Escuela de Ingeniería y Arquitectura
Universidad Zaragoza

Departamento de Informática e Ingeniería de Sistemas
Universidad Zaragoza

Tema 3
Rendimiento y modelos de ejecución

Arquitectura y Organización de Computadores 2

2º curso, Grado en Ingeniería Informática

Arquitectura y Tecnología de Computadores
Departamento de Informática e Ingeniería de Sistemas

Guión del tema

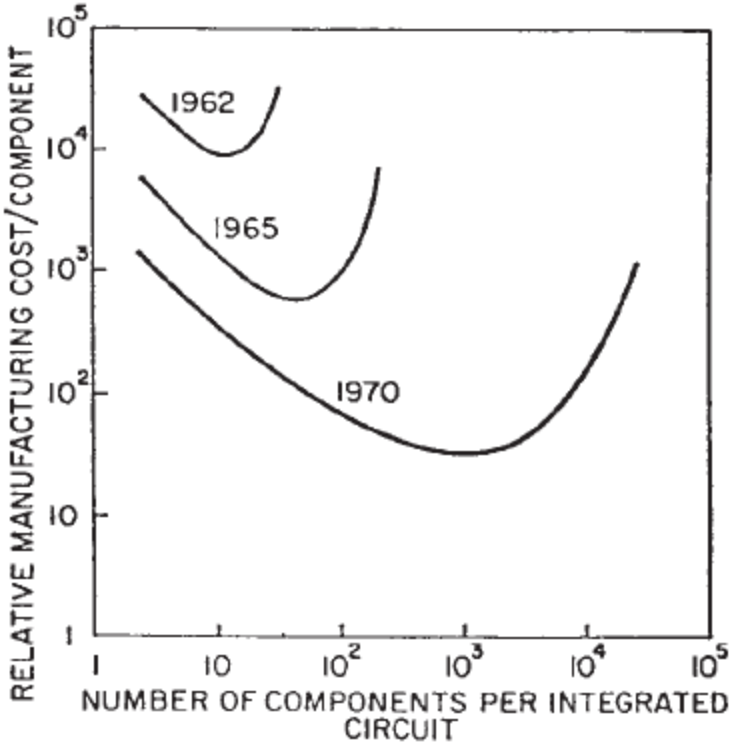
- Tendencias
- Medida del rendimiento
- Incremento del rendimiento por diseño: evolución de los modelos de ejecución

Tendencias en uso y tecnologías del computador

demanda (software) $\leftrightarrow$ oferta (tecnología)

- Demanda
  - Tamaño de los programas en ejecución
  - Número de instrucciones a ejecutar
- Oferta: ley de Gordon E. Moore (co-fundador de Intel)
  - Procesador y memoria cache
  - Memoria DRAM
  - Disco Magnético

Ley de Moore


RELATIVE MANUFACTURING COST/COMPONENT
10^5
10^4
10^3
10^2
10
1
1962
1965
1970
1
10
10^2
10^3
10^4
10^5
NUMBER OF COMPONENTS PER INTEGRATED CIRCUIT

