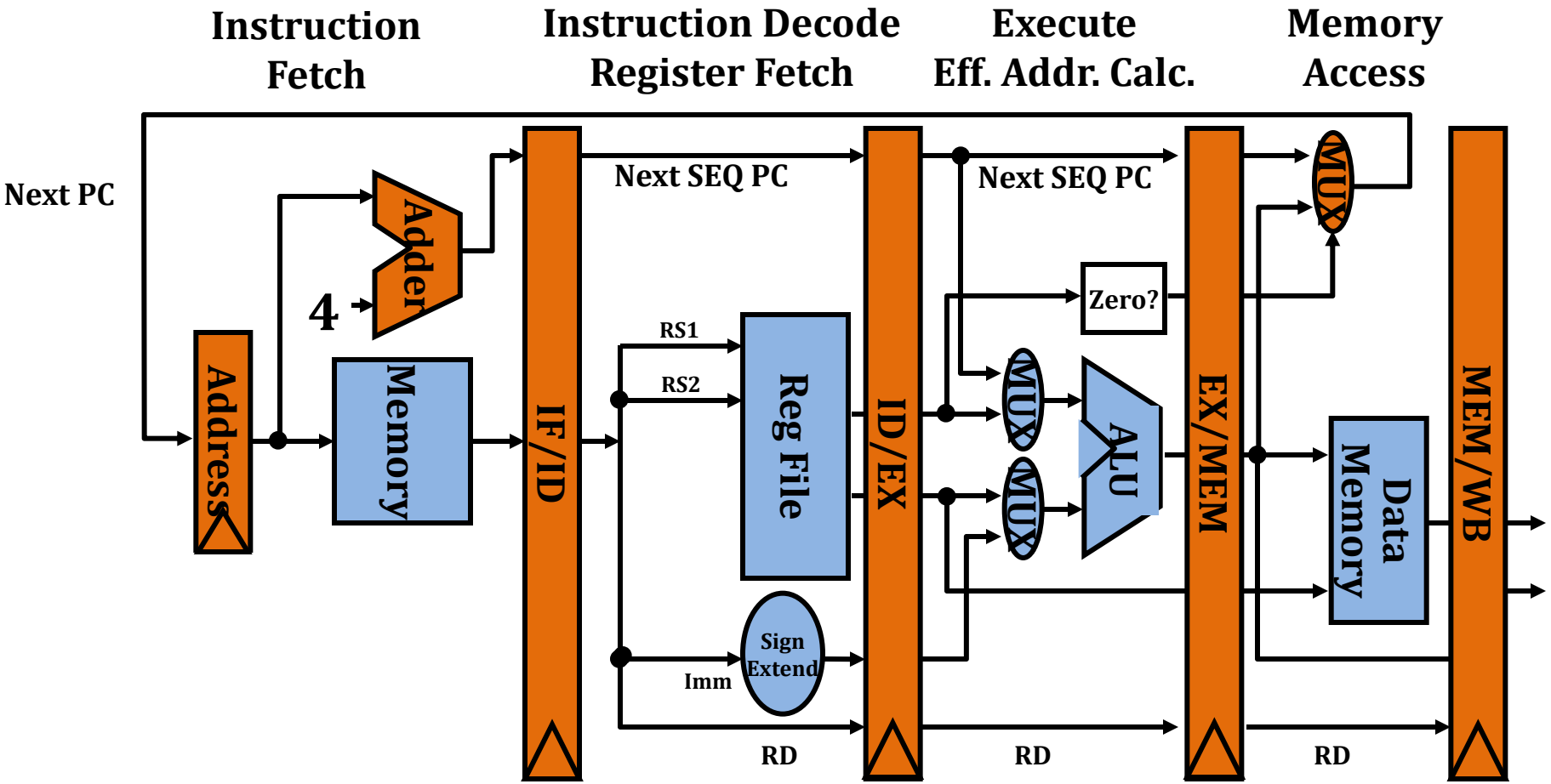
$B <= Reg[IR_{rt}]$

Execute Stage



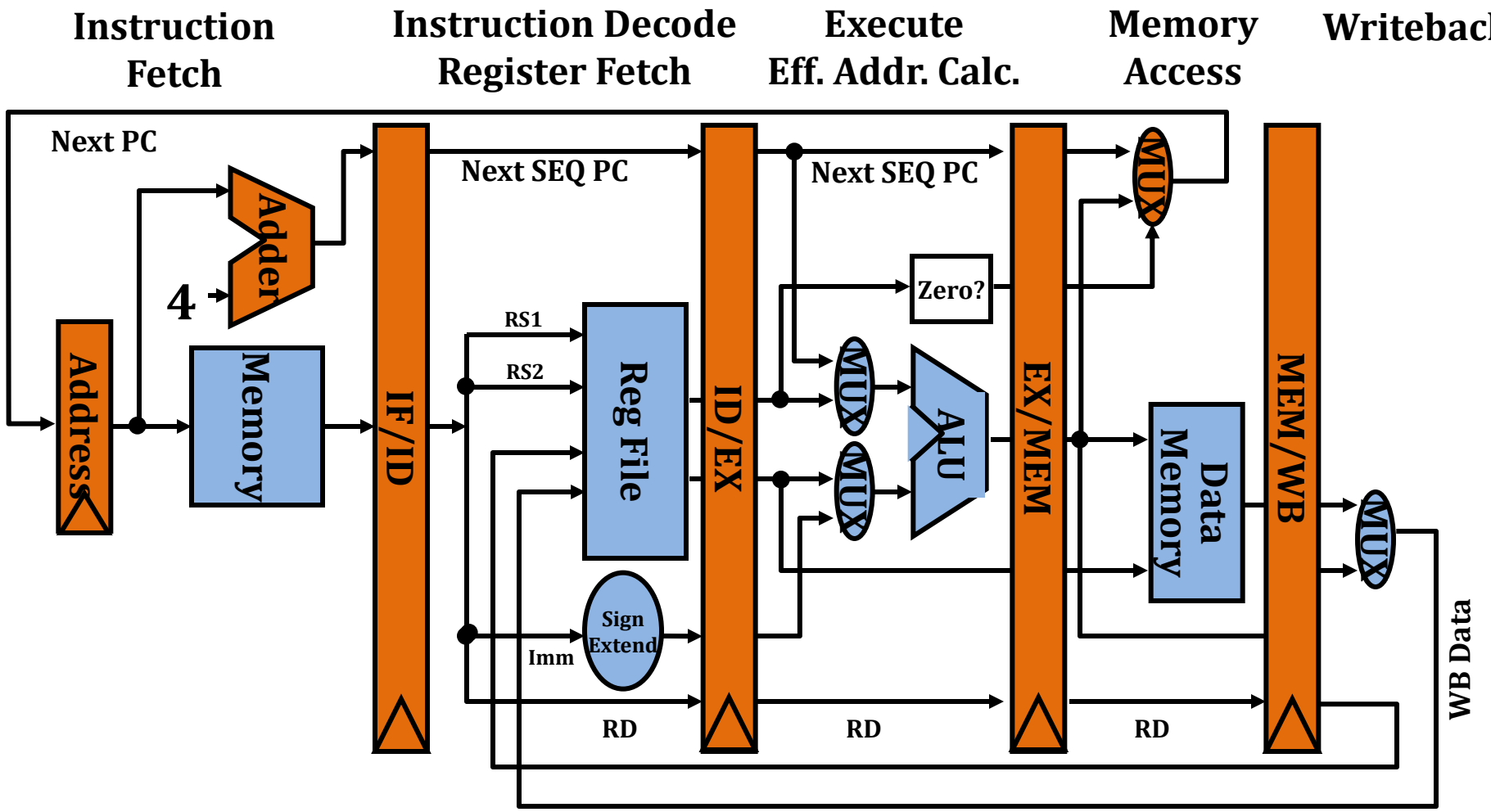
$$rslt <= A\ op_{IRop}\ B$$

Memory Stage



WB <= rslt

Writeback Stage



How many stages in commercial processors?

$Reg[IR_{rd}] <= WB$

Speedup with Pipelining

	t0	t1	t2	t3	t4	t5	t6	t7	t8
I1	IF_1	ID_1	EX_1	MA_1	WB_1				
I2		IF_2	ID_2	EX_2	MA_2	WB_2			
I3			IF_3	ID_3	EX_3	MA_3	WB_3		
I4				IF_4	ID_4	EX_4	MA_4	WB_4	
I5					IF_5	ID_5	EX_5	MA_5	WB_5

Speedup achieved with pipelining? *25/9*

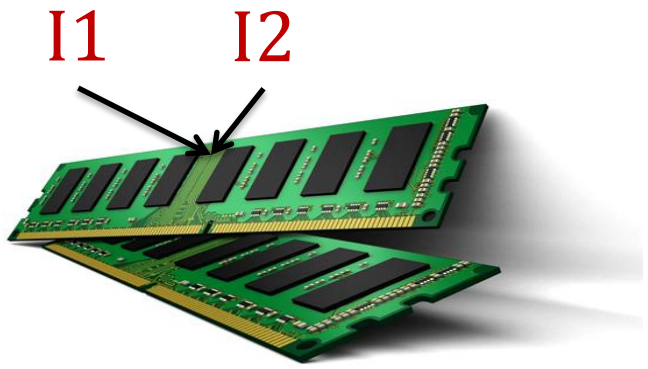
What about THE ideal 5-stage pipeline for a billion of instructions? *~5*

Real World (! Ideal one)

