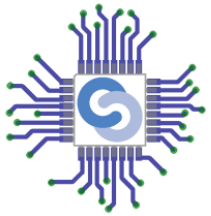


Calculatoare Numerice

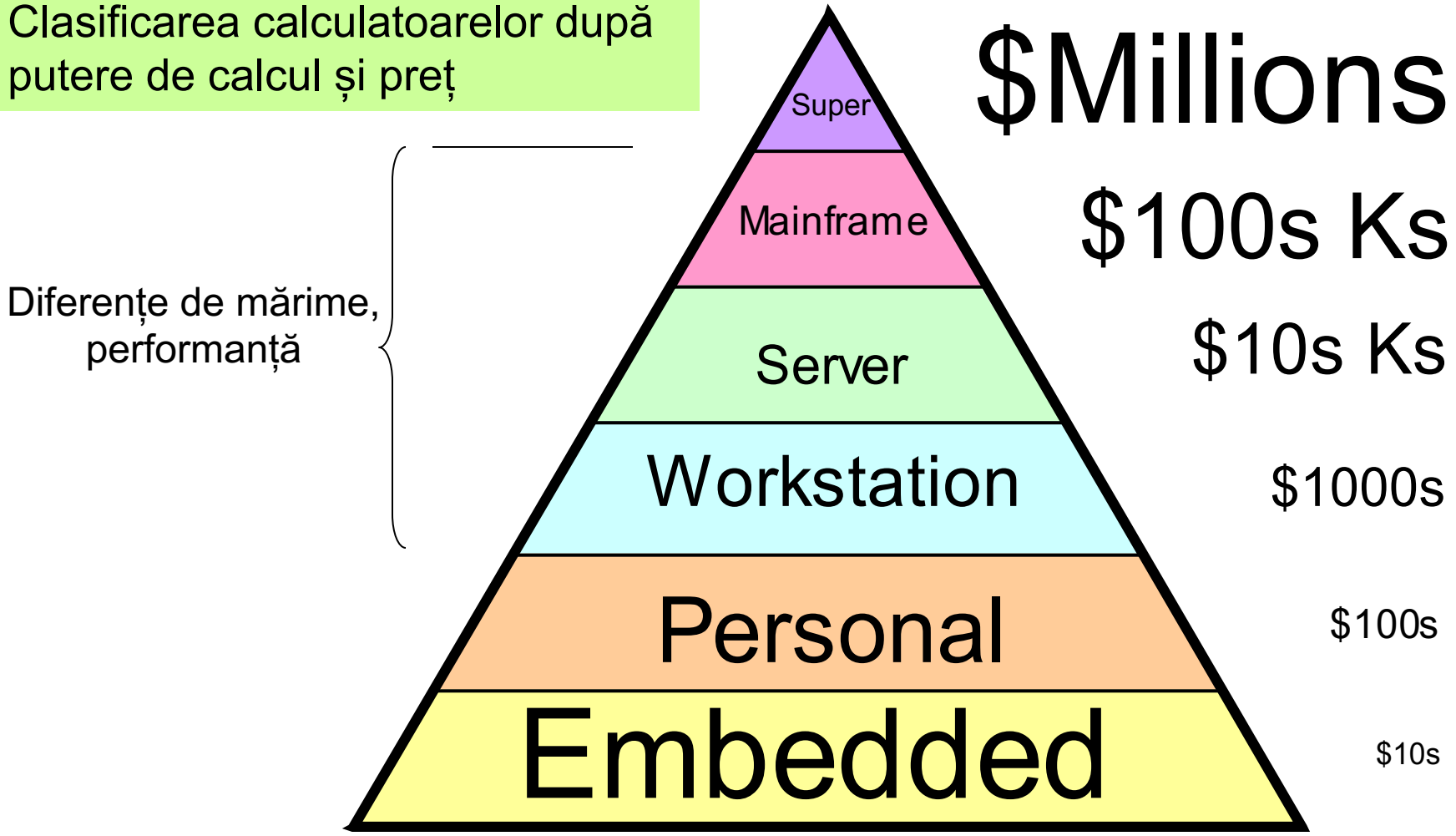
– Cursul 9 –

Măsurarea Performanței

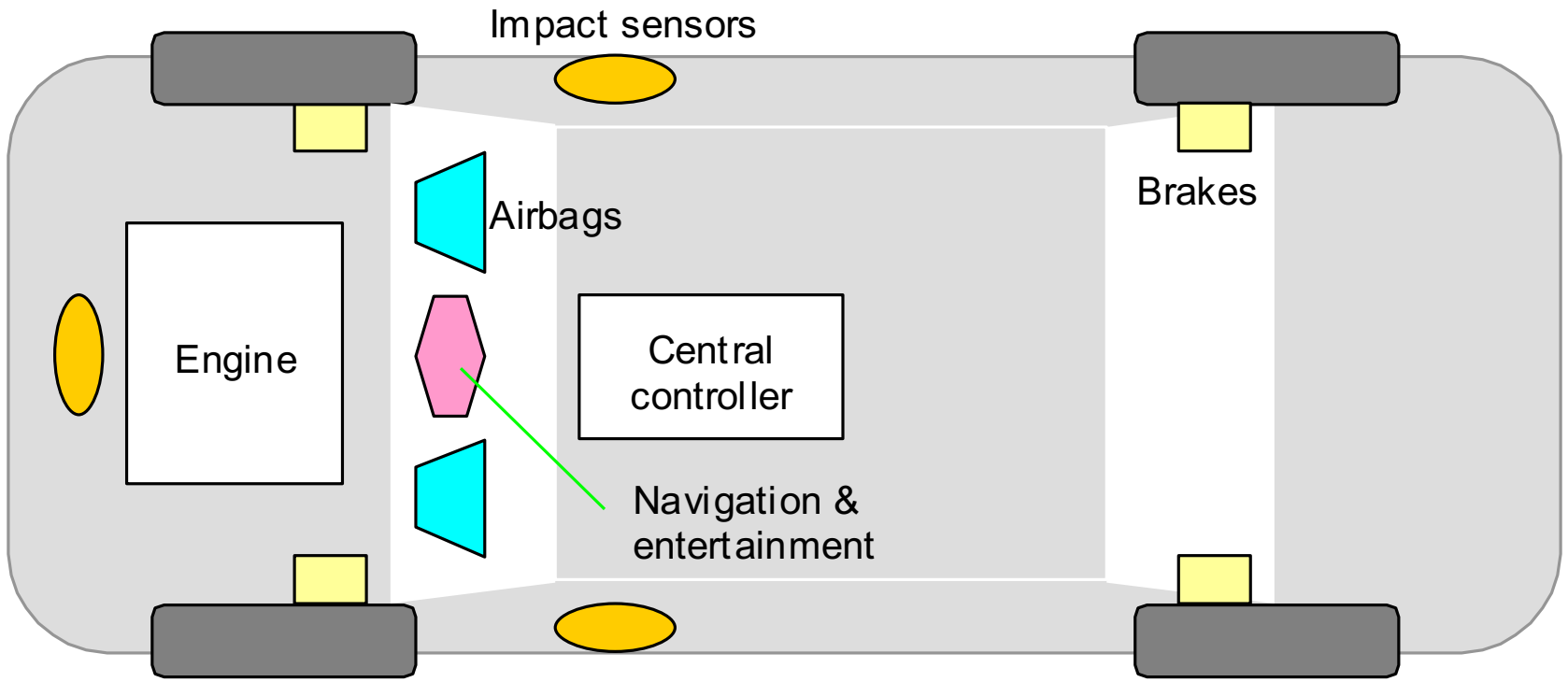
Facultatea de Automatică și Calculatoare
Universitatea Politehnica București

Piramida Preț/Performanță

Clasificarea calculatoarelor după putere de calcul și preț

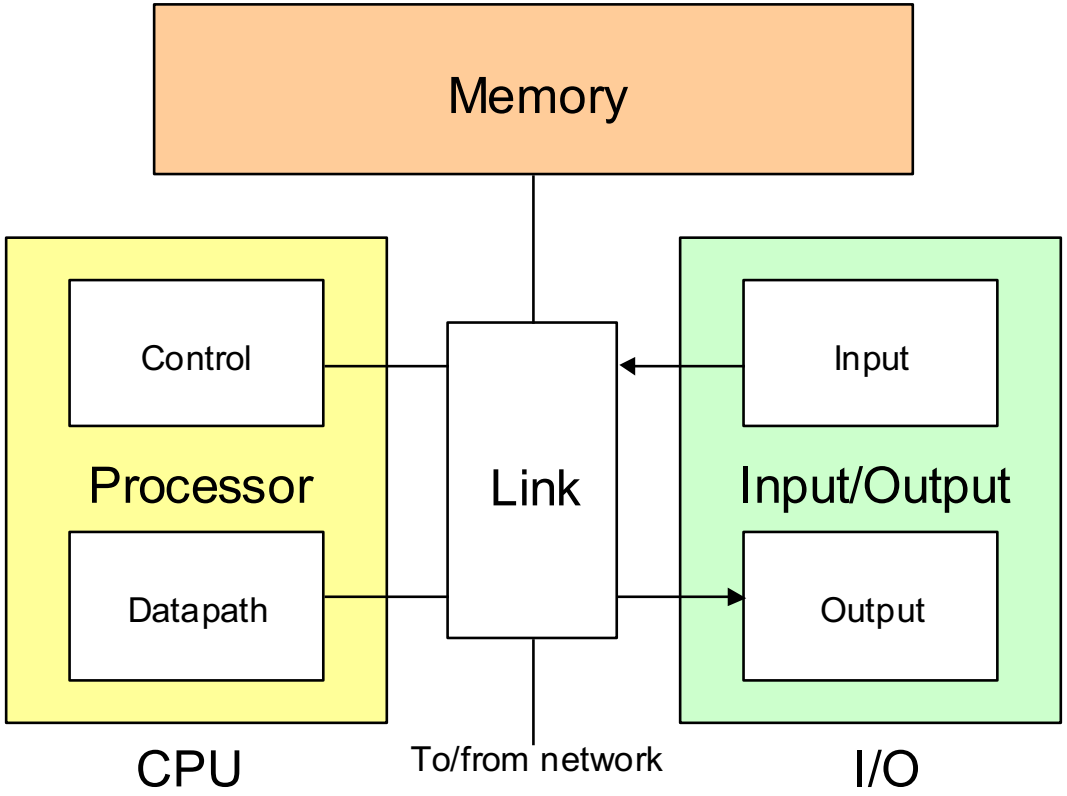


Calculatoare embedded automotive



Procesoarele embedded sunt ubicue și totuși invizibile. Pot fi găsite în automobile, electronice, electrocasnice și multe alte produse.