Pasa el tiempo, los programas crecen en tamaño

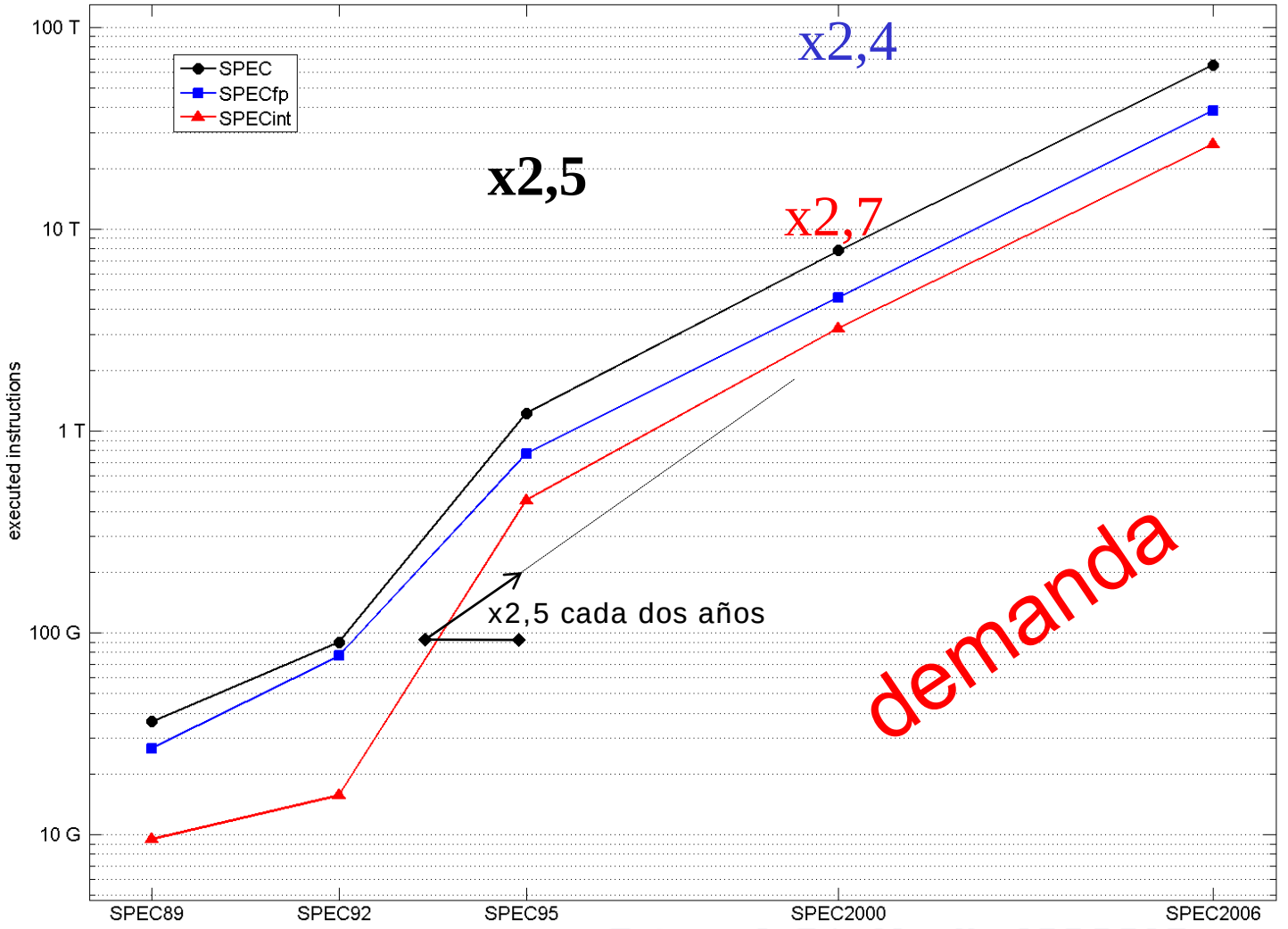
- Tamaño procesos (inst. + datos)
  - x1,5 - 2 cada año
  - 0,5 - 1 bit *más* de direccionamiento cada año
- Por qué?
  - Incremento de funcionalidad y/o precisión
- Primera ley del Software
  - Nathan Myhrvold (top-executive de Microsoft)

demanda

El software es un gas.
Se expande hasta ocupar todo el recipiente que lo contiene.

veamos ...