Pipeline hazards

Next instruction can not proceed through pipeline as expected in the ideal case

Structural



I1 and I2 conflicting for a resource: Memory, Caches

Data

I1 → X=X+1

I2 → Y=X+2

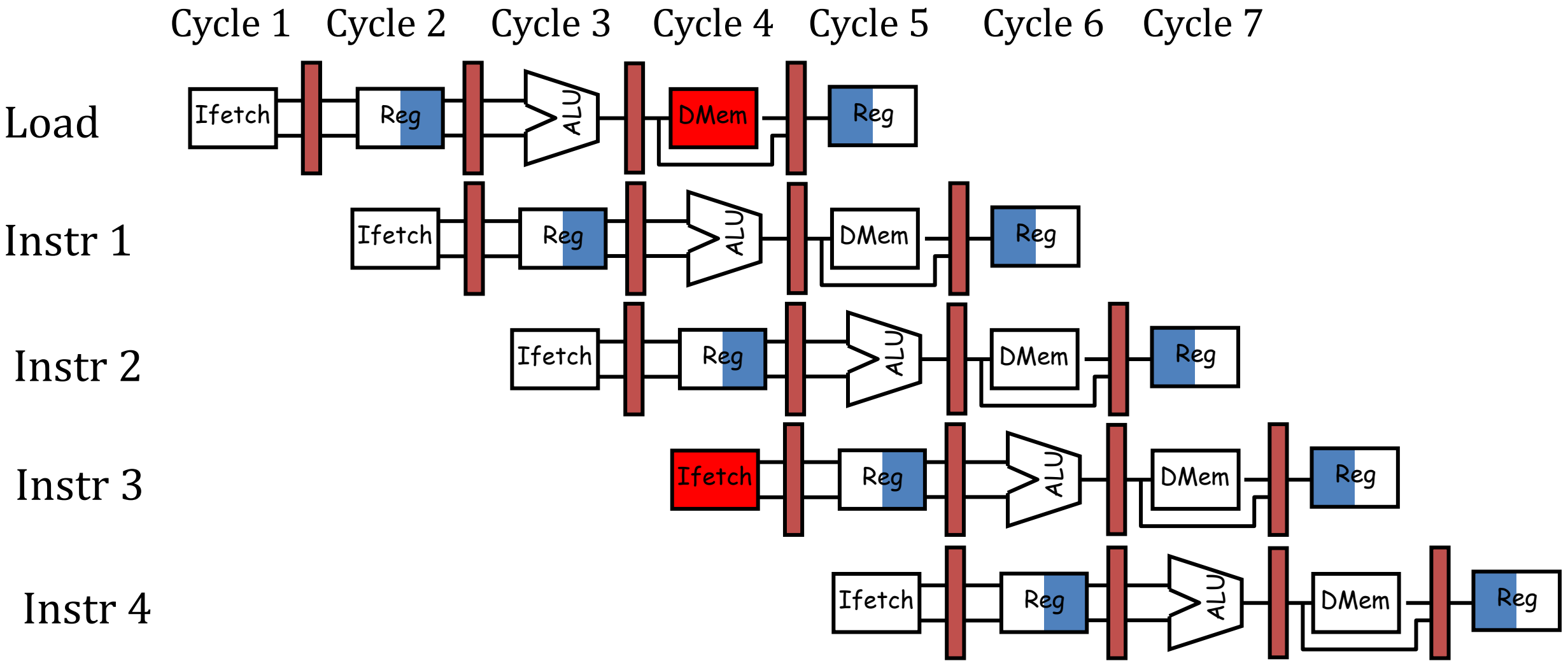
I2 dependent on data value generated by I1

Control

I1 → If(X==2) I2 else I3

I2 dependent on I1, where I1 is a branch instruction

Structural Hazard



Adapted from Computer Architecture: A Quantitative Approach, Copyright 2005 UCB

Data Hazard

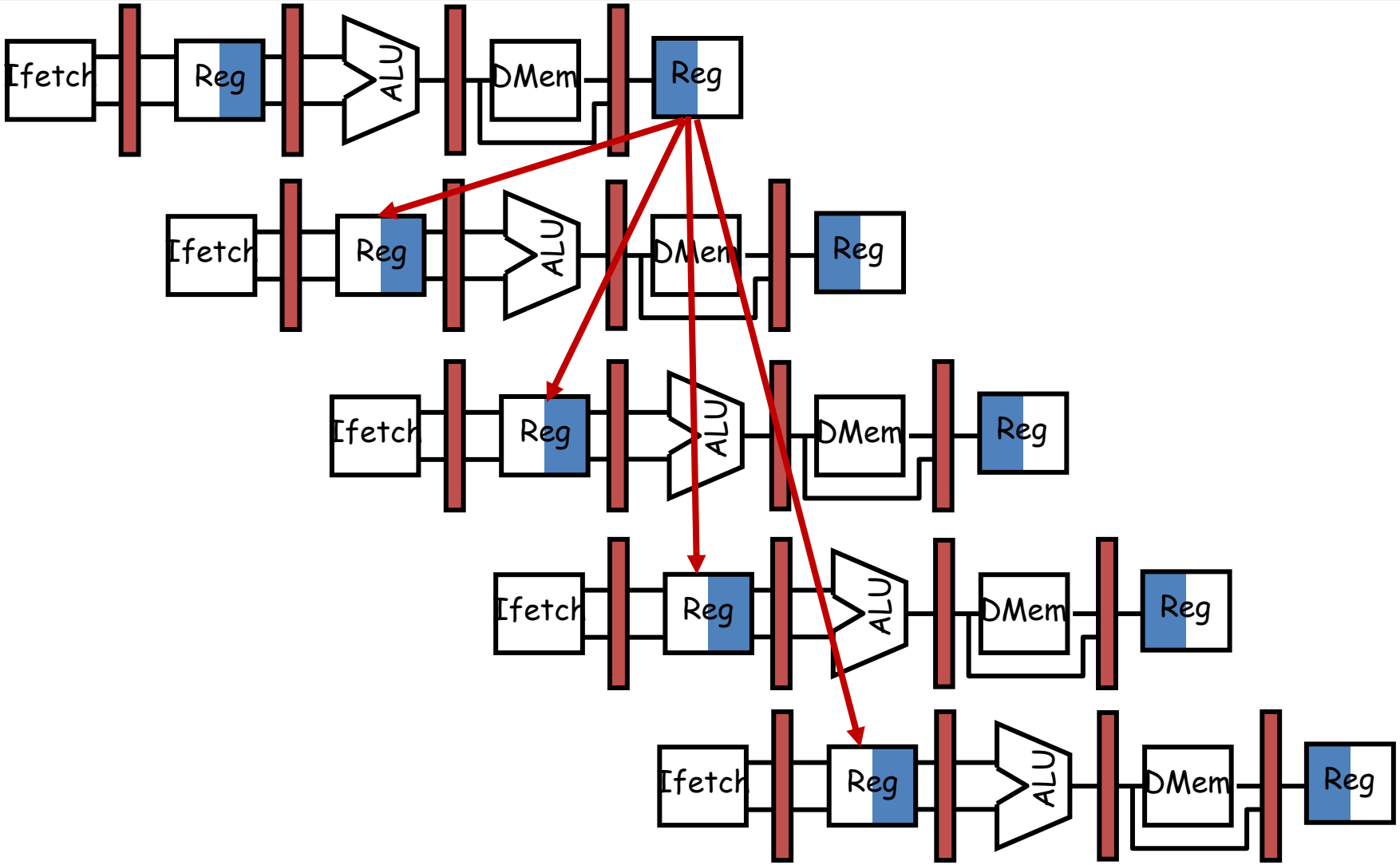
add r1,r2,r3

sub r4,r1,r3

and r6,r1,r7

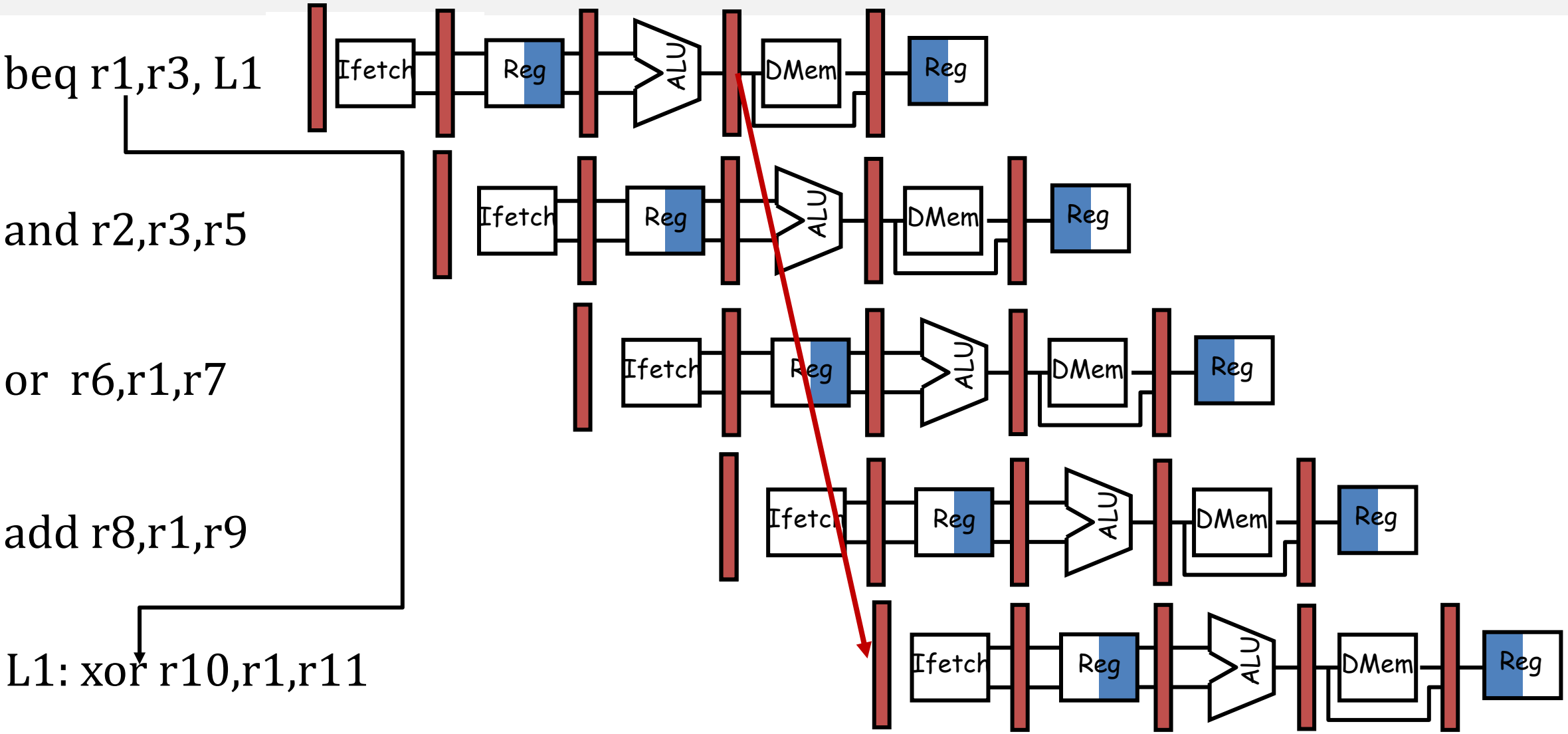
or r8,r1,r9

xor r10,r1,r11



Adapted from Computer Architecture: A Quantitative Approach, Copyright 2005 UCB

Control Hazard



Adapted from Computer Architecture: A Quantitative Approach, Copyright 2005 UCB

Branch Predictors

A hardware structure that predicts T/NT of a branch instruction

Why? Branch miss penalty ~ 20 cycles

Used in stage ?? F/D/E ??

