COMP 273

Virtual Memory & Cache Hierarchies

Part 2

Prof. Joseph Vybihal

COMP 273
Introduction to Computer Systems

Announcements

COMP 273
Introduction to Computer Systems

Part 1

Programming with Virtual Addresses

Virtual Addressing

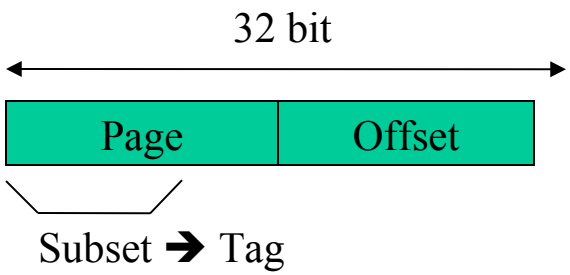
```
        .text
        .globl Start

Start:  :
        :
        lw $1, Value
        :
        :

        .data
Value:  .asciiz "Value = "
```

Compiled as offsets

Loaded into RAM as Virtual Addresses:



Architecture

CACHE

TLB

Valid	Dirty	TAG	Page No.
1	0	tag	A
0	1	tag2	B

RAM

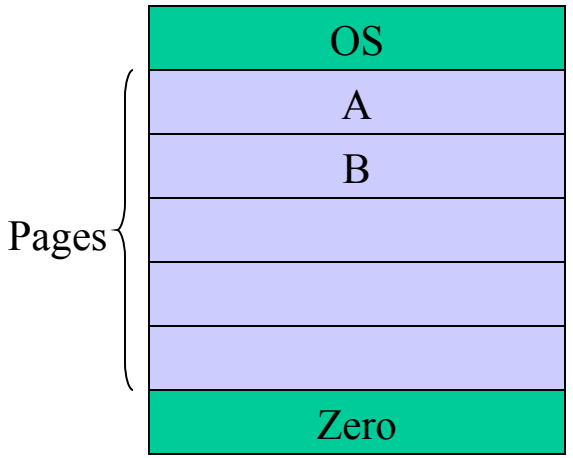
Page Table

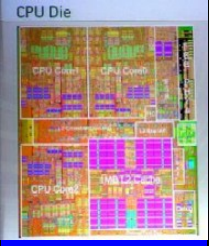
Valid	Address
1	A
1	B
0	C

Cache (the regular one)

Valid	TAG	Data
1	tag	A

RAM





Computing Real Addresses

Version One

Tag TLB	Cache :: Index

TLB hit & in cache

Version Two

Page TLB "A"	"A" :: Offset

TLB hit & not in cache

Version Three

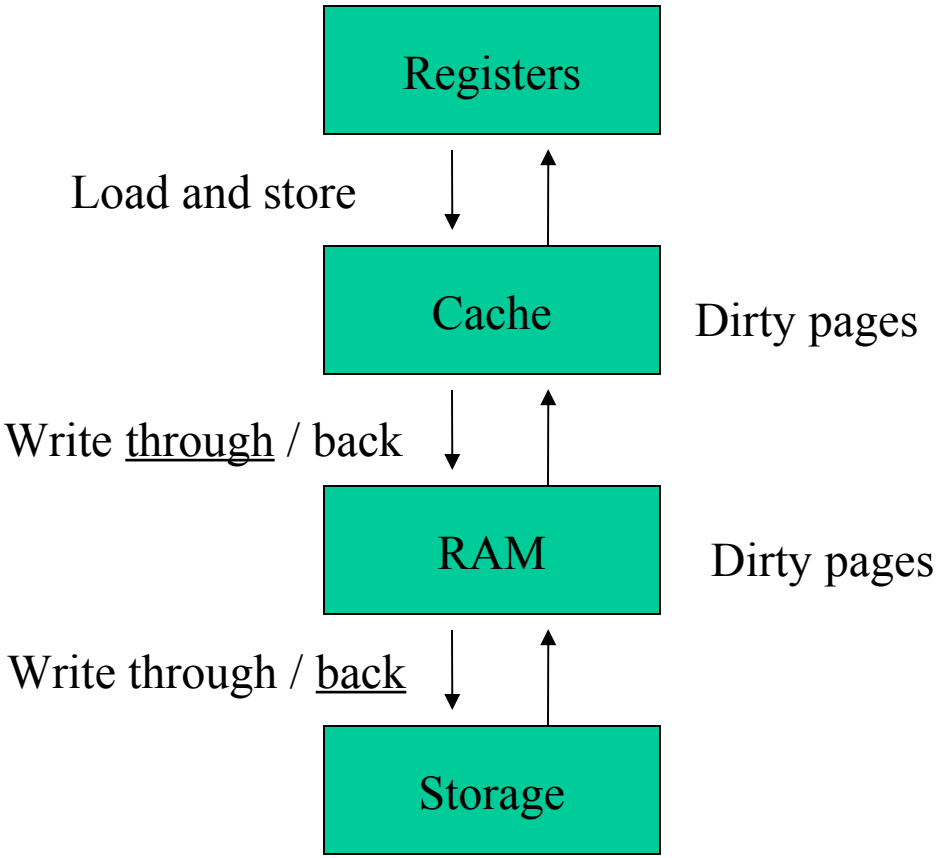
Page Table "B"	"B" :: Offset

TLB miss & in RAM

OS Interaction

- CPU Controllable
  - RAM Page Table register present
  - TLB present
- OS Controllable
  - Page faults
  - RAM Page Table
  - Dirty bit management
    - Write back (copy page to storage)
    - Write through (access storage directly)

Memory Hierarchy



COMP 273
Introduction to Computer Systems

Part 2

Calculating Cache Performance

Properties of Cache Performance

- CPU Time =
  (Prog. execution cs + MemoryStall cs) * csTime
- MemoryStall cs = ReadStall cs + WriteStall cs
- ReadStall cs =
  ProgramReads * ReadMissRate * ReadMissPenalty
- WriteStall cs =
  (ProgramWrites * WriteMissRate * WriteMissPenalty) + WriteBufferStalls