Subsistemele unui calculator digital



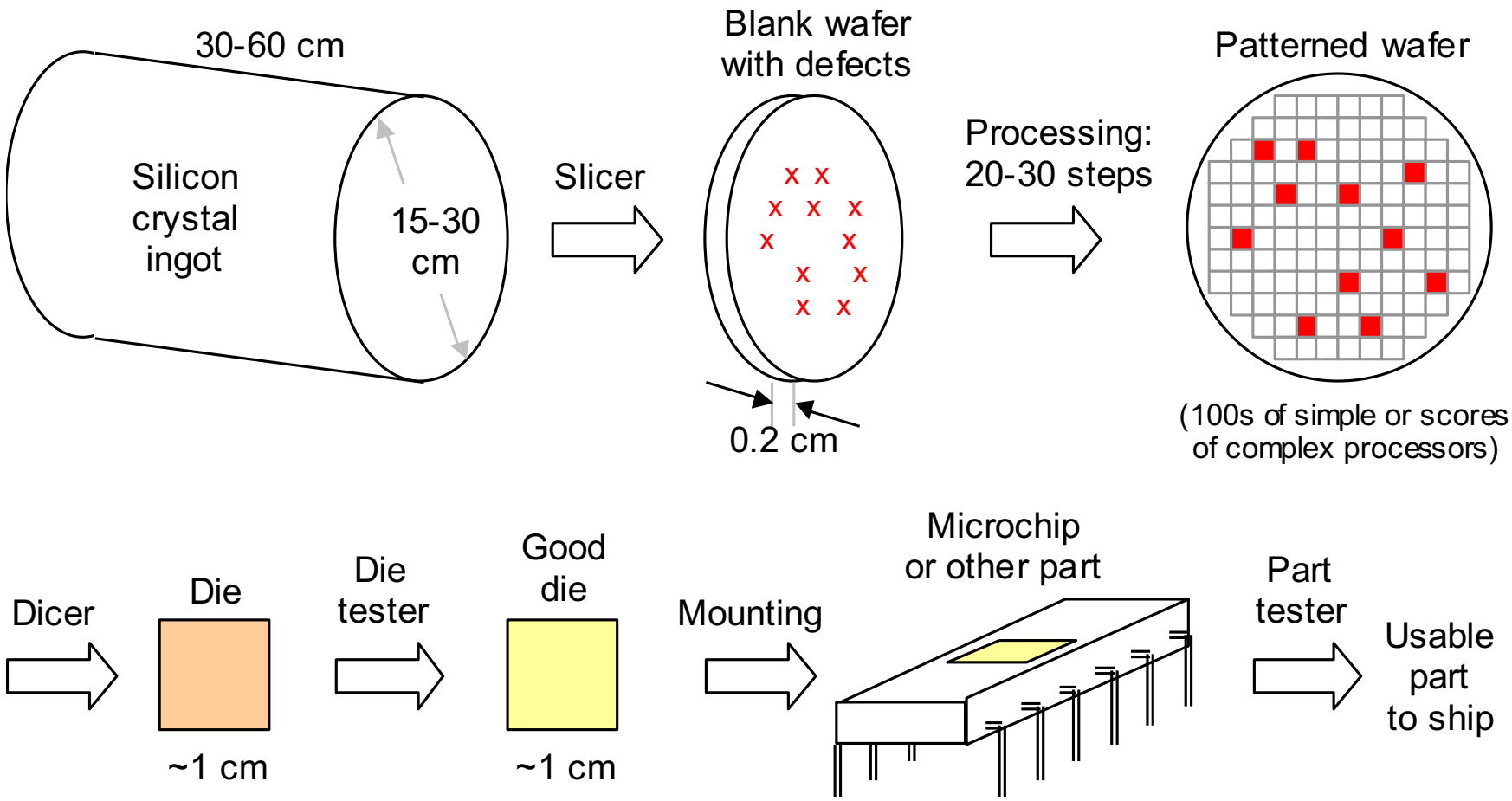
Cele (trei, patru, cinci sau) șase componente principale ale unui calculator numeric. De obicei, unitatea de legătură (link unit), care poate să fie o magistrală simplă sau o matrice de interconexiuni, nu este inclusă în acest tip de diagramă.

Progresul pe generații

Cele cinci generații de calculatoare digitale

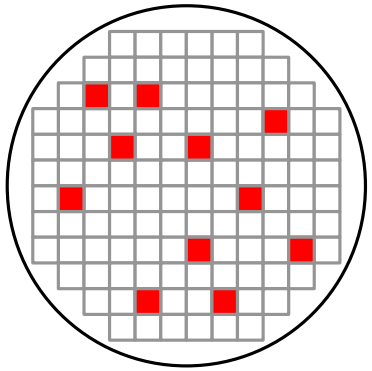
Generația (data început)	Tehnologia procesorului	Sistem memorie	Dispozitive I/O noi	Look & feel
0 (1600s)	(Electro-) mecanic	Roți, cartele	Pârghii, cartele perforate	Echipament de fabrică
1 (1950s)	Tuburi cu vid	Tambur magnetic	Bandă hârtie & magnetică	Echipament cât o sală mare
2 (1960s)	Tranzistoare	Miez de ferită	Imprimantă, terminal text	Cameră server
3 (1970s)	SSI/MSI	RAM/ROM	Disc, tastatură, monitor	Echipament de birou
4 (1980s)	LSI/VLSI	SRAM/DRAM	Rețea, CD, mouse,sunet	Desktop/ laptop micro
5 (1990s)	ULSI/GSI/ WSI, SOC	SDRAM, flash	Senzori/actuatoare , point&click, touch	Invizibil, embedded

Producerea (și productivitatea) circuitelor integrate

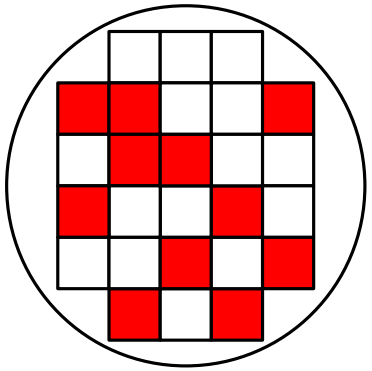


Procesul de fabricație pentru un circuit integrat

Efectele mărimii matriței asupra productivității



120 dies, 109 good



26 dies, 15 good

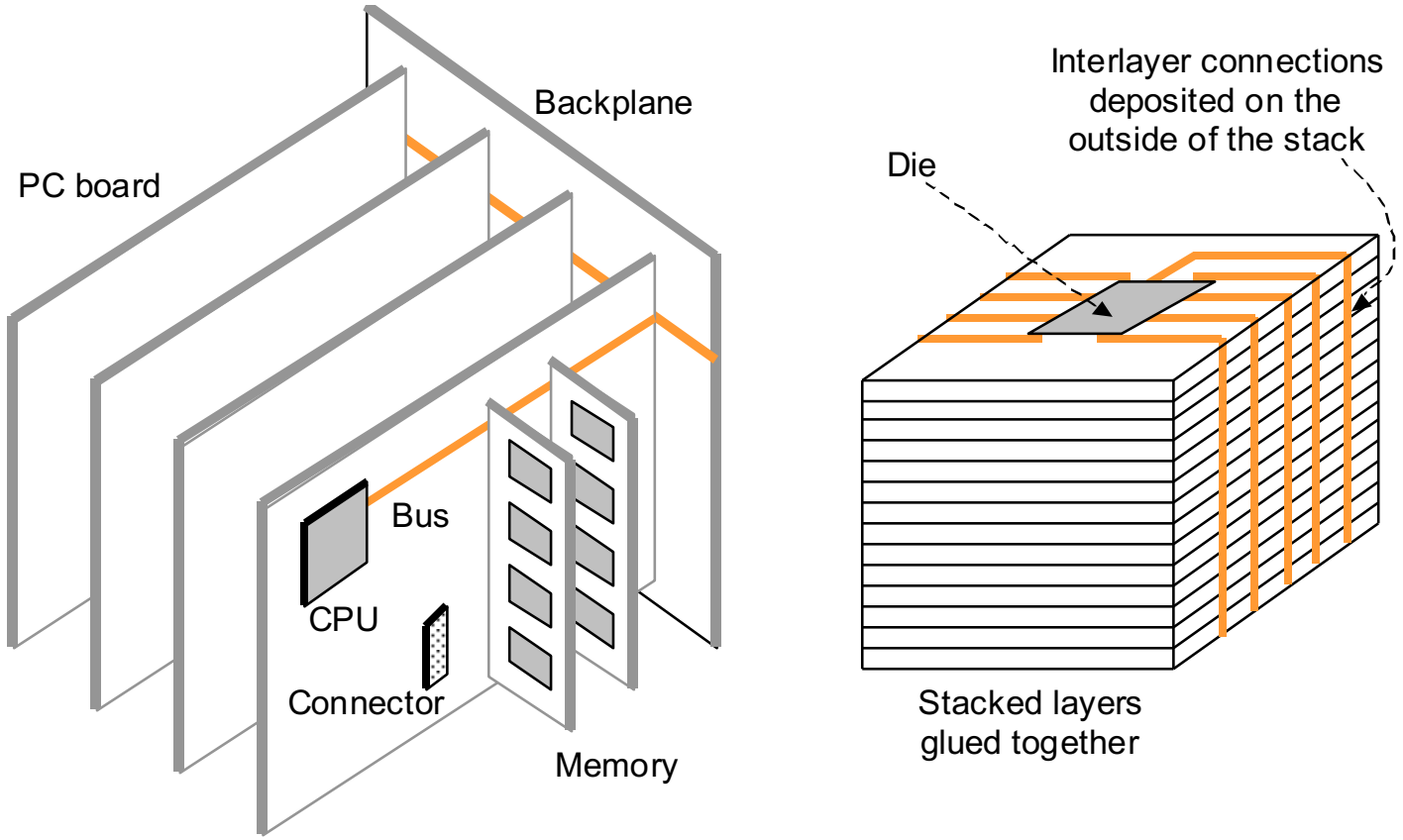
Scăderea dramatică a productivității cu creșterea suprafeței chip-ului