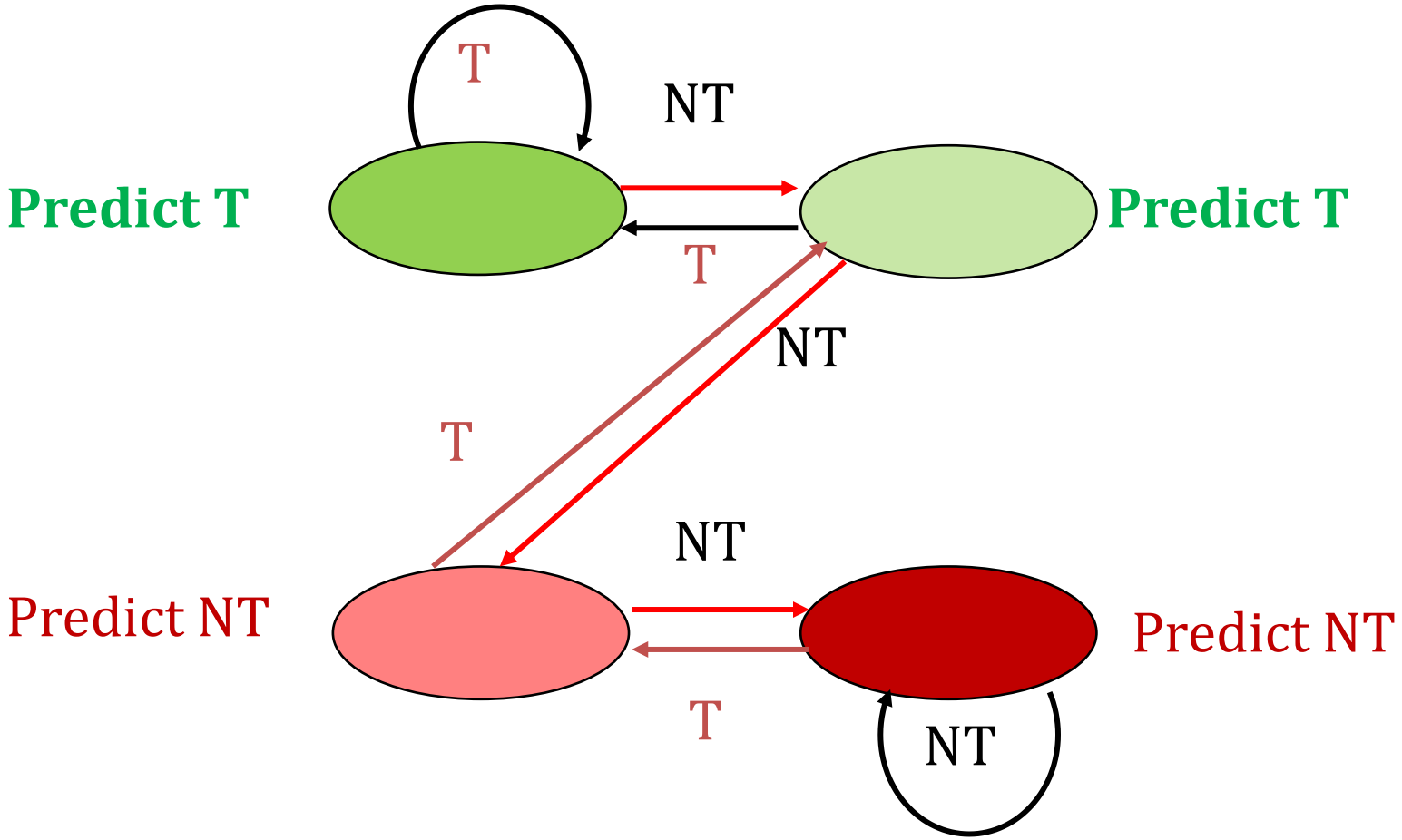
Static Prediction – Always T/NT

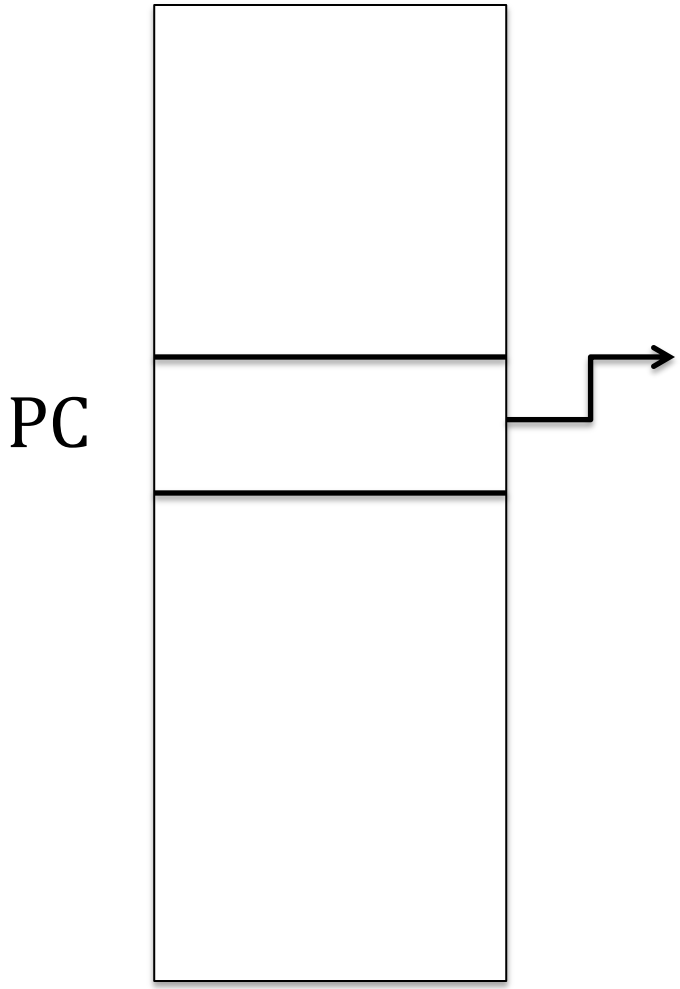
Dynamic Prediction – Exploits the dynamic nature of code.

Dynamic Branch Predictors

TAGE, Perceptron, Tournament, Agree, Alloyed, Hybrid, Neural, Correlating, Skewed, O-GHEL, Franken,