Basic Cache Performance Calculations

- Assume:
  - Instruction cache miss rate is 2%
  - Data cache miss rate is 4%
  - Cycles per Instruction (CPI) = 2
  - Miss penalty of 40 cs
  - Loads and stores is 36% of instructions
- Speed improvement with a perfect cache with zero misses?
  - Instruction miss cycles = N * 2% * 40 = 0.8N
  - Data miss cycles = N * 36% * 4% * 40 = 0.58N
  - Totals memory stalls = 0.8N + 0.58N = 1.38N
  - CPI & Stalls = 2 + 1.38 = 3.38
  - Ratio = (CPU Stall / CPU perfect) = 3.38 / 2 = 1.69

Faster Execution?

- Assume:
  - CPI increased from 2 to 1
- Improvement:
  - CPI with stalls = 1 + 1.38 = 2.38
  - Perfect cache = 2.38 / 1 = 2.38
- Implication
  - Time spent on memory stalls =
    - 1.38 / 3.38 = 41%, to
    - 1.38 / 2.38 = 58% !!

Optimizing execution ALONE does not help.

Increased Clock Rate

- Assume:
  - Same scenario as in previous examples
  - Clock rate doubled
- How much faster will the new machine be?
  - Miss penalty of 80 cs (given new speed)
  - Miss CPI = (2% * 80) + (36% * 4% * 80) = 2.75
  - New machine CPI = 2 + 2.75 = 4.75
  - Performance fast / Performance slower

    IC * CPI_{slow} * CS / IC * CPI * (CS / 2) =

    3.38 / 4.75*0.5 = 1.41 !!

Not by a factor of 2.

What to do?

- Did it help...
  - Perfect? 1.38
  - No optimization? 3.38
  - Make instruction exec faster? 2.38
  - Speed up clock? 1.41

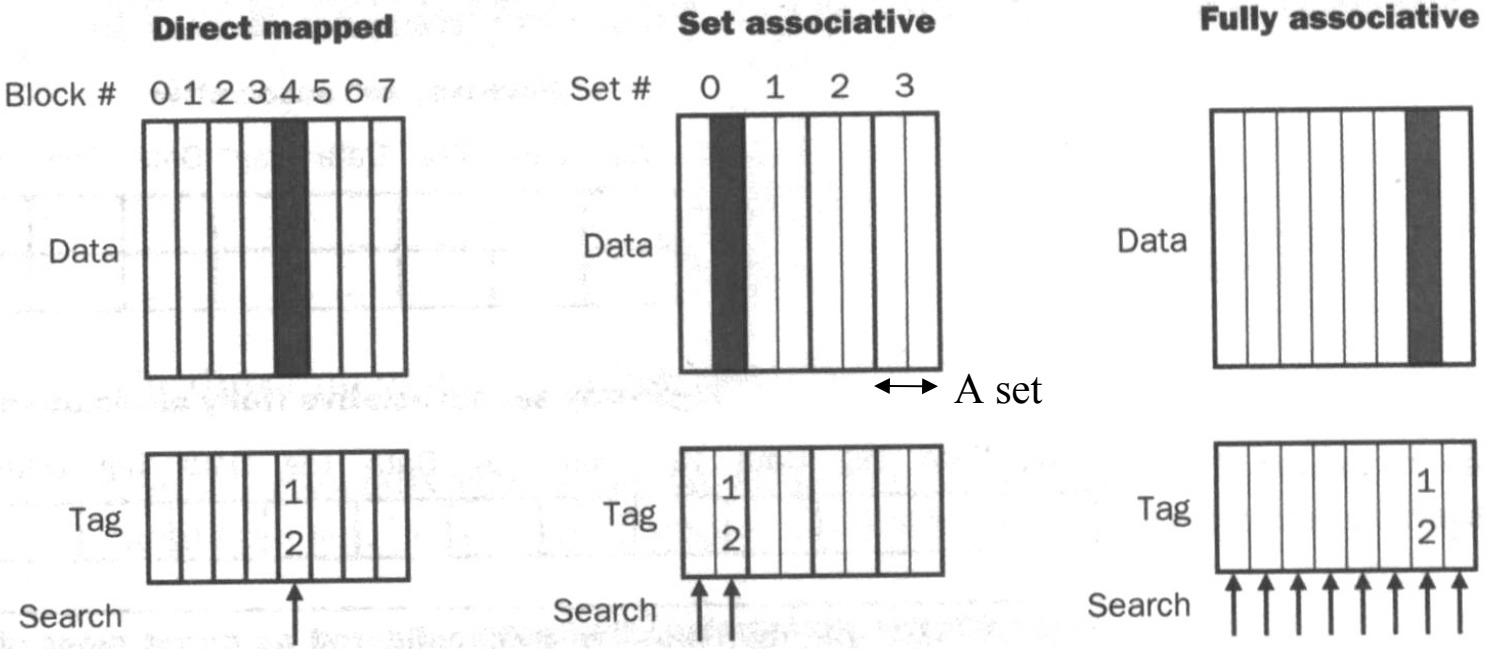
COMP 273
Introduction to Computer Systems

Part 3

Reducing Cache Misses

Associativity

Assume address 12



Direct: Block No. MODULO Number of cache blocks

Assoc: Block NO. MODULO Number of sets in cache

Associative Caches

One-way set associative (direct mapped)