Die yield $=_{def}$ (number of good dies) / (total number of dies)

Die yield = Wafer yield $\times$ [1 + (Defect density $\times$ Die area) / a]$^{-a}$

Die cost = (cost of wafer) / (total number of dies $\times$ die yield)
= (cost of wafer) $\times$ (die area / wafer area) / (die yield)

Tehnologii pentru procesoare și memorii

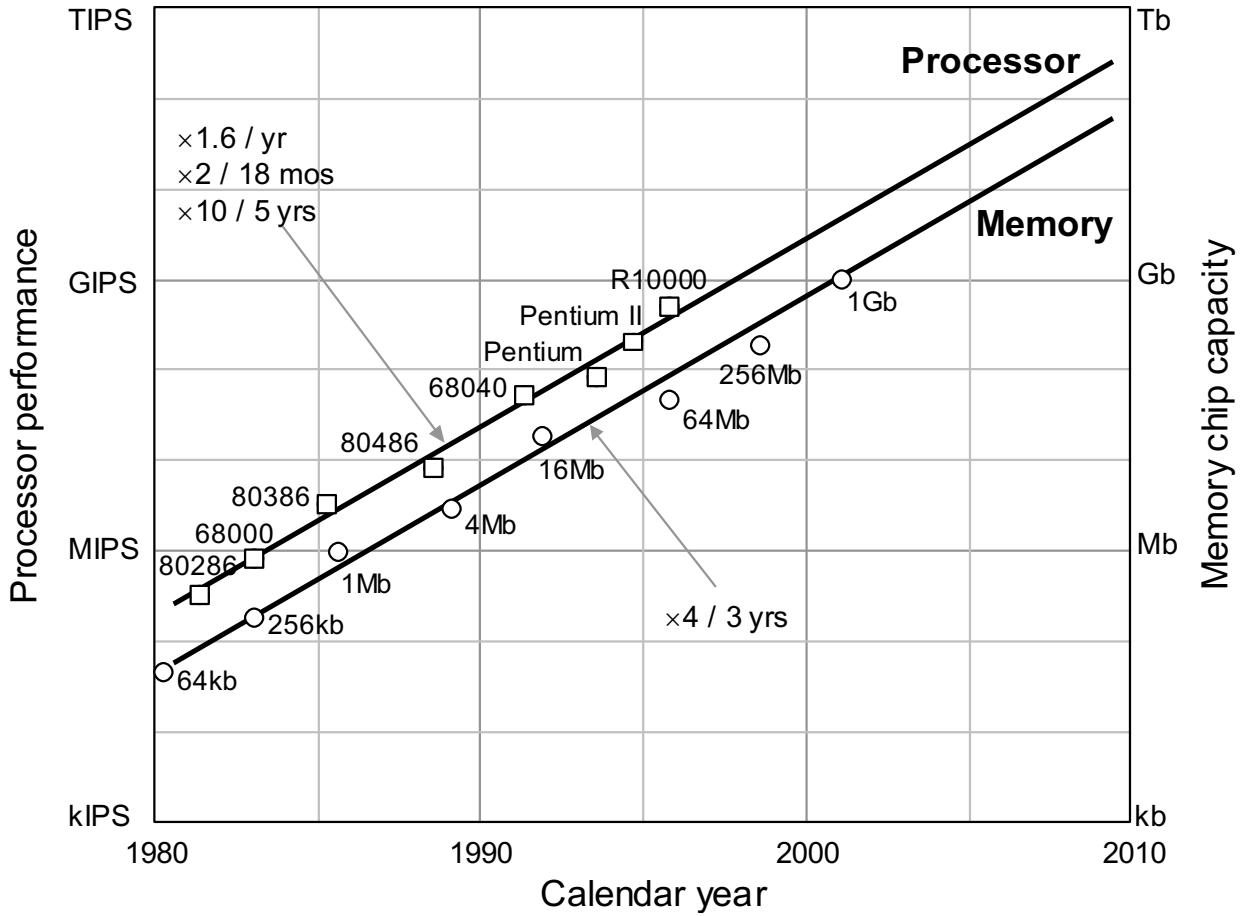


(a) 2D or 2.5D packaging now common

(b) 3D packaging of the future

Încapsularea procesoarelor, memoriilor și a altor componente.

Legea lui Moore



Creșterea performanței procesoarelor și memoriilor de-a lungul anilor.

”Rateuri” în previziunile despre viitorul tehnicii de calcul

“DOS addresses only 1 MB of RAM because we cannot imagine any applications needing more.” Microsoft, 1980

“640K ought to be enough for anybody.” Bill Gates, 1981

“Computers in the future may weigh no more than 1.5 tons.” *Popular Mechanics*

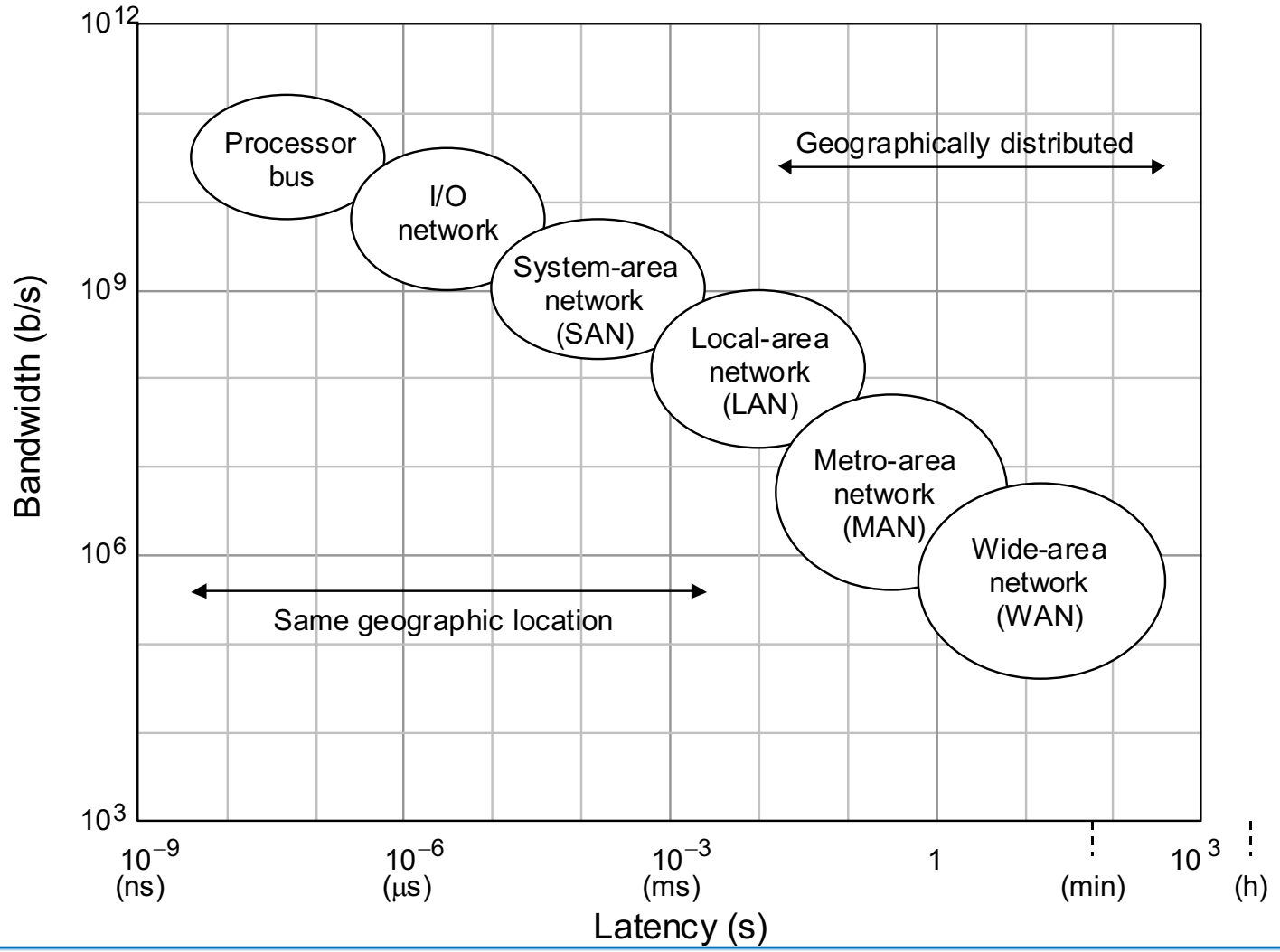
“I think there is a world market for maybe five computers.” Thomas Watson, IBM Chairman, 1943