2-bit Predictor

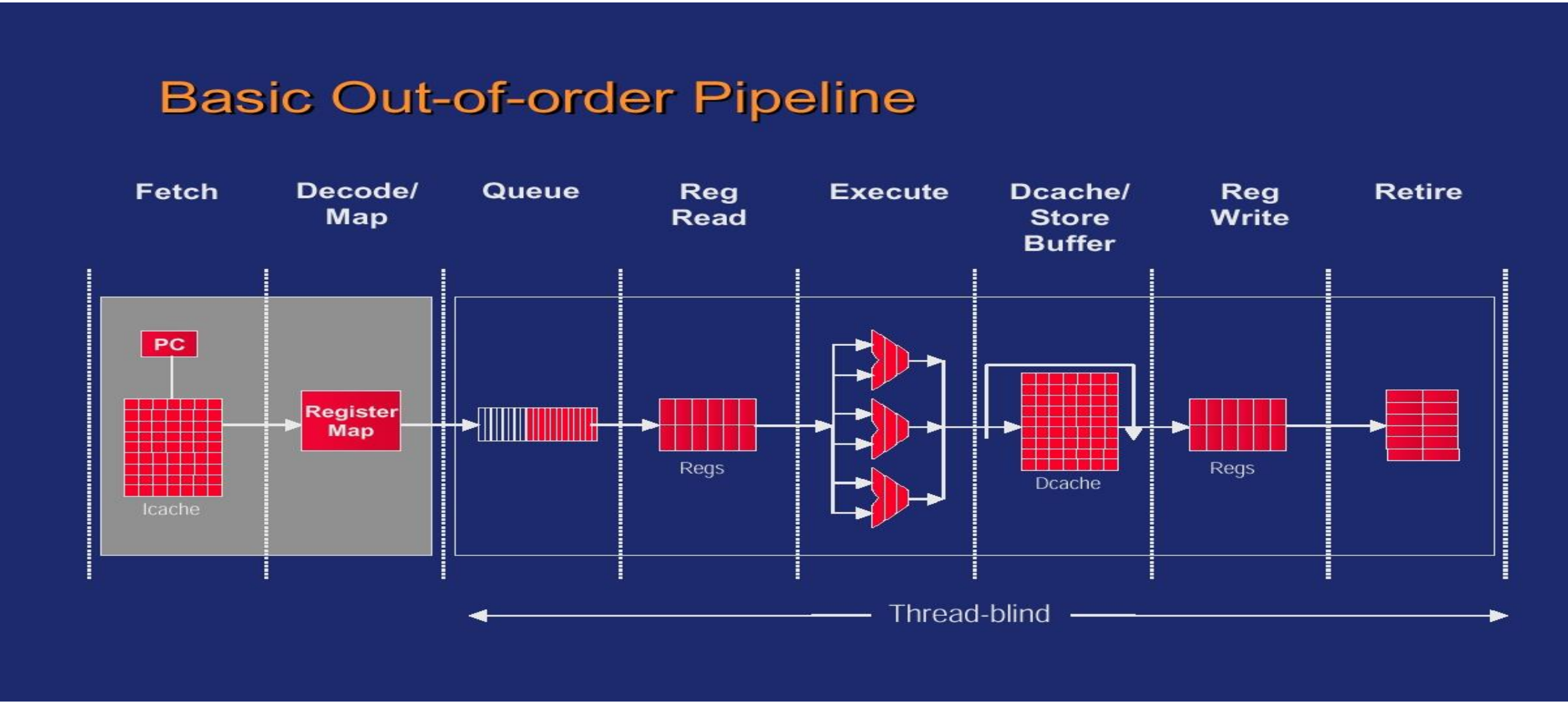




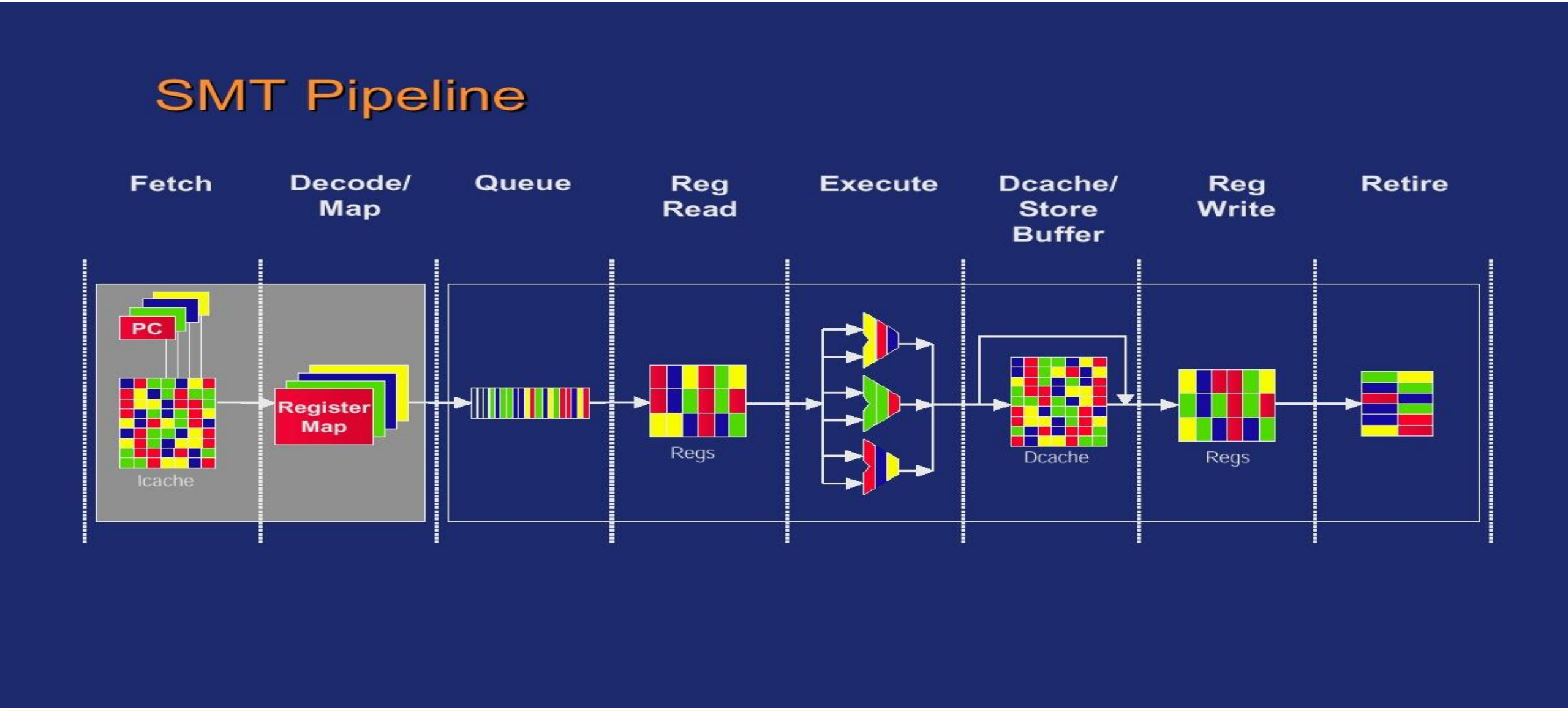
Buzzwords from Processor Design

ROB, LSQ, Register renaming, Superscalar, Tomasulo Algorithm, IQ, Register data flow, Reservation stations, Load bypassing (forwarding)

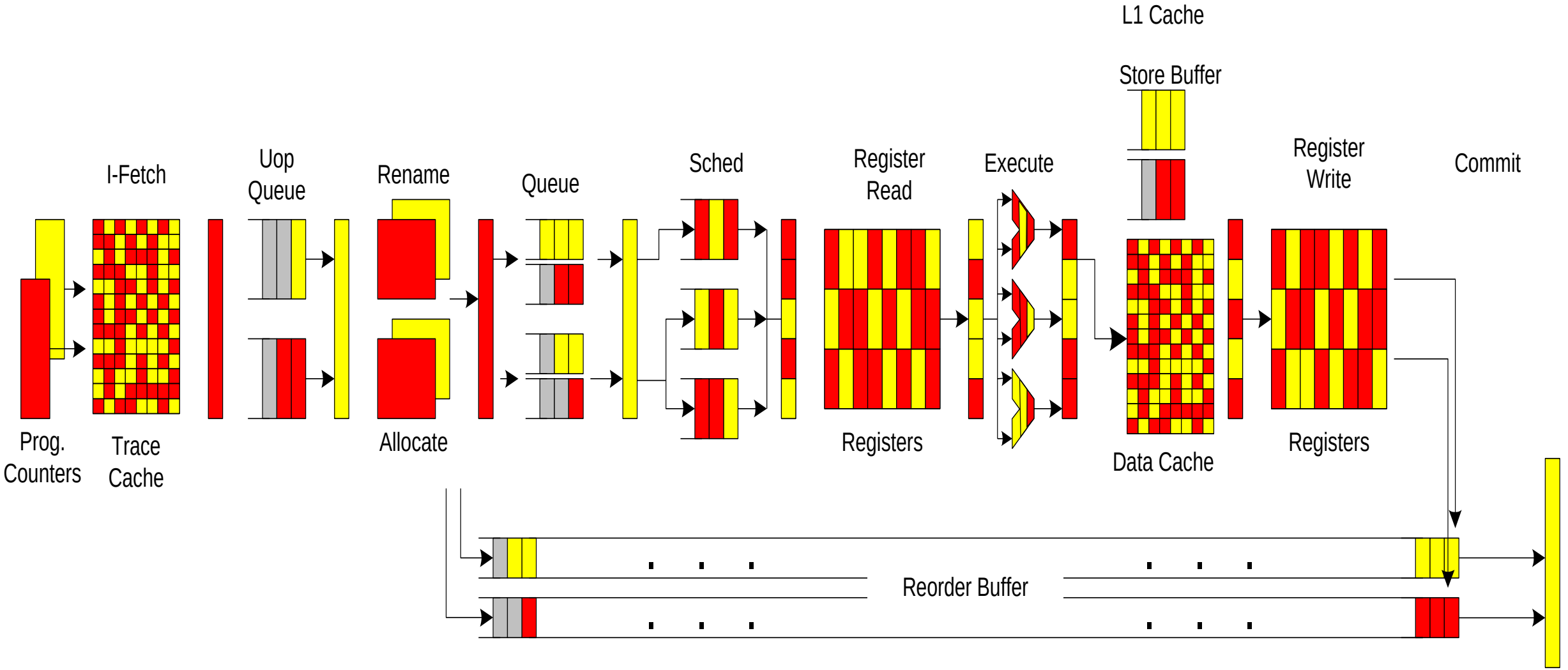
O3 - 101


Basic Out-of-order Pipeline
Fetch
Decode/ Map
Queue
Reg Read
Execute
Dcache/ Store Buffer
Reg Write
Retire
PC
Icache
Register Map
Regs
Dcache
Regs
Thread-blind

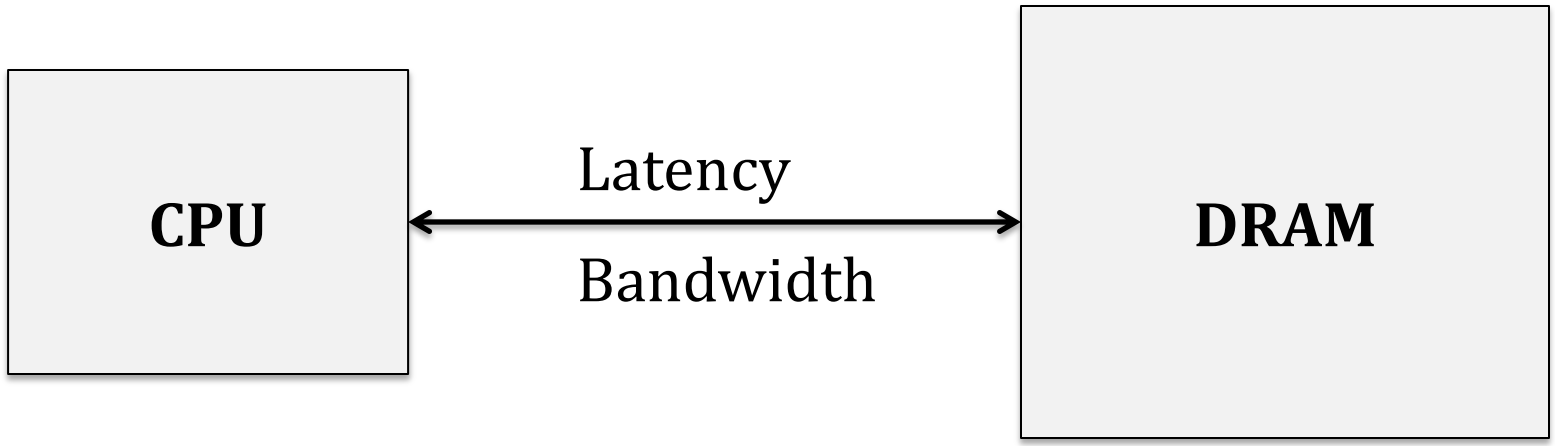
SMT-101


SMT Pipeline
Fetch
Decode/ Map
Queue
Reg Read
Execute
Dcache/ Store Buffer
Reg Write
Retire
PC
Icache
Register Map
Regs
Dcache
Regs