Block	Tag	Data
0		
1		
2		
3		
4		
5		
6		
7		

Two-way set associative

Set	Tag	Data	Tag	Data
0				
1				
2				
3				

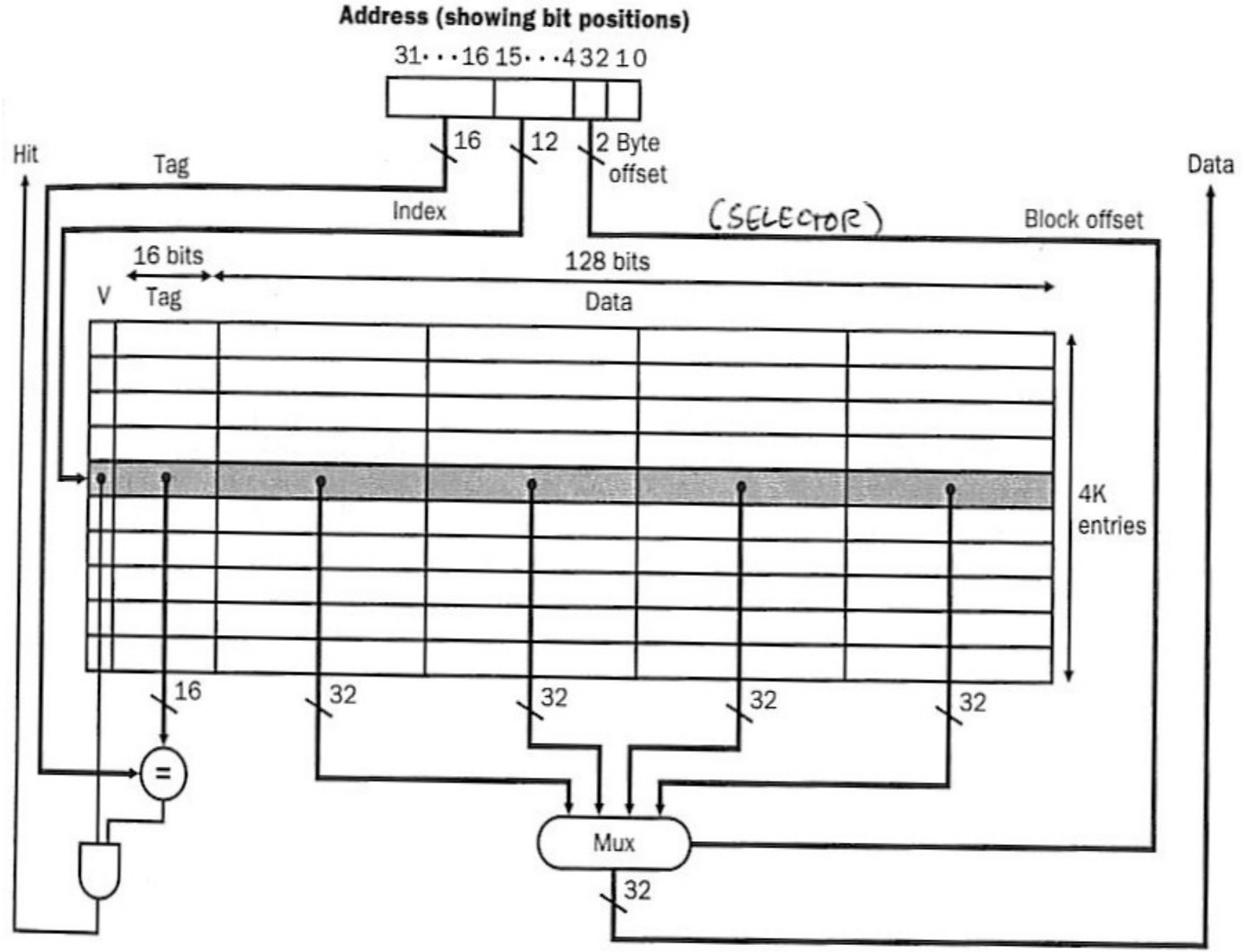
Four-way set associative

Set	Tag	Data	Tag	Data	Tag	Data	Tag	Data
0								
1								

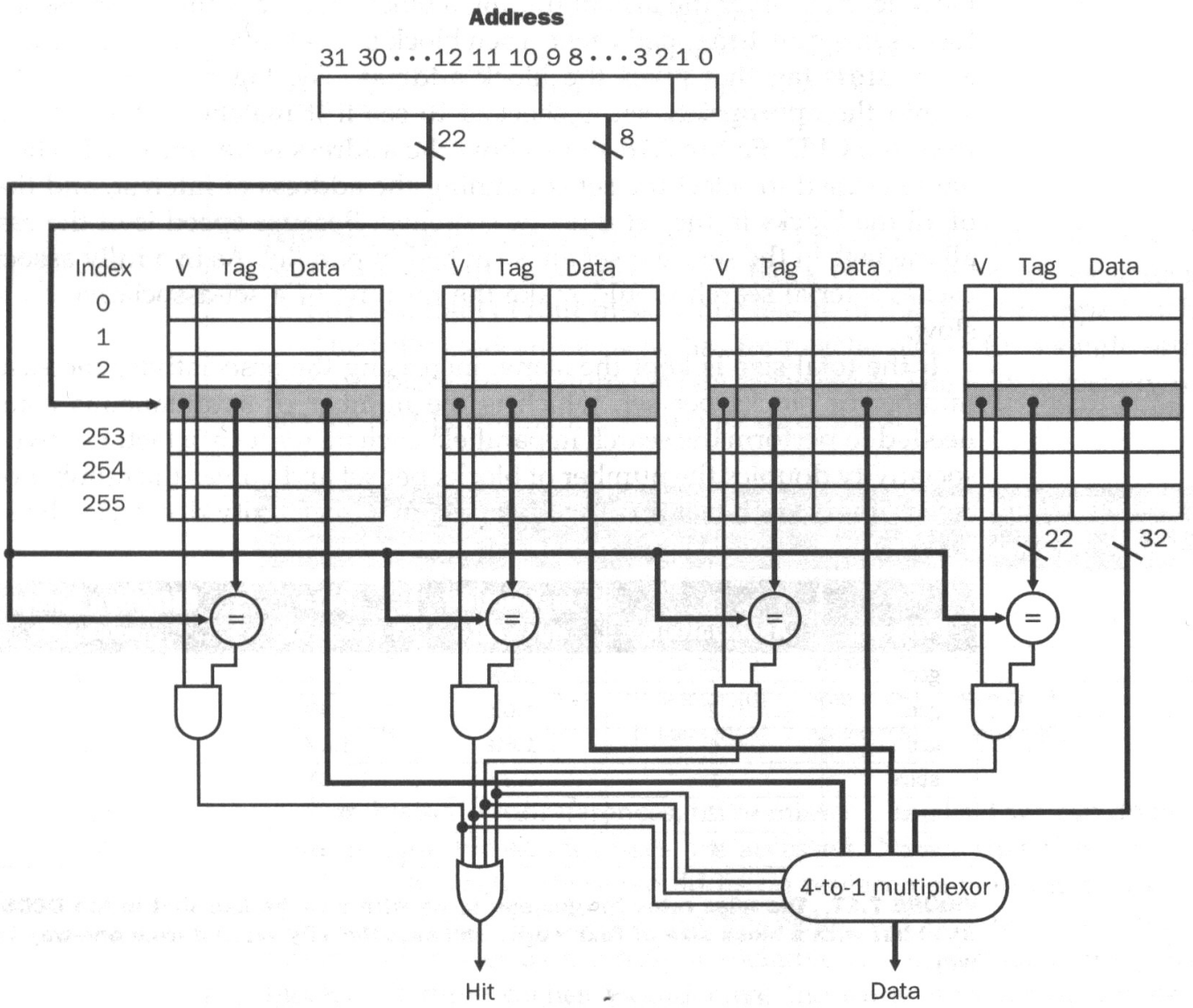
Eight-way set associative (fully associative)

Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data

Addressing in Direct Blocked Cache



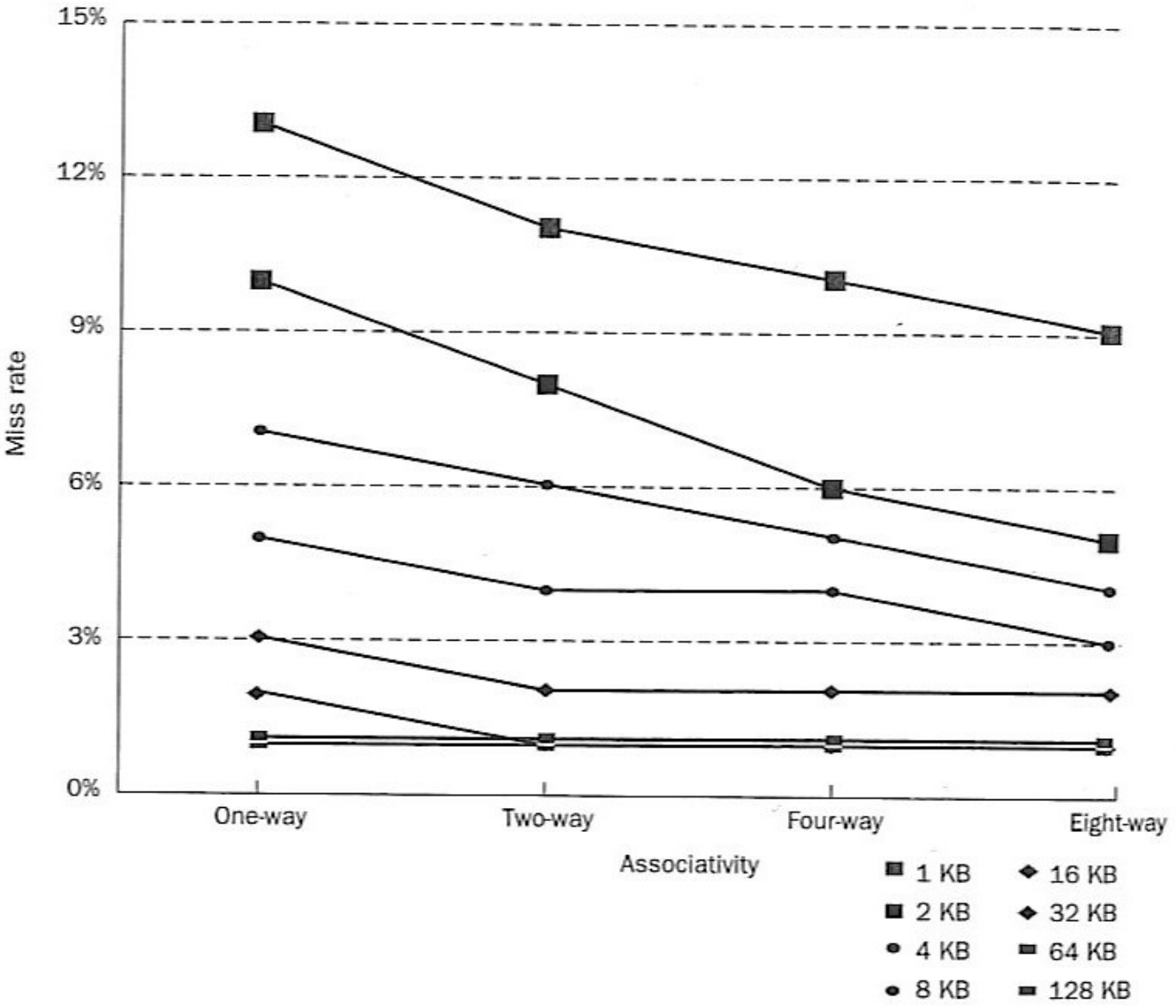
Addressing in Associative Cache



Why would this help?