“There is no reason anyone would want a computer in their home.” Ken Olsen, DEC founder, 1977

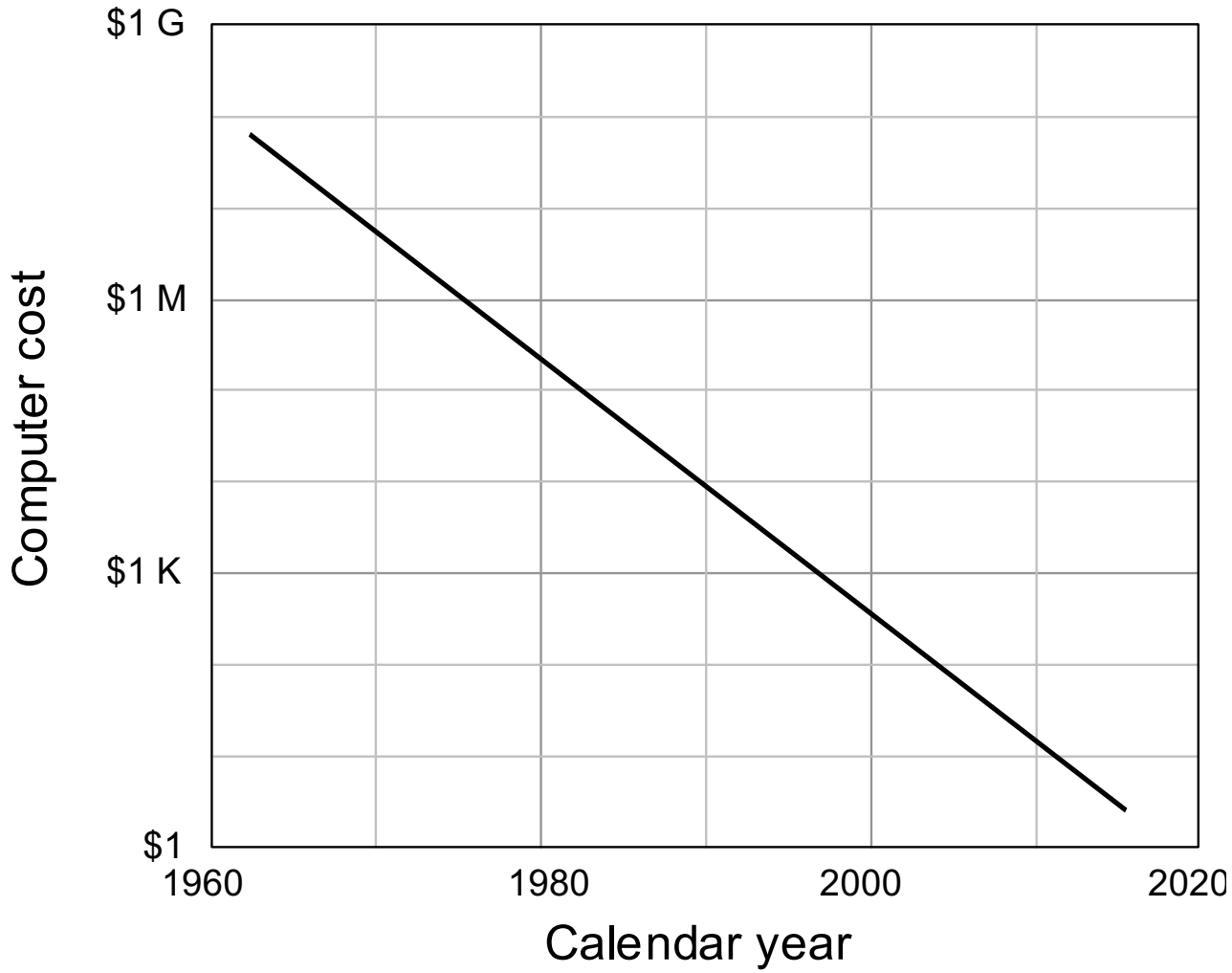
“The 32-bit machine would be an overkill for a personal computer.” Sol Libes, *ByteLines*

Tehnologii de comunicație

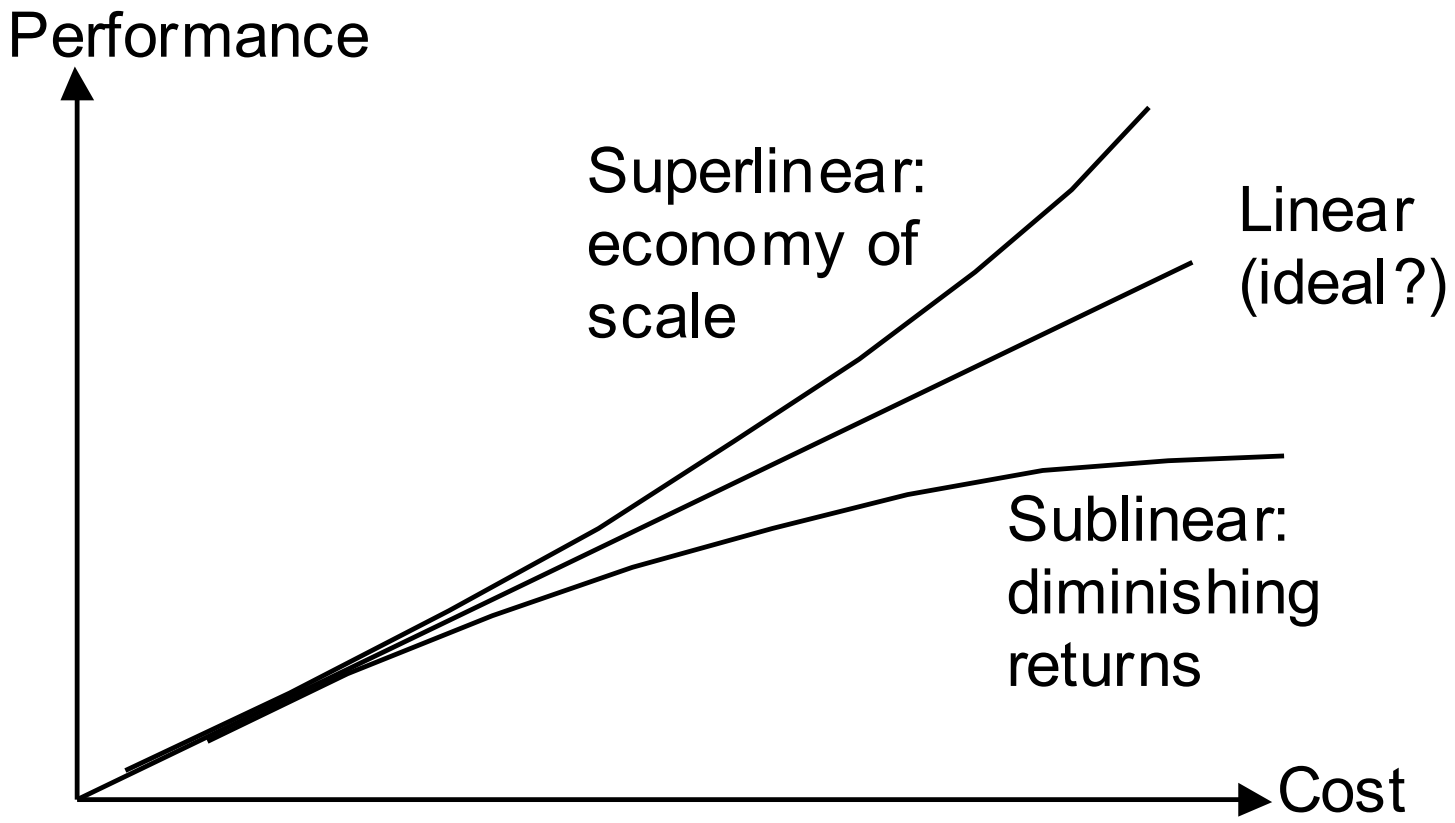
Caracteristicile de latență și lățime de bandă ale diferitelor clase de rețele de comunicație.