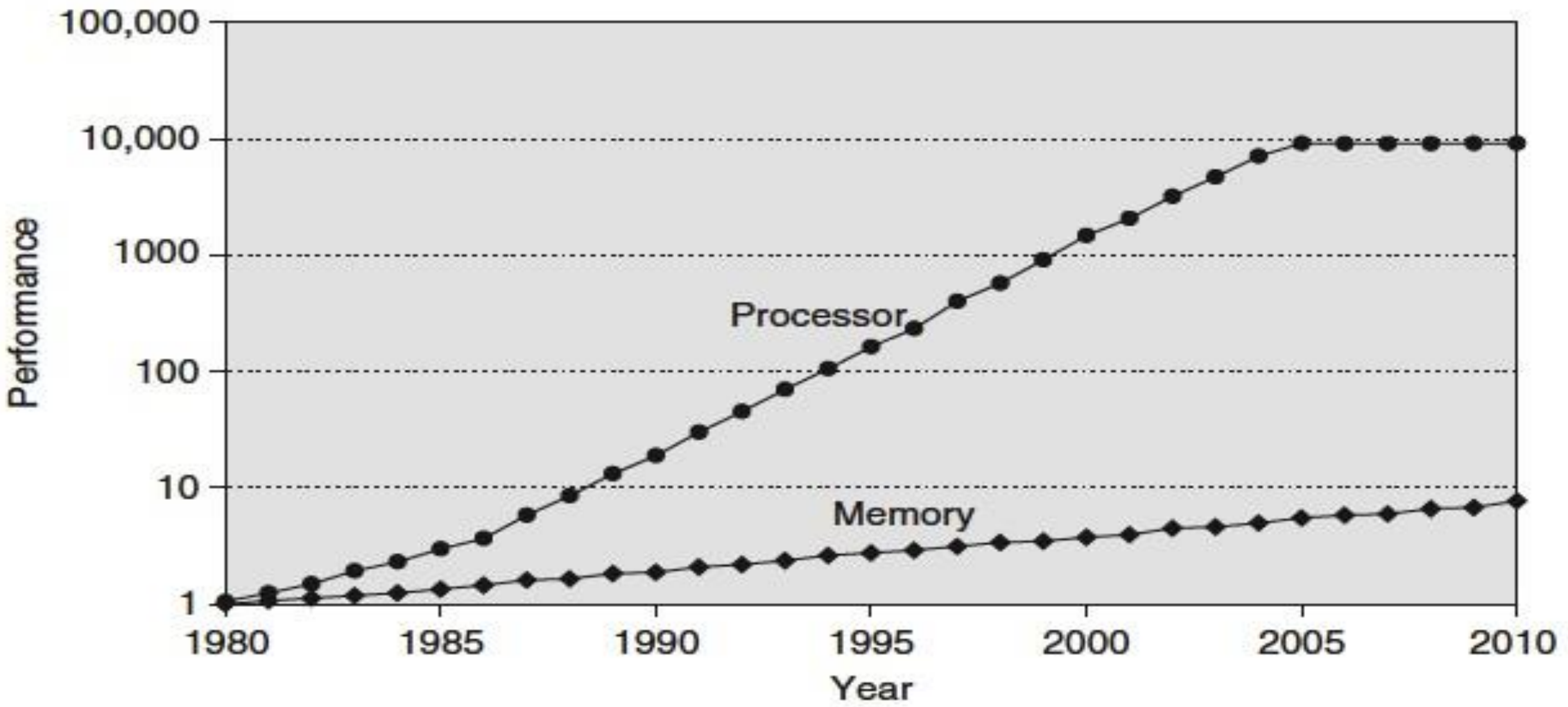
Intel's Hyperthreading



CPU-Memory Gap



CPU-Memory Gap



CPU-Memory Gap

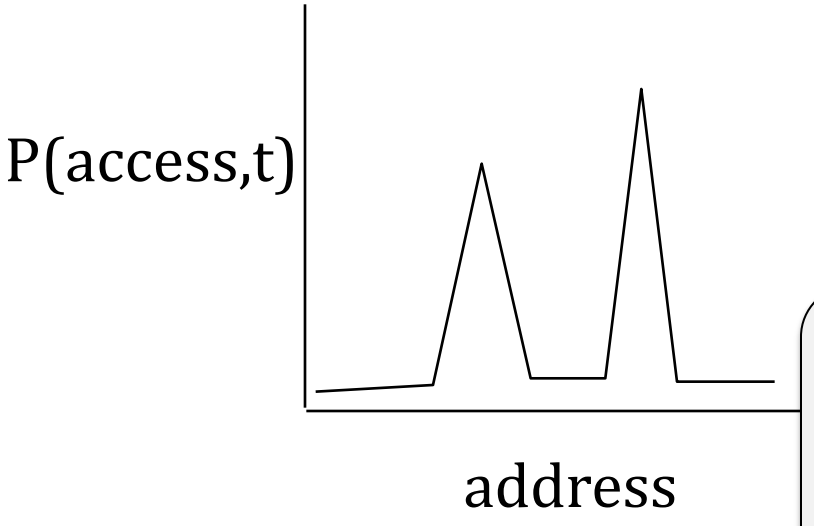
A 4-issue processor running on 4GHz
DRAM latency: 100 ns
#instructions executed during one DRAM access ??