- Time Locality
  - Code tends to be iterative
- Space Locality
  - Code tends to be sequential
- Question:
  - Where does it break down?
    - GOTO statements
    - Function calls
    - Global variables
    - Page sizes

Performance

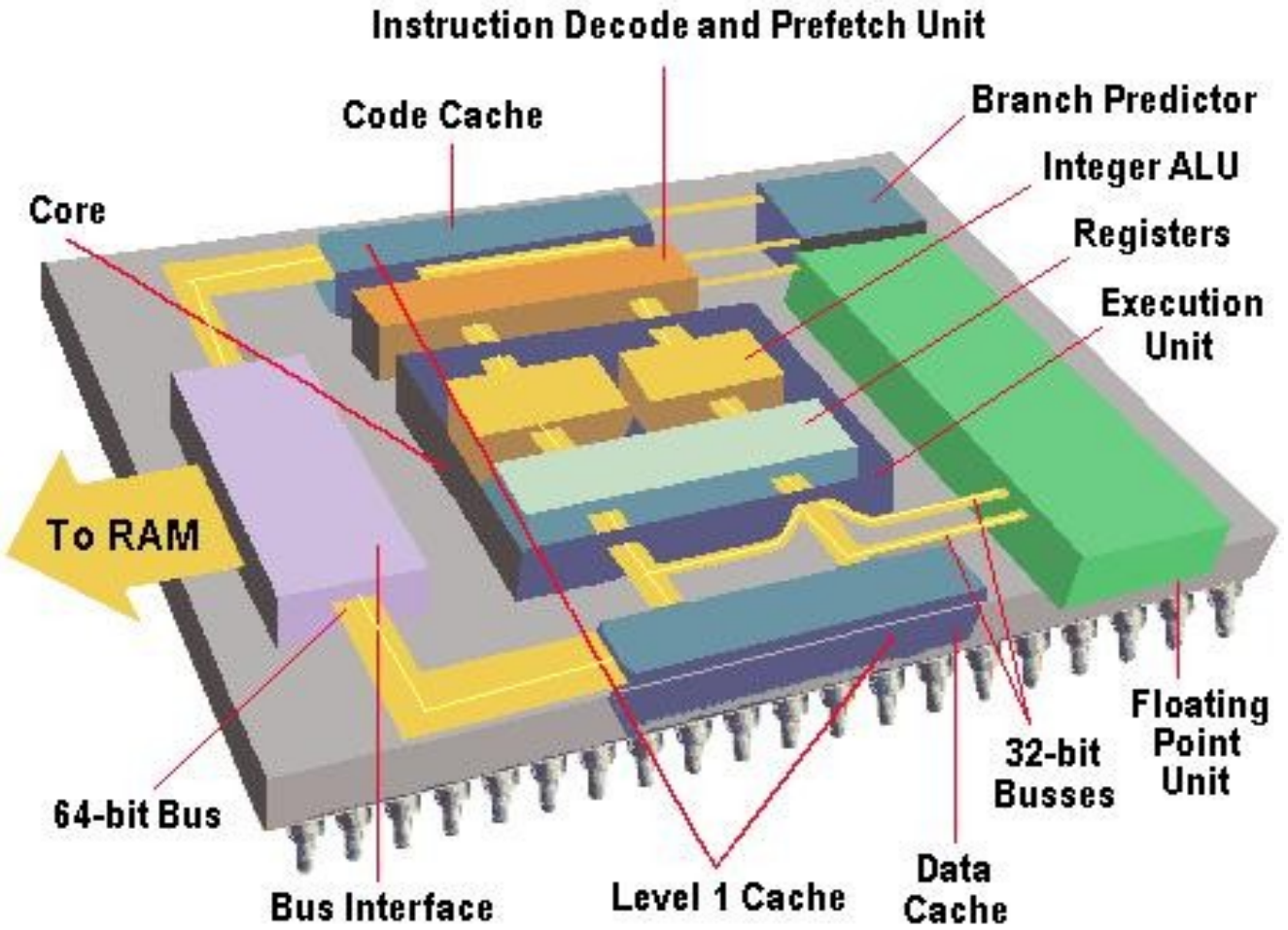


Part 4

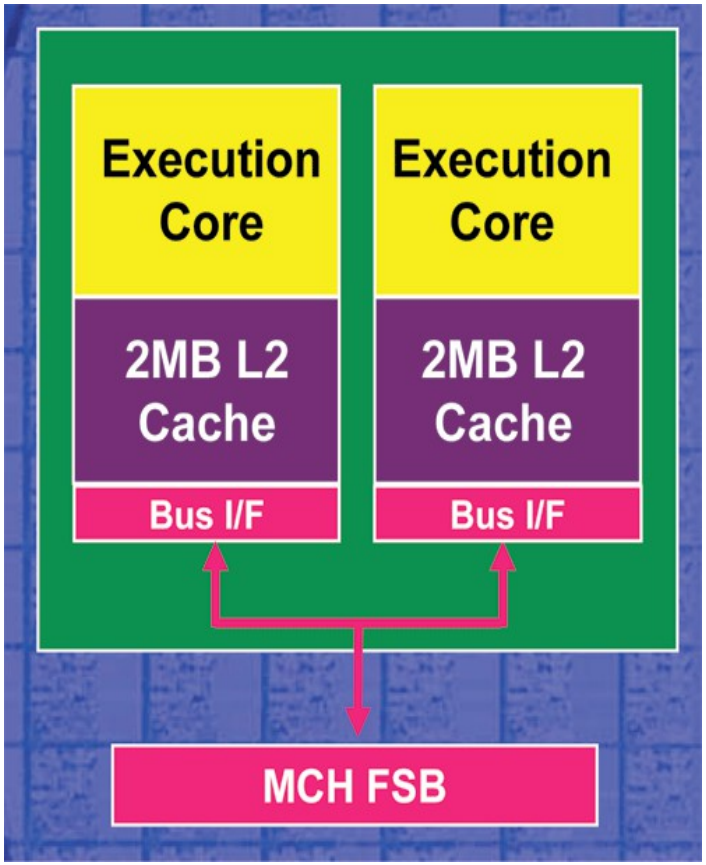
Some Real Hardware

Old Statistics

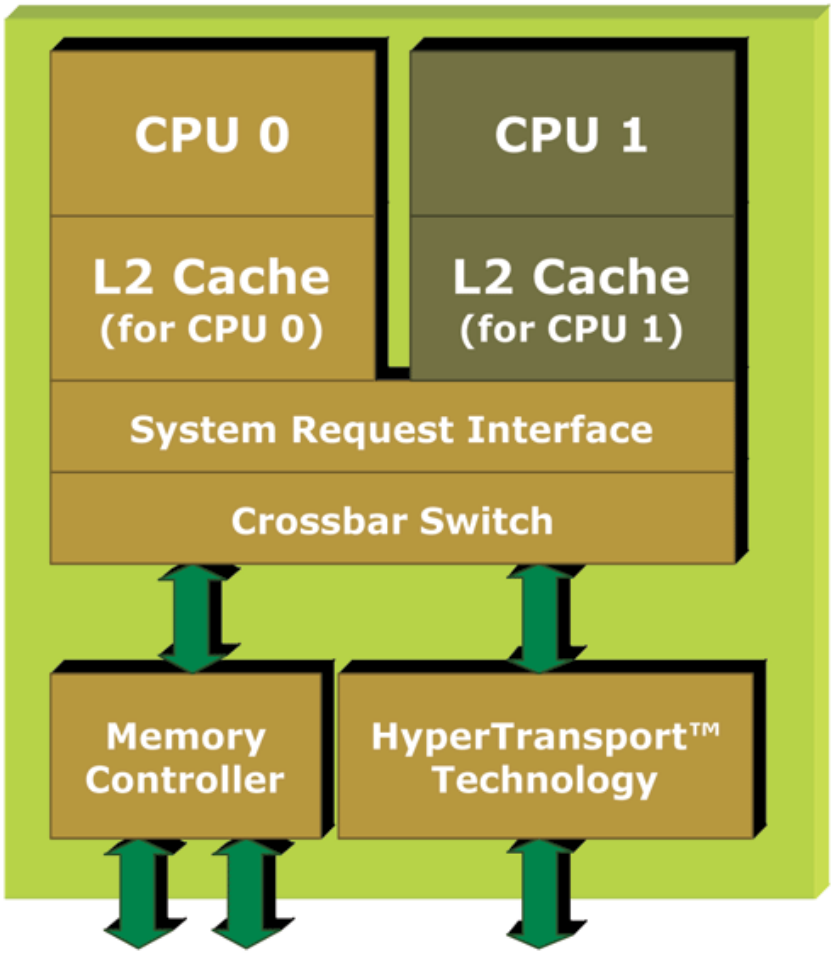
Standard CPU