Cost, performanță și Cost/Performanță

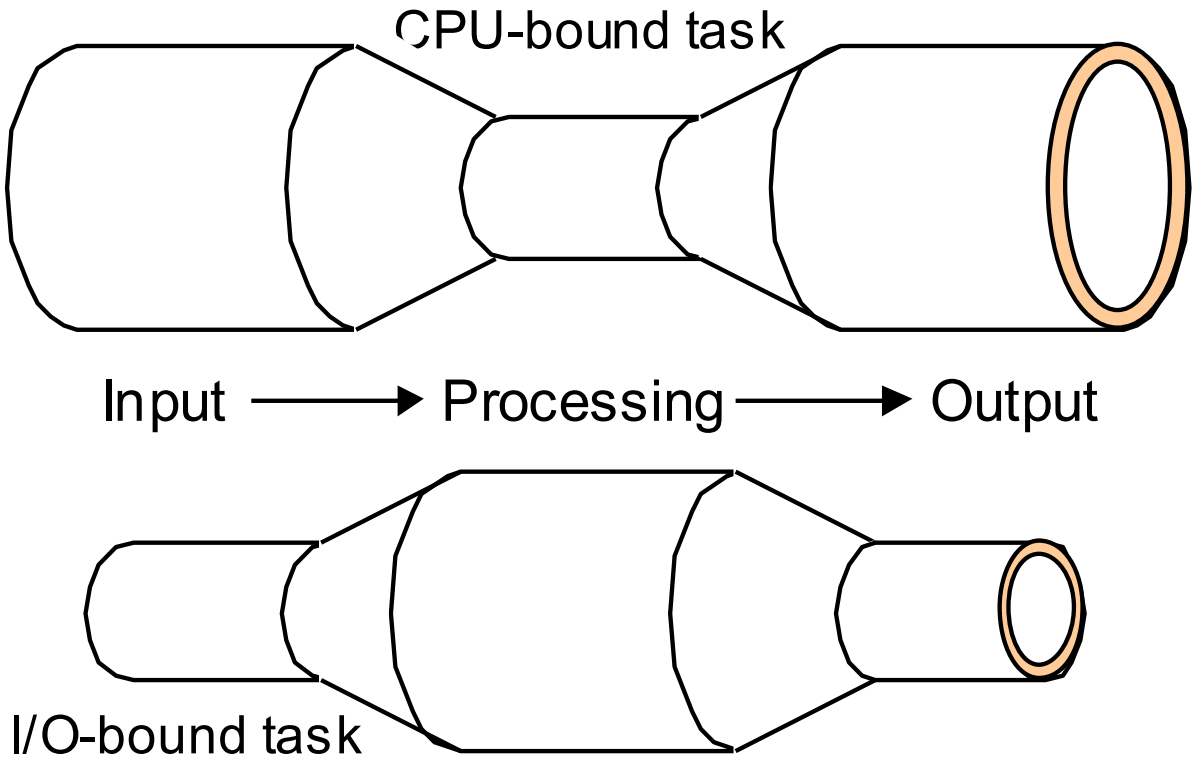


Cost/Performanță



Îmbunătățirea performanței în funcție de cost

Definirea performanței sistemelor de calcul



Analogia cu un pipeline arată că discrepanța dintre puterea de procesare și capabilitățile I/O duce la o constrângere (bottleneck) a performanței.

Analogie: comparăm șase avioane comerciale

Performanța avioanelor

Caracteristicile cheie pentru șase avioane de pasageri (cifrele sunt aproximative și sunt alese ca valori medii ale diferitelor configurații posibile);

Avion	Pasageri	Rază (km)	Viteză (km/h)	Preț ($M)
Airbus A310	250	8 300	895	120
Boeing 747	470	6 700	980	200
Boeing 767	250	12 300	885	120
Boeing 777	375	7 450	980	180
Concorde	130	6 400	2 200	350
DC-8-50	145	14 000	875	80

Diferite definiții ale performanței

Performanța din punctul de vedere al pasagerului: Viteza

Viteza este doar una din variabile. Timpul total al călătoriei este mai mare decât timpul de zbor. Dacă distanța depășește raza de acțiune a unui avion rapid, e de preferat alegerea unui avion mai lent, dar care nu are nevoie de escală pentru realimentare.

Performanța din pct. de vedere al companiei aeriene: Productivitate

Măsurată în pasageri-km/oră (relavantă dacă prețurile biletelor ar fi d.p. cu distanța parcursă, ceea ce nu este chiar adevărat)

Airbus A310	$250 \times 895 = 0.224$ M pasageri-km/hr
Boeing 747	$470 \times 980 = 0.461$ M pasageri-km/hr
Boeing 767	$250 \times 885 = 0.221$ M pasageri-km/hr
Boeing 777	$375 \times 980 = 0.368$ M pasageri-km/hr
Concorde	$130 \times 2200 = 0.286$ M pasageri-km/hr
DC-8-50	$145 \times 875 = 0.127$ M pasageri-km/hr

Performanța din pct. de vedere al FAA: Siguranța

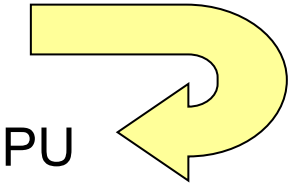
Eficiența: Cost/Performanță

Caracteristicile cheie pentru șase avioane de pasageri (cifrele sunt aproximative și sunt alese ca valori medii ale diferitelor configurații posibile);

Avion	Pasageri	Rază (km)	Viteză (km/h)	Preț ($M)	Productivitate (M P km/hr)	Cost / Performanță
					Valorile mari sunt ok	**Valorile mici sunt ok**
A310	250	8 300	895	120	**0.224**	**536**
B 747	470	6 700	980	200	**0.461**	**434**
B 767	250	12 300	885	120	**0.221**	**543**
B 777	375	7 450	980	180	**0.368**	**489**
Concorde	130	6 400	2 200	350	**0.286**	**1224**
DC-8-50	145	14 000	875	80	**0.127**	**630**

Concepte de performanță și speedup

Performanță = 1 / Timp execuție



se poate simplifica

Performanță = 1 / Timp execuție CPU

(Performanța lui M_1) / (Performanța lui M_2) = Speedup M_1 față de M_2
= (Timp execuție M_2) / (Timp execuție M_1)

Terminologie: M_1 is *x* times **as fast as** M_2 (e.g., 1.5 times as fast)
M_1 is 100(*x* – 1)% **faster than** M_2 (e.g., 50% faster)

CPU time = Instructions × (Cycles per instruction) × (Secs per cycle)
= Instructions × CPI / (Clock rate)

Instruction count, CPI, și clock rate nu sunt complet independente, așa că îmbunătățirea uneia de x ori poate duce la îmbunătățirea cu aproape x ori a timpului total de execuție

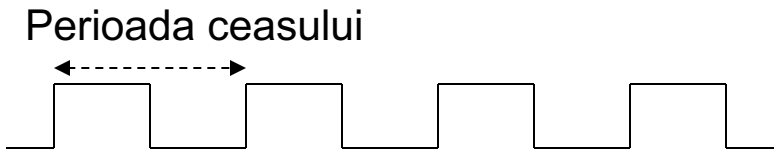
Dezvoltarea formulei pt timpul de execuție CPU

CPU time = Instructions × (Cycles per instruction) × (Secs per cycle)
= Instructions × Average CPI / (Clock rate)

Instrucțiuni: Numărul de instrucțiuni executate, != numărul de instrucțiuni dintr-un program (estimare dinamică)

Average CPI: Calculat în funcție de estimarea dinamică a numărului de instrucțiuni executate și de numărul de cicli de ceas necesari pentru execuția unei instrucțiuni

Clock rate: 1 GHz = 10^9 cicli / s (1 ciclu = 10^{-9} s = 1 ns)
200 MHz = 200 × 10^6 cicli / s (1 ciclu = 5 ns)



Dynamic Instruction Count

Cât de multe instrucțiuni sunt executate în acest fragment de cod?

```
250 instrucțiuni
for i = 1, 100 do
20 instrucțiuni
    for j = 1, 100 do
    40 instrucțiuni
        for k = 1, 100 do
        10 instrucțiuni
        endfor
    endfor
endfor
```

Static count = 326

Fiecare ”for” constă din 2 instrucțiuni: incrementarea indexului și verificarea condiției de iesire

12,422,450 Instrucțiuni

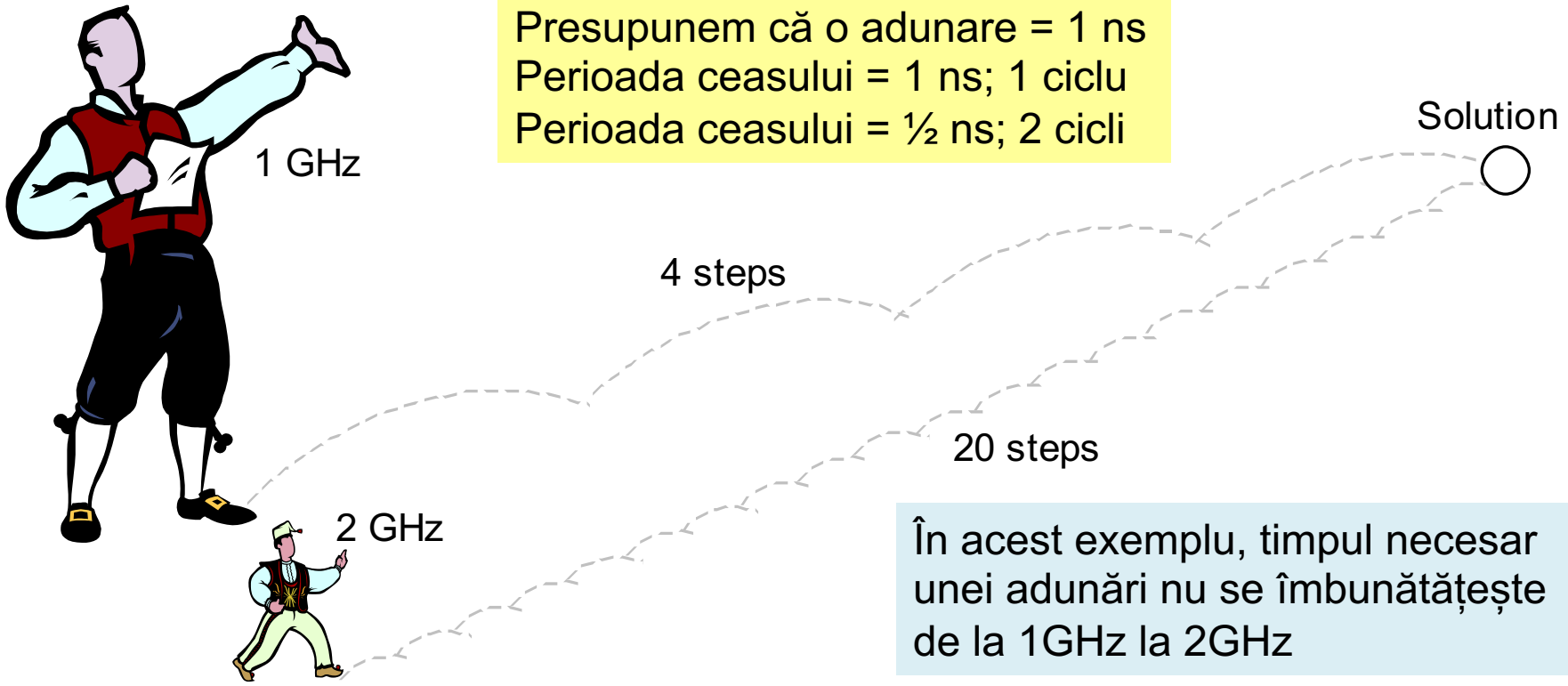
2 + 20 + 124,200 instrucțiuni
100 iterații
12,422,200 instrucțuni în total