1600

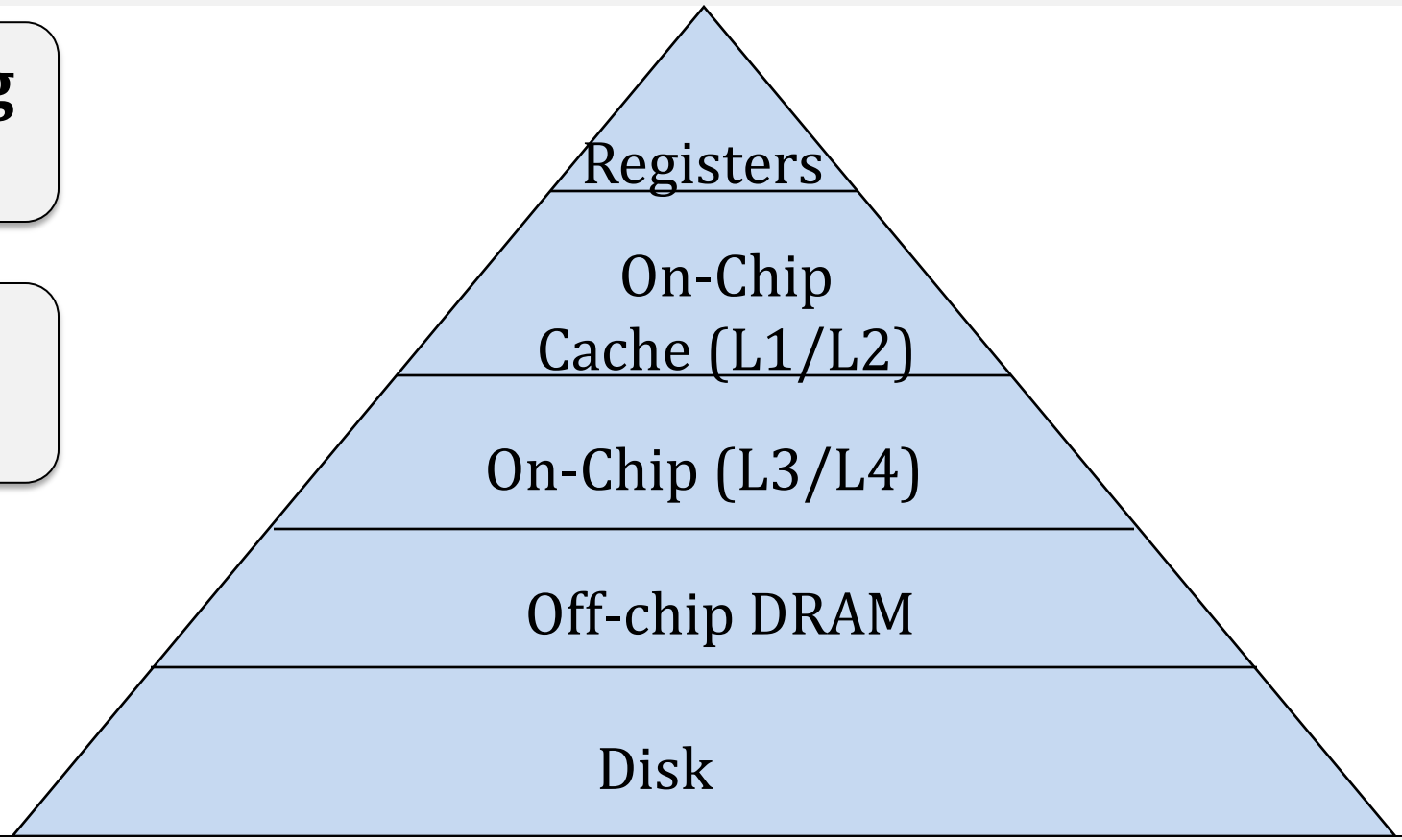
Memory Hierarchy

Reduces the gap by using small and fast caches

Exploits temporal and spatial localities

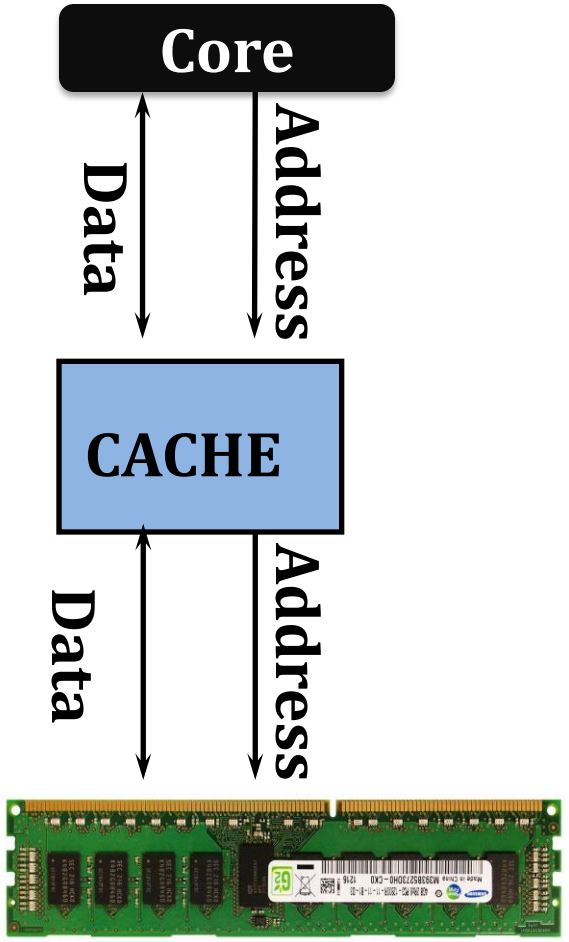


Adapted from Copyright 2005 UCB



Program access a relatively small portion of the address space at any instant of time.
Example: 90% of time in 10% of the code

Cache Events



Hit (found in cache)/Miss?

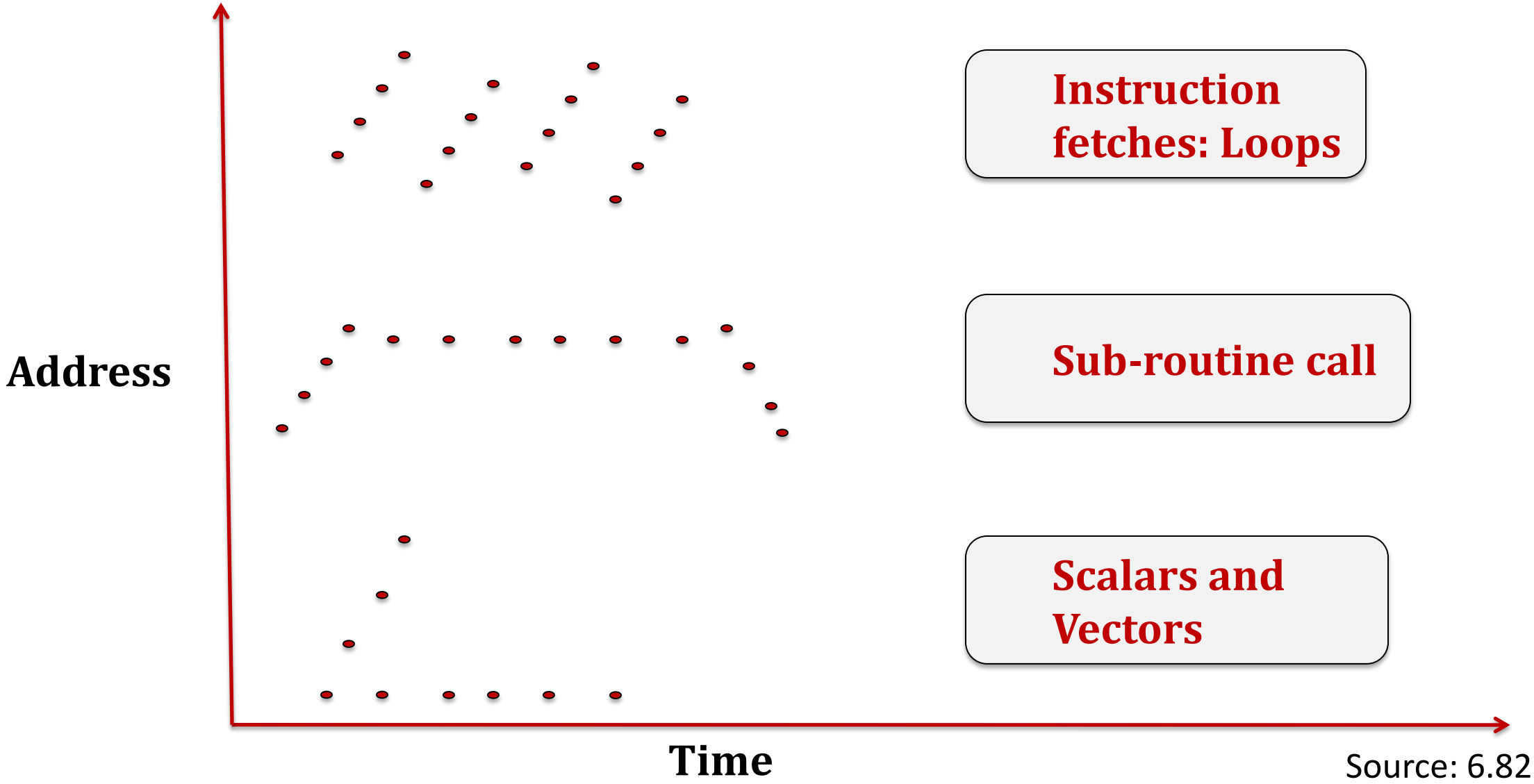
Compulsory Miss

Capacity Miss

Conflict Miss

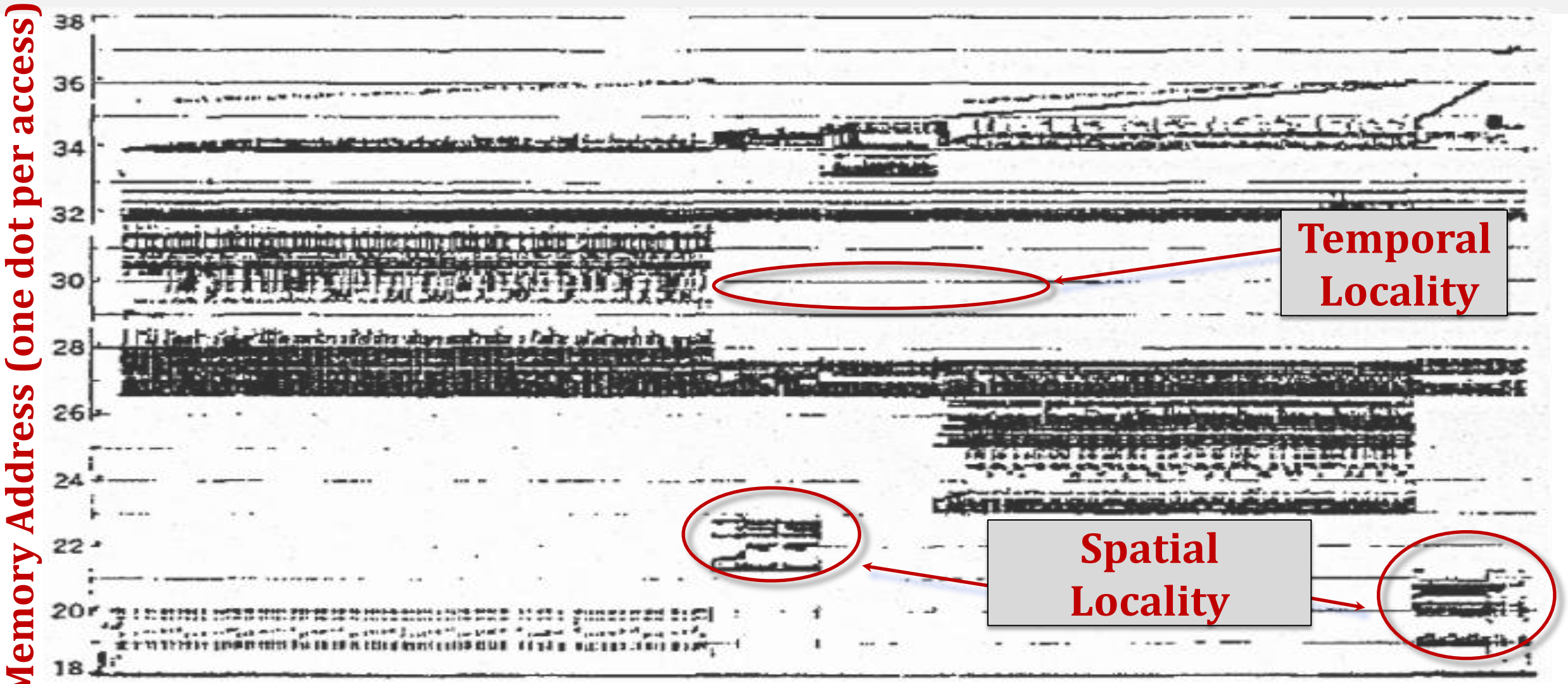
Replacement

Access Patterns



Source: 6.823, MIT

Locality



Donald J. Hatfield, Jeanette Gerald: Program Restructuring for Virtual Memory. IBM Systems Journal 10(3): 168-192 (1971)