Dual Core

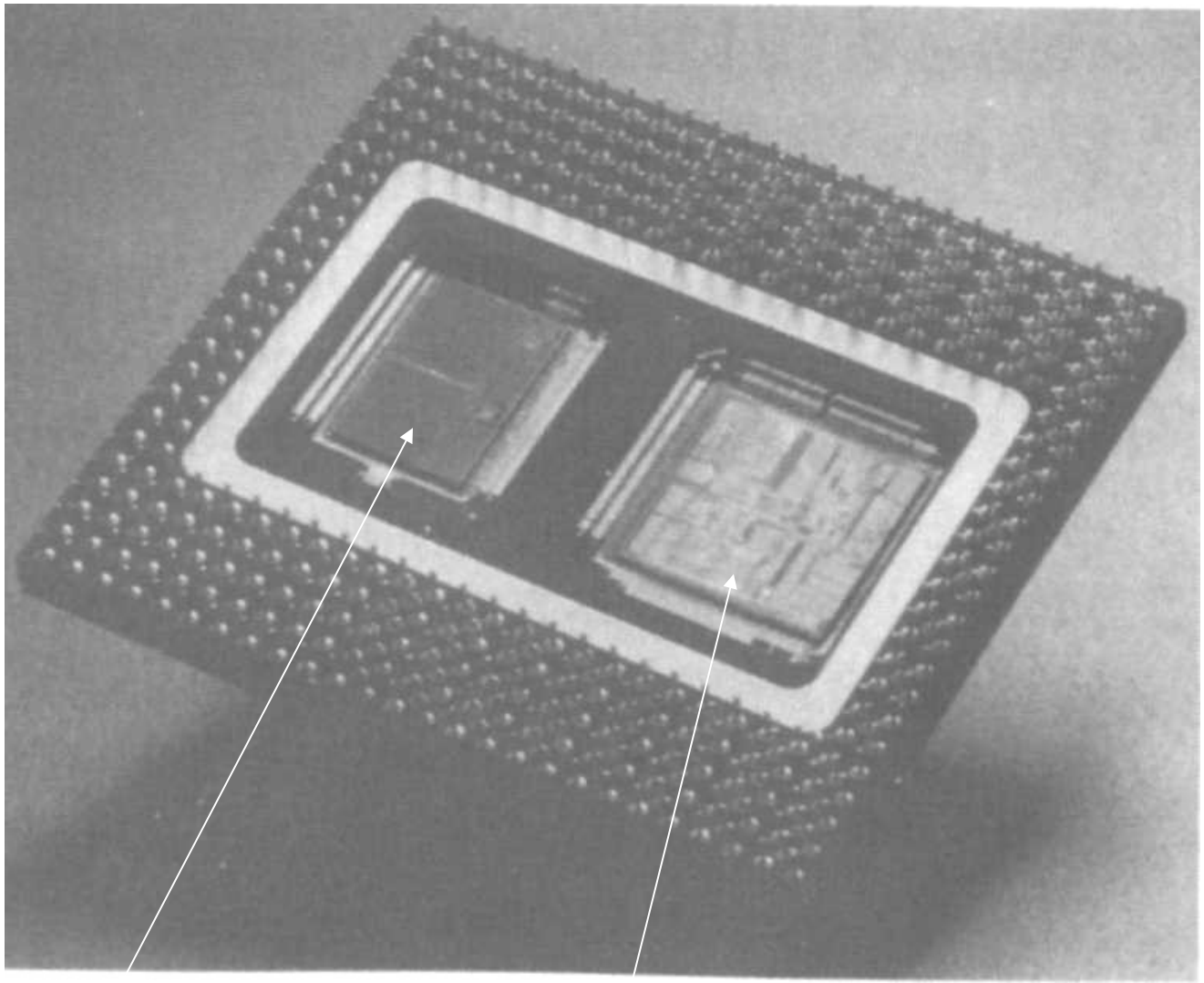


Intel's Pentium D dual core architecture



AMD Athlon™ 64 X2 Dual-Core Processor Design

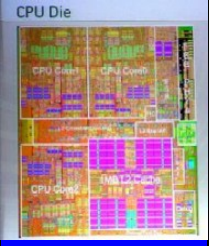
INTEL CPU CHIP



On-board Cache

The CPU

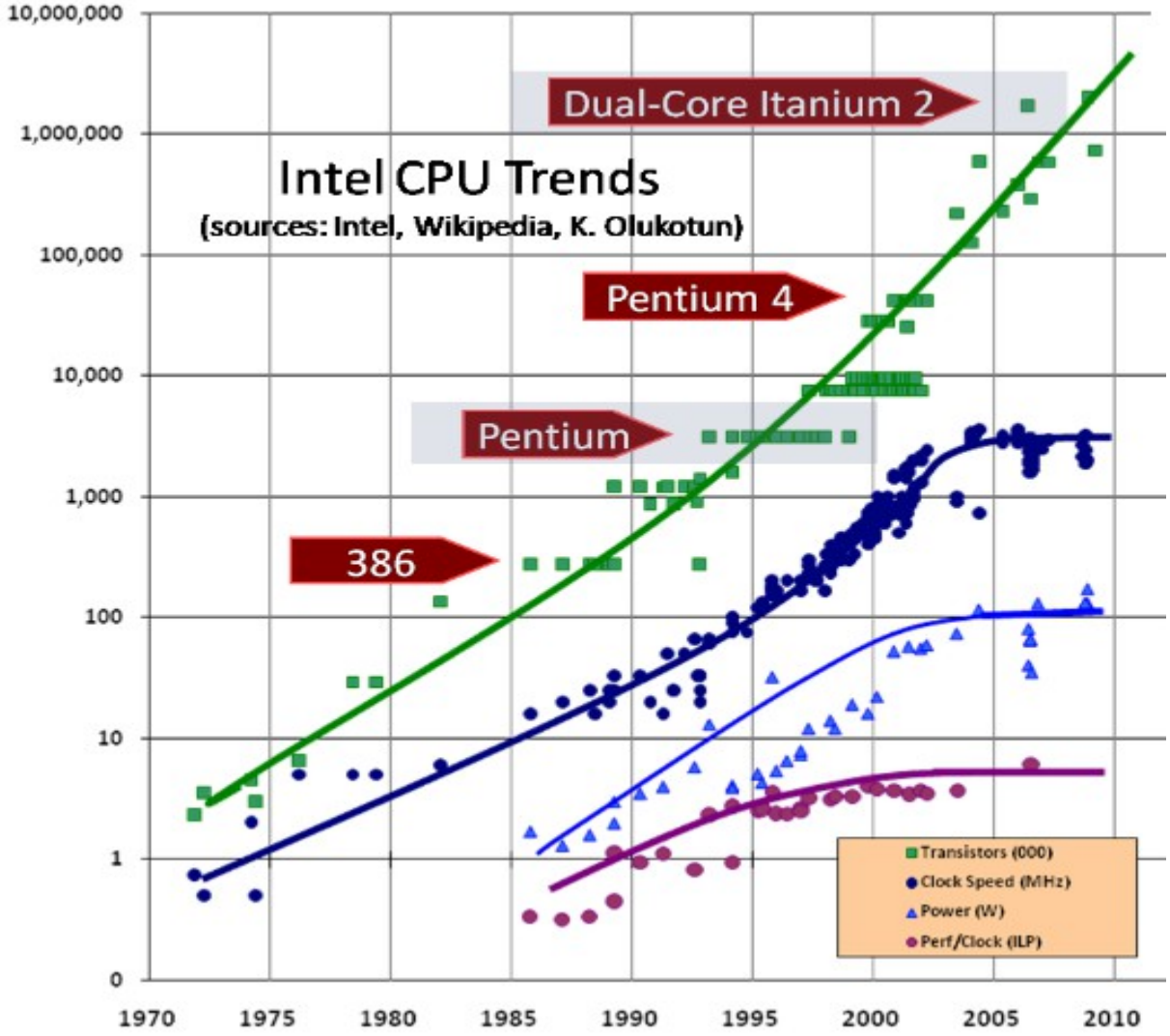
Additional Optional Information



Characteristic	Intel Pentium Pro	PowerPC 604
Virtual address	32 bits	52 bits
Physical address	32 bits	32 bits
Page size	4 KB, 4 MB	4 KB, selectable, and 256 MB
TLB organization	A TLB for instructions and a TLB for data Both four-way set associative Pseudo-LRU replacement Instruction TLB: 32 entries Data TLB: 64 entries TLB misses handled in hardware	A TLB for instructions and a TLB for data Both two-way set associative LRU replacement Instruction TLB: 128 entries Data TLB: 128 entries TLB misses handled in hardware