2 + 40 + 1200 instrucțiuni
100 iterații
124,200 instrucțiuni în total

2 + 10 instrucțiuni
100 iterații
1200 instrucțiuni în total

```
for i = 1, n
while x > 0
```

Ceas mai rapid ≠ Timp de execuție mai mic

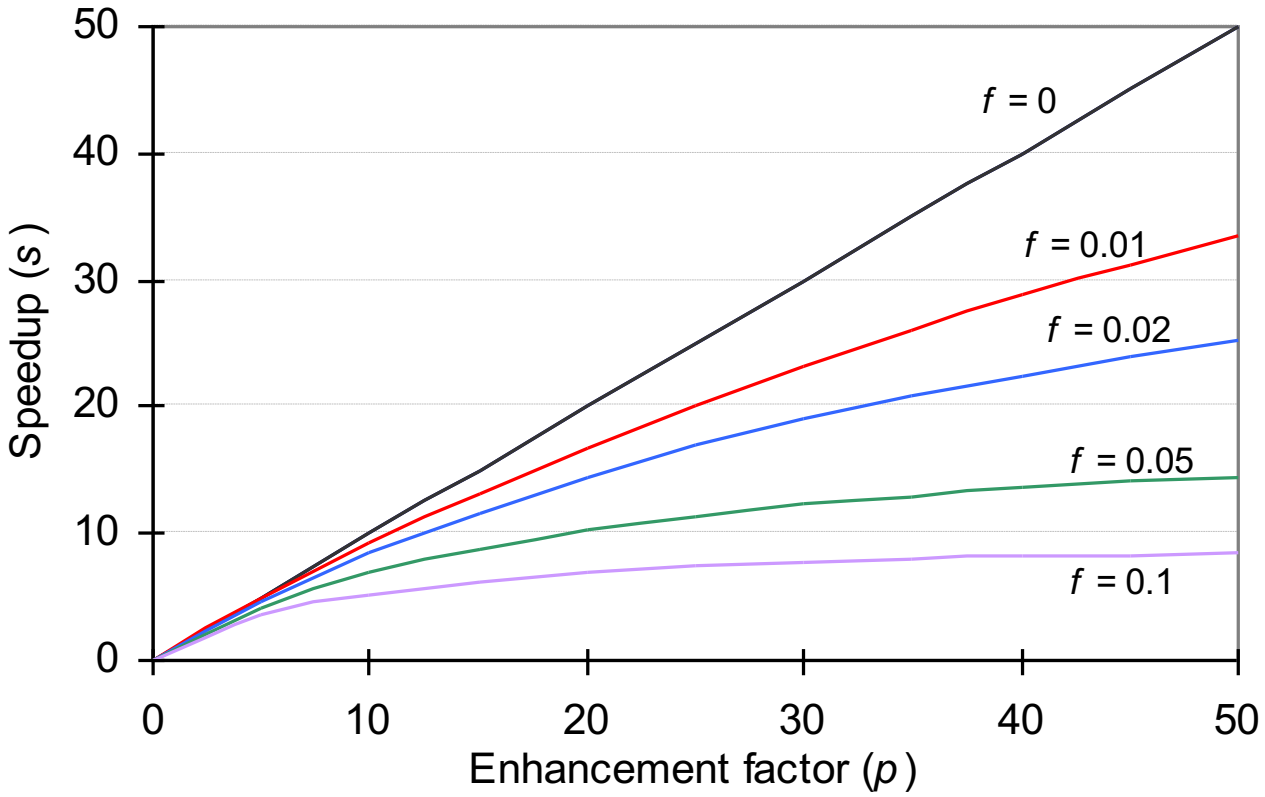


Presupunem că o adunare = 1 ns
Perioada ceasului = 1 ns; 1 ciclu
Perioada ceasului = ½ ns; 2 cicli

În acest exemplu, timpul necesar unei adunări nu se îmbunătățește de la 1GHz la 2GHz

Pași mai rapizi nu înseamnă neapărat că ajungi la destinație într-un timp mai scurt.

Creșterea performanței: Legea lui Amdahl



f = fracțiunea ne-paralelizabilă

p = speedup pentru restul

$$s = \frac{1}{f + (1-f)/p}$$

$$\leq min(p, 1/f)$$

Legea lui Amdahl: Îmbunătățirea vitezei de execuție dacă fracțiunea f a unui task este ne-paralelizabilă și restul de 1-f din task rulează de p ori mai repede.

Legea lui Amdahl în proiectare

Un procesor petrece 30% din timp cu adunări flp, 25% cu înmulțiri flp și 10% cu împărțiri flp. Evaluați îmbunătățirea performanței pentru:

a. Sumatorul flp este de 2x mai rapid
b. Multiplicatorul flp este de 3x mai rapid.
c. Unitatea de împărțire flp este de 10x mai rapidă

Soluție

a.Speedup adder = 1 / [0.7 + 0.3 / 2] = 1.18
b.Speedup multiplier = 1 / [0.75 + 0.25 / 3] = 1.20
c.Speedup divider = 1 / [0.9 + 0.1 / 10] = 1.10

Dar dacă și sumatorul și multiplicatorul sunt reproiectate simultan?

Legea lui Amdahl în management

Membrii unui grup de cercetare vizitează frecvent biblioteca. Fiecare vizită durează 20 de minute. Grupul decide să se aboneze la o suită de publicații care acoperă 90% din vizitele la bibliotecă; timpul de acces la publicații este redus la 2 minute.

a. Cu cât s-a îmbunătățit timpul mediu de acces la publicații?
b. Dacă grupul are 20 de membri, fiecare făcând 2 vizite/săpt. La bibliotecă, care este cheltuiala maxim admisă pt. abonamente? Presupuneți că sunt 50 săpt. de muncă pe an și un cercetător e plătit cu 25$/h.

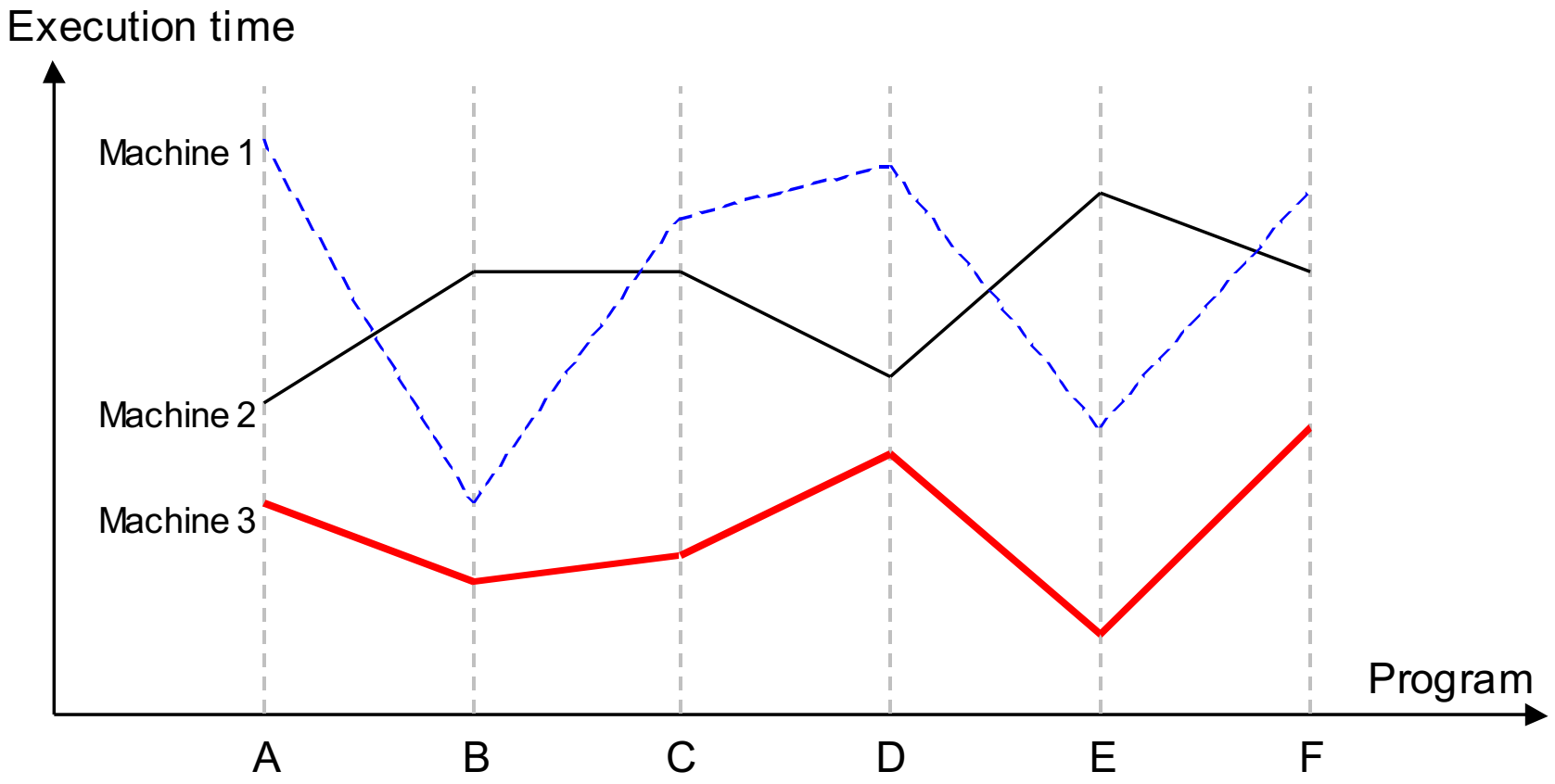
Soluție

a.Speedup în accesul la publicații = 1 / [0.1 + 0.9 / 10] = 5.26

b.Timp economisit = $20 \times 2 \times 50 \times 0.9$ (20 – 2) = 32,400 min = 540 h