Source: 6.823, MIT

Placement Policy

Block #

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

Set #

Cache

0 1 2 3

0 1 2 3 4 5 6 7

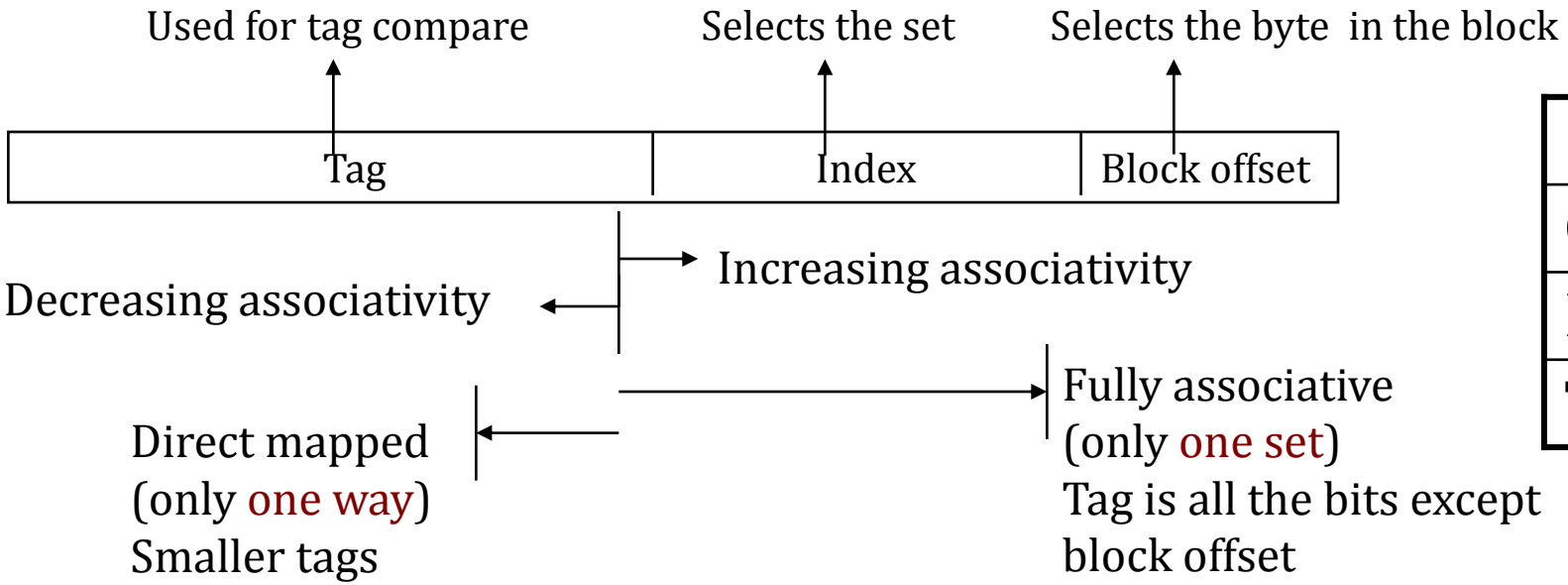
Fully Associative

(2-way) Set Associative

Direct Mapped

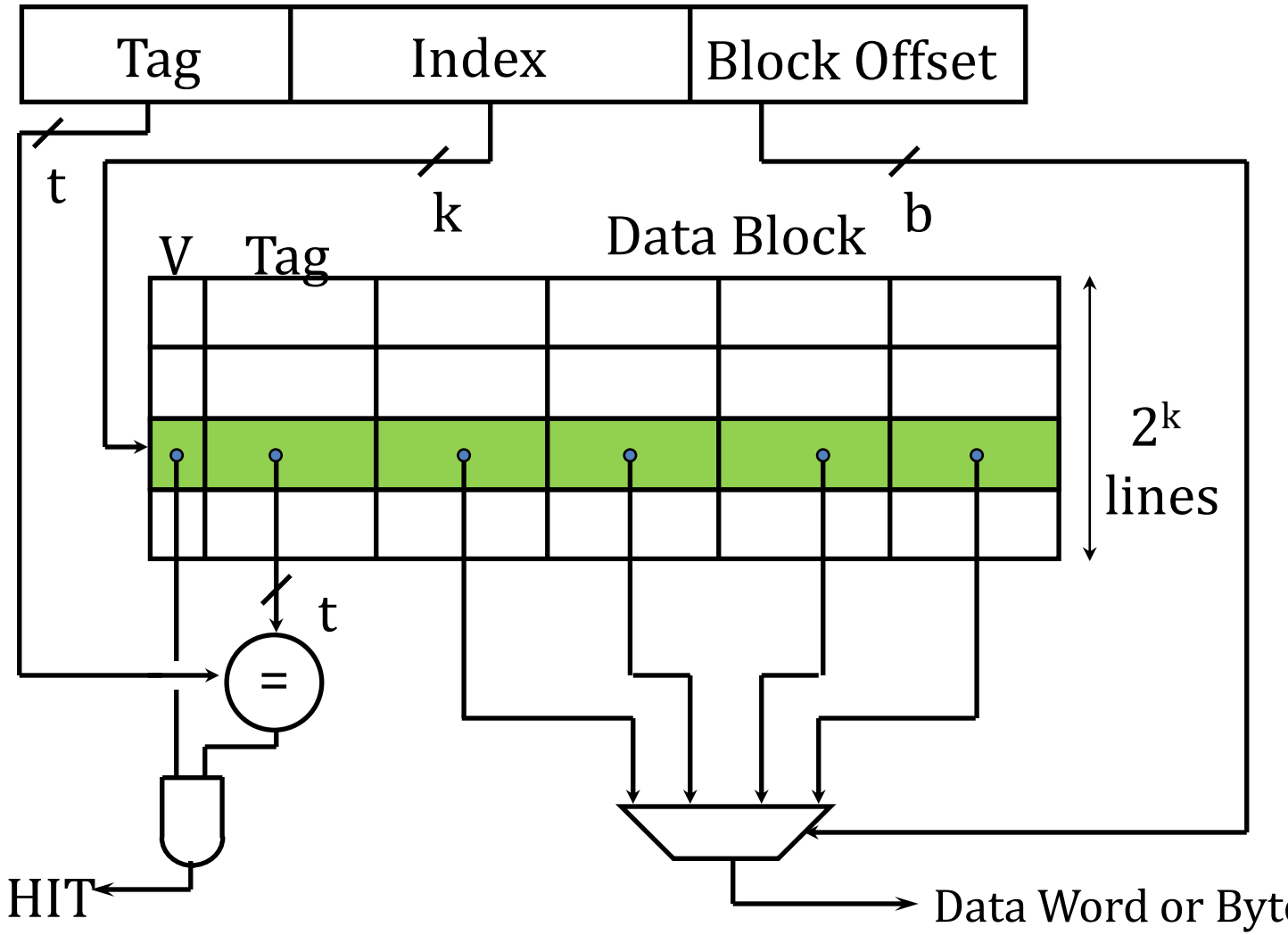
Mapping

n-bit Address



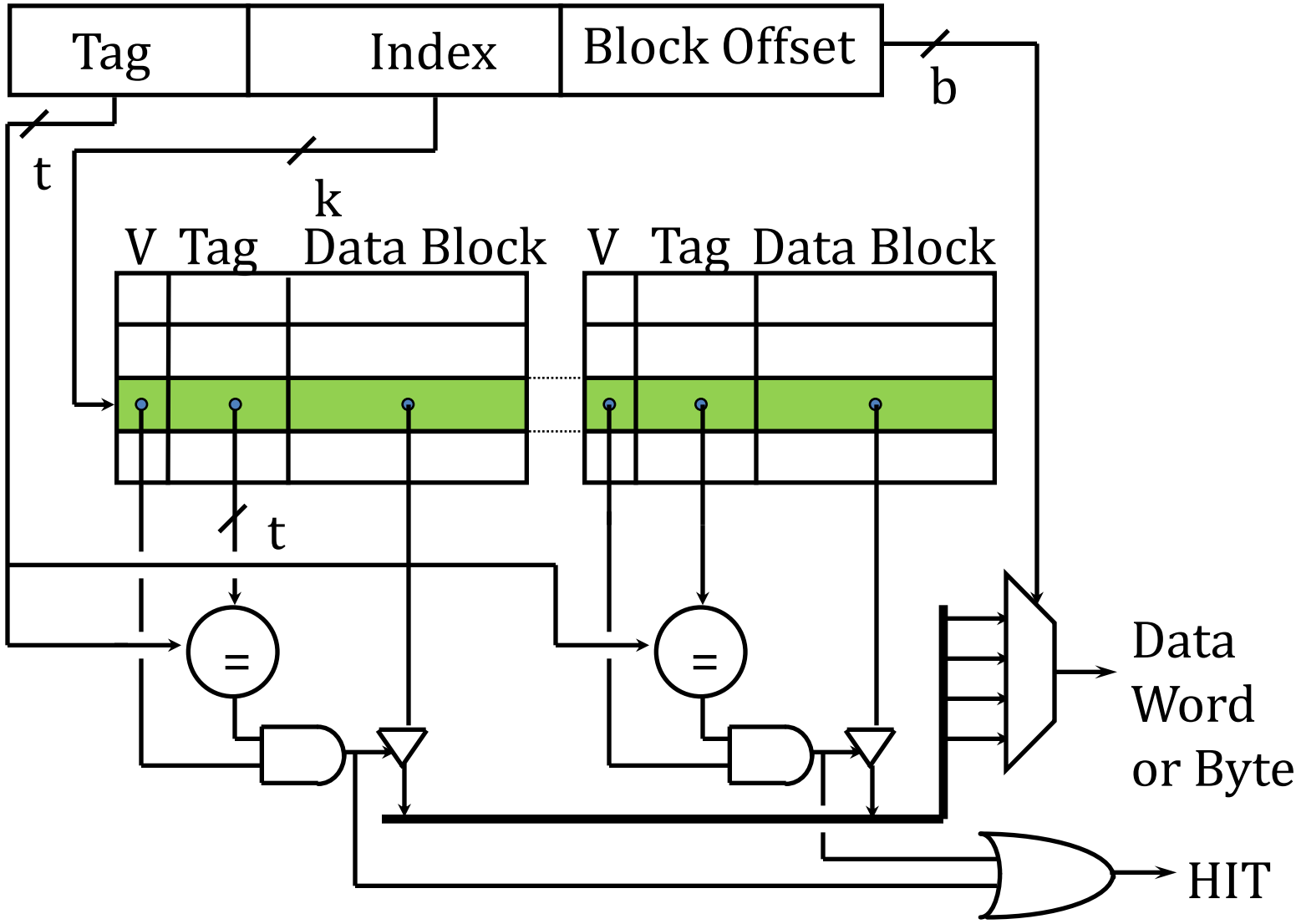
	Width
Offset	$\log_2$(block size)
Index	$\log_2$(number of sets)
Tag	n-offset-index