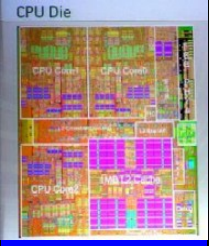
Characteristic	Intel Pentium Pro	PowerPC 604
Cache organization	Split instruction and data caches	Split instruction and data caches
Cache size	8 KB each for instructions/data	16 KB each for instructions/data
Cache associativity	Four-way set associative	Four-way set associative
Replacement	Approximated LRU replacement	LRU replacement
Block size	32 bytes	32 bytes
Write policy	Write-back	Write-back or write-through

Historical Trends

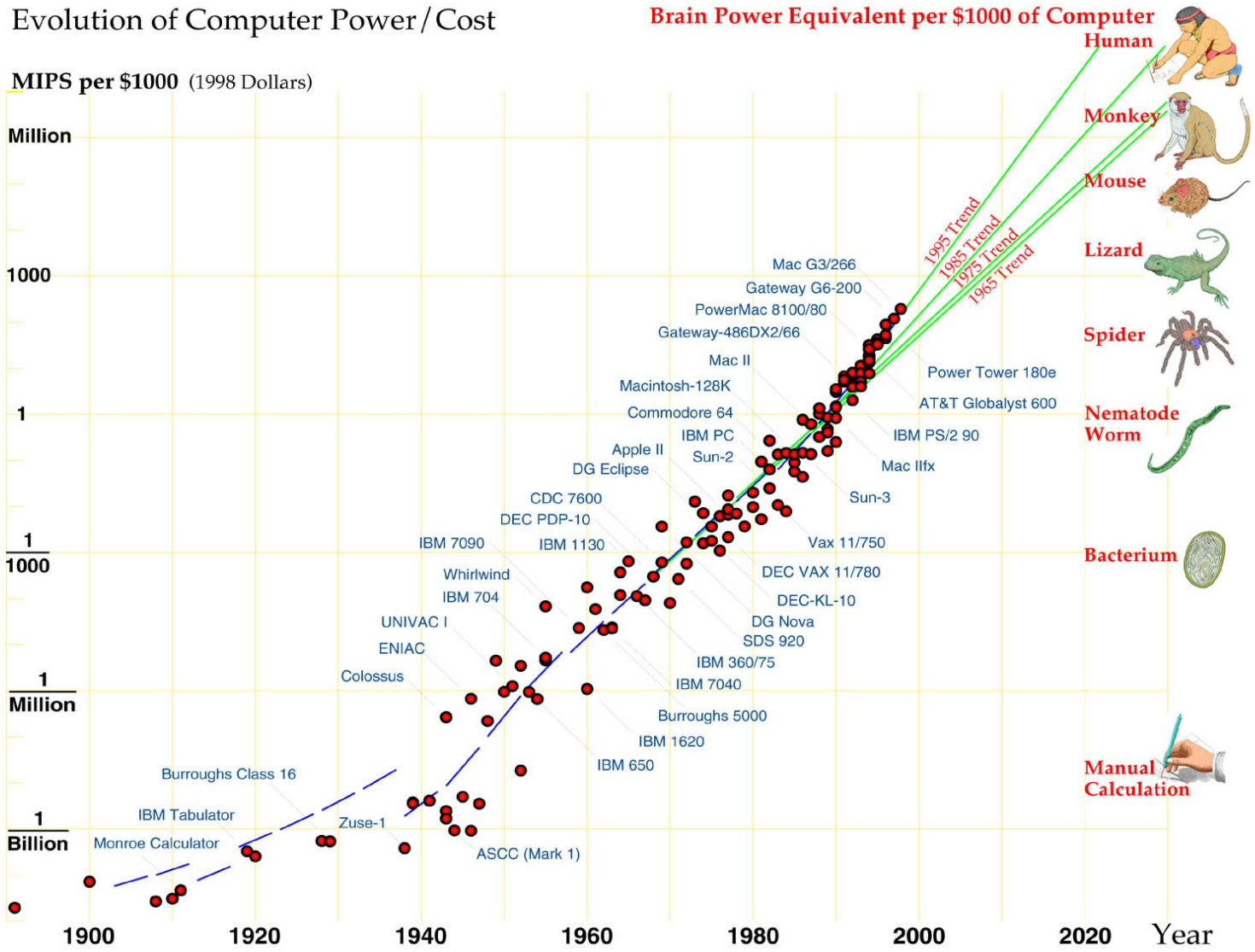


Moor's first law: CPU power doubles every 2 years

Moors 2nd law: CPU manufacturing cost doubles every 3 years


CPU Die

Trends


Evolution of Computer Power / Cost
MIPS per $1000 (1998 Dollars)
Million
1000
1
1/1000
1/Million
1/Billion
1900
1920
1940
1960
1980
2000
2020
Year
Brain Power Equivalent per $1000 of Computer
Human
Monkey
Mouse
Lizard
Spider
Nematode Worm
Bacterium
Manual Calculation
1995 Trend
1985 Trend
1975 Trend
1965 Trend
Mac G3/266
Gateway G6-200
PowerMac 8100/80
Gateway-486DX2/66
Mac II
Macintosh-128K
Commodore 64
IBM PC
Apple II
Sun-2
DG Eclipse
CDC 7600
DEC PDP-10
IBM 7090
IBM 1130
Whirlwind
IBM 704
UNIVAC I
ENIAC
Colossus
Burroughs Class 16
IBM Tabulator
Monroe Calculator
Zuse-1
ASCC (Mark 1)
IBM 650
IBM 1620
Burroughs 5000
IBM 7040
IBM 360/75
SDS 920
DG Nova
DEC-KL-10
DEC VAX 11/780
Vax 11/750
Sun-3
Mac IIfx
IBM PS/2 90
AT&T Globalyst 600
Power Tower 180e