Costuri = 540 $\times$ \$25 = \$13,500 = Cheltuiala maxim admisă

Măsurarea vs. Modelarea performanței



Timpii de execuție a șase programe pe trei mașini diferite

Legea lui Amdahl generalizată

Timpul inițial de rulare pt un program = 1 = $f_1 + f_2 + \ldots + f_k$

Noul timp de rulare, după ce fracțiunea f_i este rulată de p_i ori mai repede

$$\frac{f_1}{p_1} + \frac{f_2}{p_2} + \ldots + \frac{f_k}{p_k}$$

Formula pentru speedup

$$S = \frac{1}{\frac{f_1}{p_1} + \frac{f_2}{p_2} + \ldots + \frac{f_k}{p_k}}$$

Dacă o fracțiune este încetinită, se folosește $s_j\, f_j$ în locul f_j / p_j, unde $s_j > 1$ este factorul de încetinire (slowdown)

Teste de performanță (benchmarking)

Ești un inginer la Outtel, companie start-up care dorește să concureze cu Intel folosind noile sale procesoare care sunt mai bune de 2.5x decât Intel pt. operații floating-point. Acest nivel de performanță a putut fi atins printr-un compromis de design care a dus la creșterea cu 20% a timpului de execuție pentru toate celelalte instrțiuni. Job-ul tău este de a selecta benchmark-ul care să arate superioritatea procesoarelor Outtel.

a. Care este fracția minimă de timp *f* de operații fpu pt un program care ruleaza pe un procesor Intel pt ca Outtel sa aibă un speedup de 2x sau mai bun?

Soluție

a.Folosim o formă generalizată a legii lui Amdahl pt care *f* are un speedup de 2.5 și restul un slowdown de 1.2 (100%+20%): $1 / [1.2(1 - f) + f / 2.5] \geq 2 \Rightarrow f \geq 0.875$

Estimarea performanței

Average CPI = $\sum_{\text{All instruction classes}}$ (Class-*i* fraction) × (Class-*i* CPI)

Machine cycle time = 1 / Clock rate

CPU execution time = Instructions × (Average CPI) / (Clock rate)

Frecvența folosirii, în procente, pt. diferitele tipuri de instrucțiuni în patru aplicații reprezentative

Aplicație → Clasa instr. ↓	Compresie de date	Compilator C	Simulator reactor	Modelarea mișcării atomilor
A: Load/Store	25	37	32	37
B: Integer	32	28	17	5
C: Shift/Logic	16	13	2	1
D: Float	0	0	34	42
E: Branch	19	13	9	10
F: All others	8	9	6	4

Calcule pentru CPI și IPS

Considerăm două implementări M_1 (600 MHz) și M_2 (500 MHz) al unui set de instrucțiuni ce conține următoarele clase:

Clasă	CPI pt. M_1	CPI pt. M_2	Comentarii
F	5.0	4.0	Floating-point
I	2.0	3.8	Integer arithmetic
N	2.4	2.0	Nonarithmetic

a. Care este performanța maximă pt. M_1 și M_2 în MIPS?
b. Dacă 50% din instrucțiuni sunt din clasa N, și restul împărțite egal între F și I, care mașină e mai rapidă? De câte ori?

Soluție

a. MIPS max. pt. M_1 = 600 / 2.0 = 300; pt. M_2 = 500 / 2.0 = 250

b. CPI mediu pt. M_1 = 5.0 / 4 + 2.0 / 4 + 2.4 / 2 = 2.95;
pt. M_2 = 4.0/4 + 3.8/4 + 2.0/2 = 2.95 $\rightarrow$ M_1 mai rapid de 1.2x

Rating-ul MIPS poate să inducă în eroare

Două compilatoare produc cod mașină pentru același program pe o mașină cu două clase de instrucțiuni:

Clasa	CPI	Compilator 1	Compilator 2
A	1	600M	400M
B	2	400M	400M

a. Care sunt timpii de rulare pentru cele 2 programe la un ceas de 1 GHz?
b. Care compilator produce cod mai rapid și de câte ori mai rapid?
c. Care program rulează la o rată MIPS mai mare?

Soluție

a. Timp rulare1 (2) = (600M × 1 + 400M × 2) / 10^9 = 1.4 s (1.2 s)
b. Codul generat de C2 e de 1.4 / 1.2 = 1.17x mai rapid
c. MIPS rating 1, CPI = 1.4 (2, CPI = 1.5) = 1000 / 1.4 = 714 (667)

Raportarea performanței

Timpul de execuție măsurat sau estimat pt. trei programe.

	Timp pe mașina X	Timp pe mașina Y	Speedup Y față de X
Program A	20	200	0.1
Program B	1000	100	10.0
Program C	1500	150	10.0
Toate trei	2520	450	5.6

Analogie: Dacă o mașină merge către un oraș aflat la 100km cu 100km/h și se întoarce cu 50km/h, viteza medie nu este (100+50)/2 ci este obținută luând în considerare că a parcurs 200km în 3 ore.

Compararea performanței totale

Timpul măsurat și estimat pentru execuția a trei programe

	Timp pe mașina X	Timp pe mașina Y	Speedup Y față de X	**Speedup X față de Y**
Program A	20	200	0.1	10
Program B	1000	100	10.0	0.1
Program C	1500	150	10.0	0.1
	Medie aritmetică		6.7	3.4
	Medie geometrică		2.15	0.46

Media geometrică nu produce o metrică eficientă a speedup-ului total, dar este un indicator că lucrurile merg în direcția dorită

Efectele amestecării instrucțiunilor asupra performanței

Luăm un exemplu de două aplicații DC și RS și două mașini M_1 și M_2:

Clasă	Data Comp.	Reactor Sim.	M_1's CPI	M_2's CPI
A: Ld/Str	25%	32%	4.0	3.8
B: Integer	32%	17%	1.5	2.5
C: Sh/Logic	16%	2%	1.2	1.2
D: Float	0%	34%	6.0	2.6
E: Branch	19%	9%	2.5	2.2
F: Other	8%	6%	2.0	2.3

a. Aflați CPI efectiv pt. cele două aplicații pe ambele mașini.

Soluție

a.CPI al DC pe M_1: $0.25 \times 4.0 + 0.32 \times 1.5 + 0.16 \times 1.2 + 0 \times 6.0 + 0.19 \times 2.5 + 0.08 \times 2.0 = 2.31$

DC pe M_2: 2.54 RS pe M_1: 3.94 RS pe M_2: 2.89

În căutarea performanței mărite

Puterea de calcul disponibilă la începutul anilor 2020:

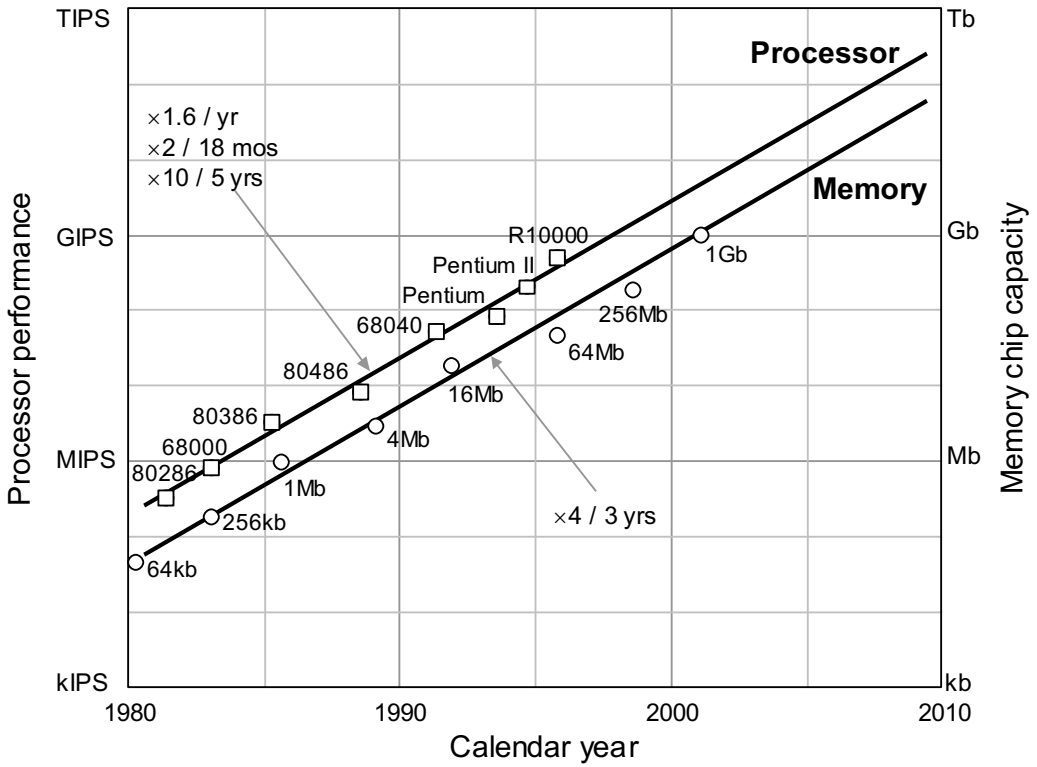
Gigaflops pe un calculator desktop
Sute de Petaflops pe un supercomputer
Exaflops în proiectare