Direct Mapped Cache



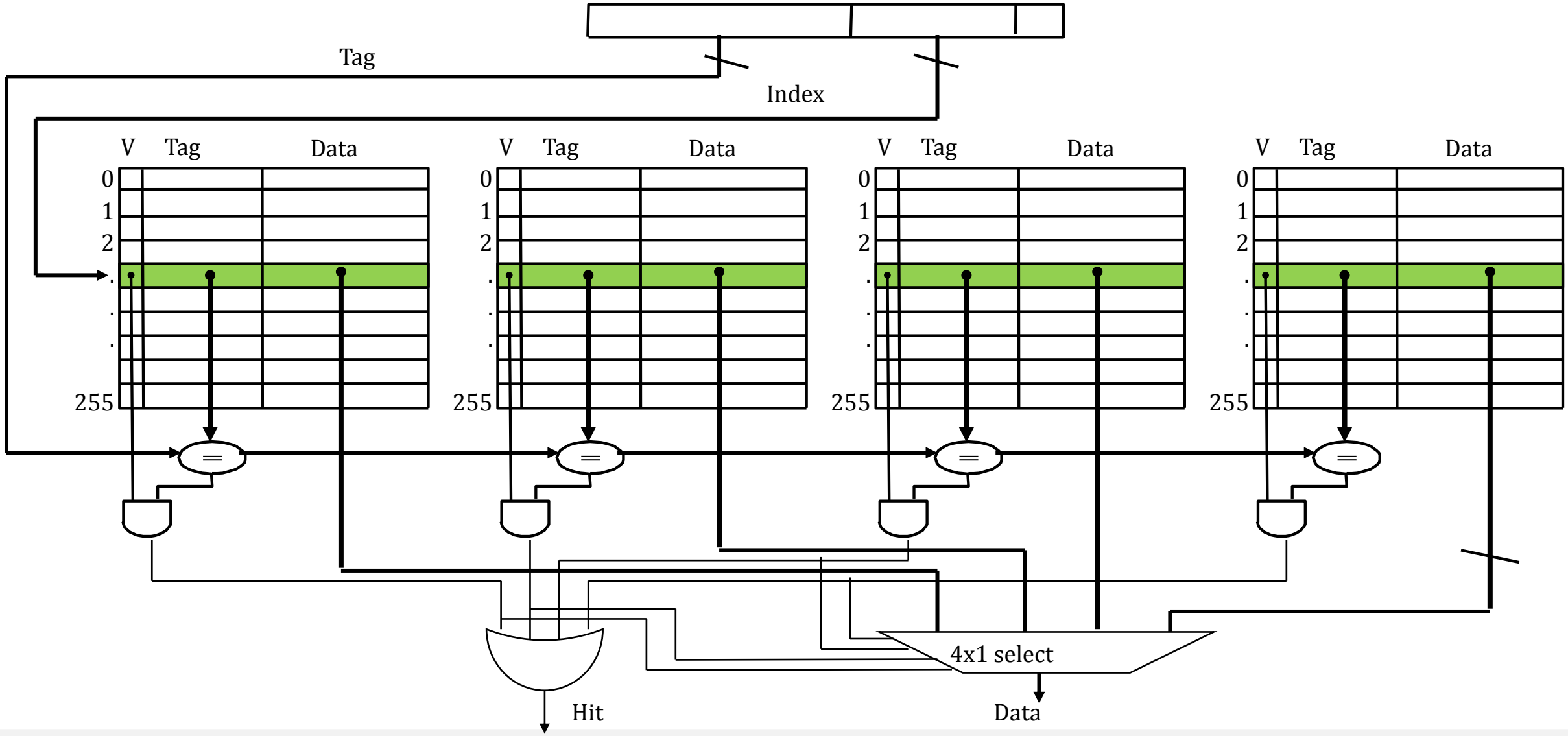
Source: 6.823, MIT

2-way Set Associative

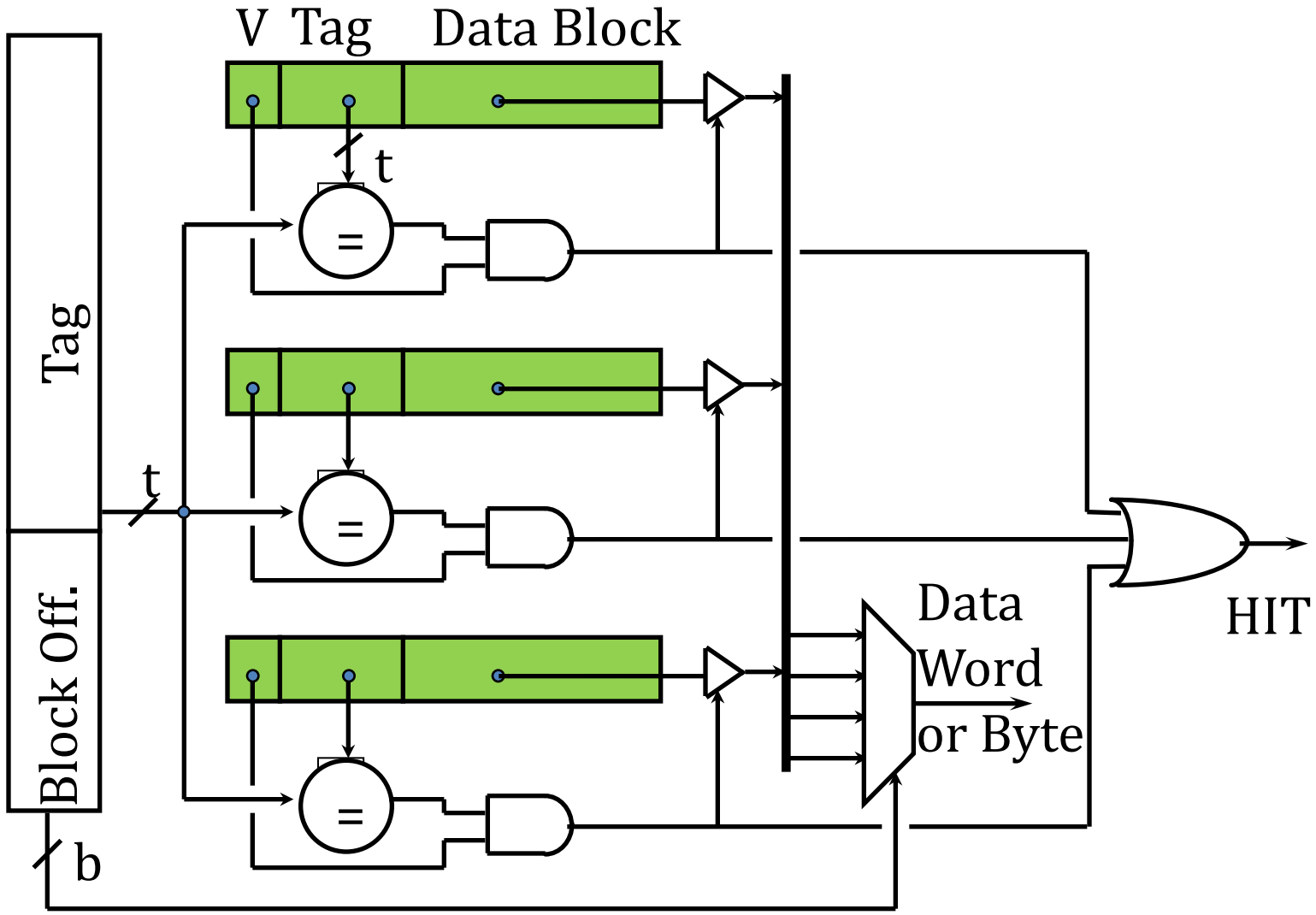


Source: 6.823, MIT

4-way



Fully-associative



Source: 6.823, MIT

Cache Misses

- Compulsory: first access to a block *(*cold start) misses
  - Infinite Cache??
- Capacity: Limited cache size too small to keep the working set
  - Infinite Cache??
- Conflict: Collisions because of associativity
  - Fully-associative??

Average Memory Access Time

Average memory access time = Hit time + Miss rate x Miss penalty

Cache Optimizations: Way-prediction, multi-level caches, prefetching, banked cache, critical word first, non-blocking cache, victim cache

Try to Find out in Your Laptop

#levels in cache, private/shared?

Cache size, associativity, block size

Hit/Miss Latencies

Replacement Policies

Additional Readings

https://www.ll.mit.edu/HPEC/agendas/proc04/invited/patterson_keynote.pdf

http://www.ecs.umass.edu/ece/koren/architecture/Cache/default.htm

http://www.ecs.umass.edu/ece/koren/architecture/Cache/page3.htm

http://www.ecs.umass.edu/ece/koren/architecture/Cache/frame1.htm

http://www.ecs.umass.edu/ece/koren/architecture/Cache/frame2.htm

For Programming Assignments

- Start Early: From Day 1 to Day n-1
- You have to submit a report containing results/insights and your code
- You have to present your findings through a 8+2 mins of presentation.

Cheating is **easy**. Try something more challenging, like being faithful.

Tutorial 1

Next Lecture

Welcome to CS698Y