Prefixuri pentru numere mari:
Kilo = 10^3, Mega = 10^6, Giga = 10^9, Tera = 10^{12}, Peta = 10^{15}

Pentru memorie:
K = 2^{10} = 1024, M = 2^{20}, G = 2^{30}, T = 2^{40}, P = 2^{50}

Prefixuri pentru numere mici:
micro = 10^{-6}, nano = 10^{-9}, pico = 10^{-12}, femto = 10^{-15}

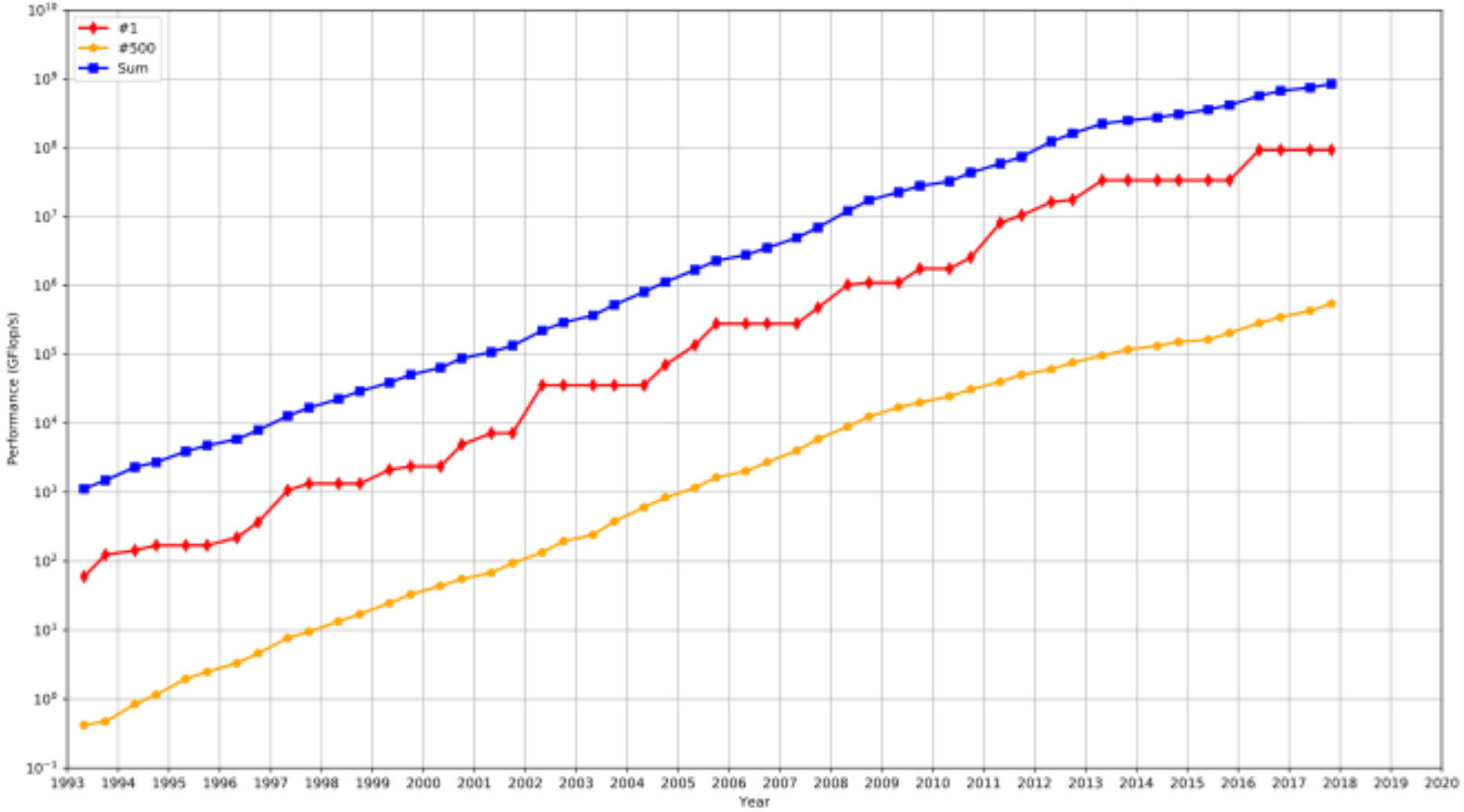
Trenduri în performanță și învechire



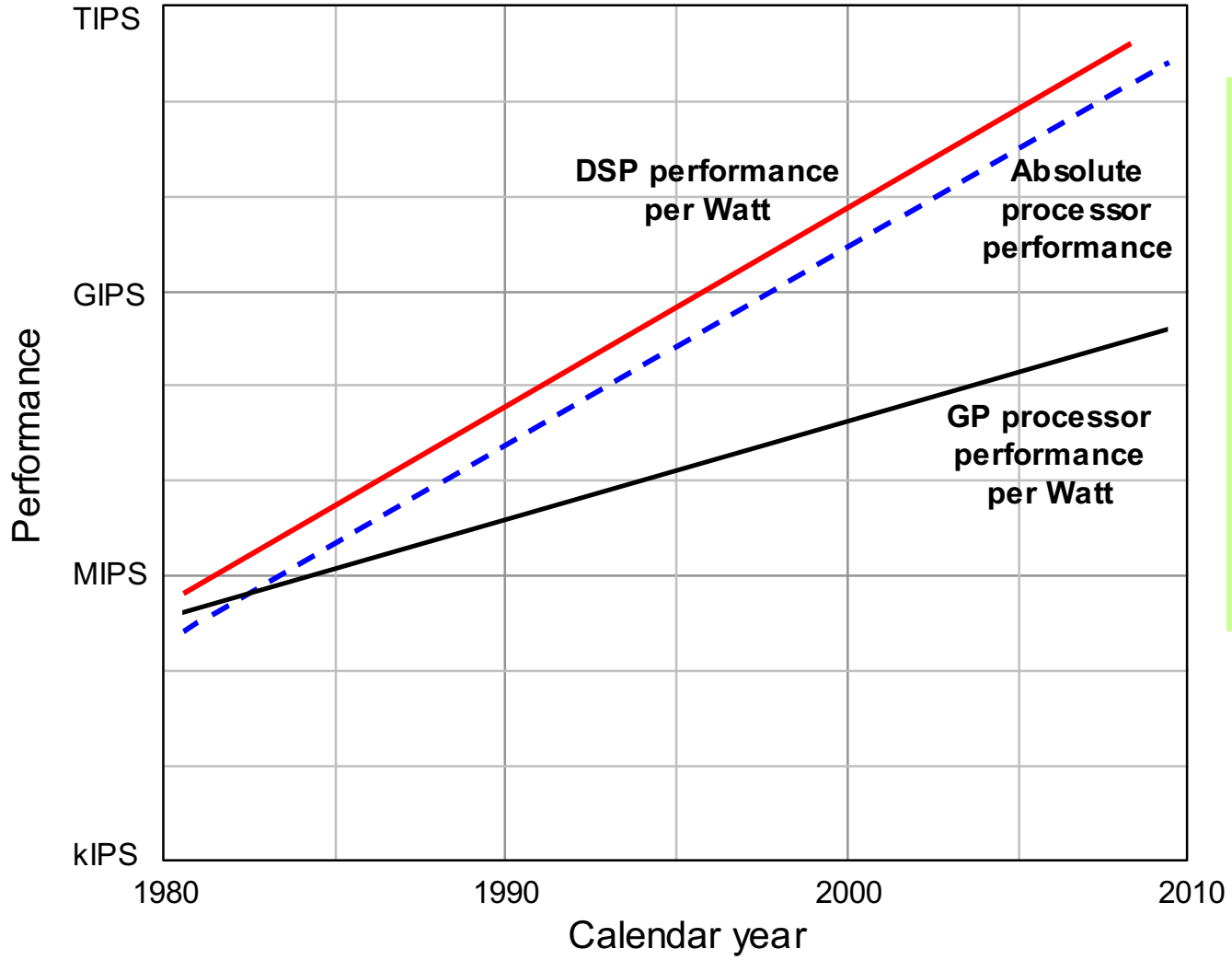
Legea lui Moore pentru procesoare și DRAM

"Can I call you back? We just bought a new computer and we're trying to set it up before it's obsolete."

Super-computere



Performanța e importantă, dar nu înseamnă totul!



Tendința în putere de calcul/Watt pentru calculatoare general-purpose și procesoare DSP.

Acknowledgements

- Aceste slide-uri conțin materiale aparținând:
  - Arvind (MIT)
  - Krste Asanovic (MIT/UCB)
  - Joel Emer (Intel/MIT)
  - James Hoe (CMU)
  - John Kubiatowicz (UCB)
  - David Patterson (UCB)
  - Behrooz Parhami (UCSB)

- MIT material derived from course 6.823
- UCB material derived from course CS252