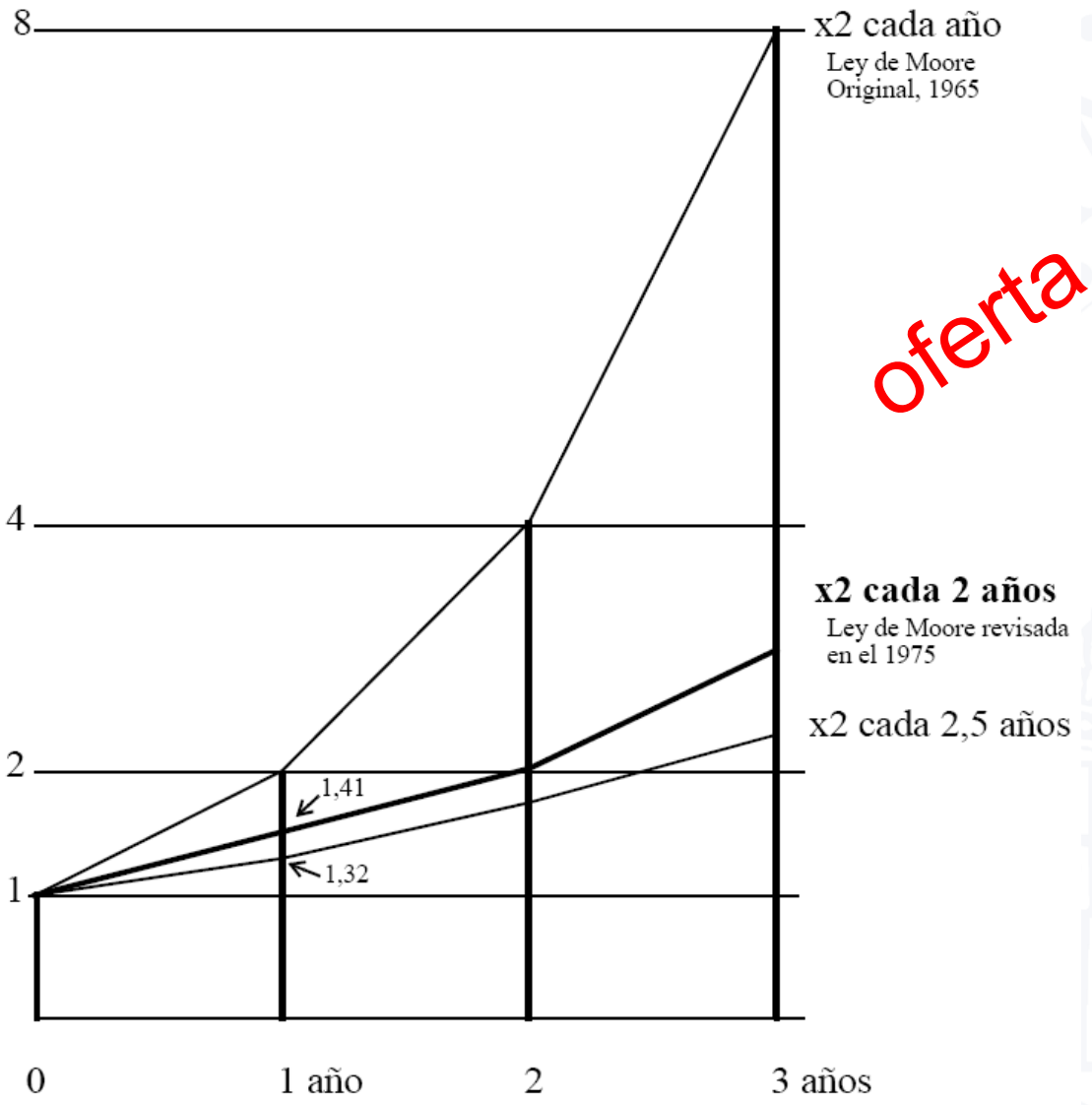
Pasa el tiempo, los programas ejecutan más instrucciones



- Se diseña la arquitectura pensando en el compilador o en el intérprete
- Vectorización y paralelización automática de programas secuenciales (C, C++, Fortran, …)
  para ejecutar en los micros actuales (CMPs)

Actualización de: J. Alastruey, J.L. Briz, P. Ibáñez y V. Viñals
"Software Demand vs. Hardware Supply: SPEC CPU and processor resources"
IEEE Micro, IEEE Computer Society 2006

Ley de Moore: nº de transistores/chip vs. tiempo (1/4)

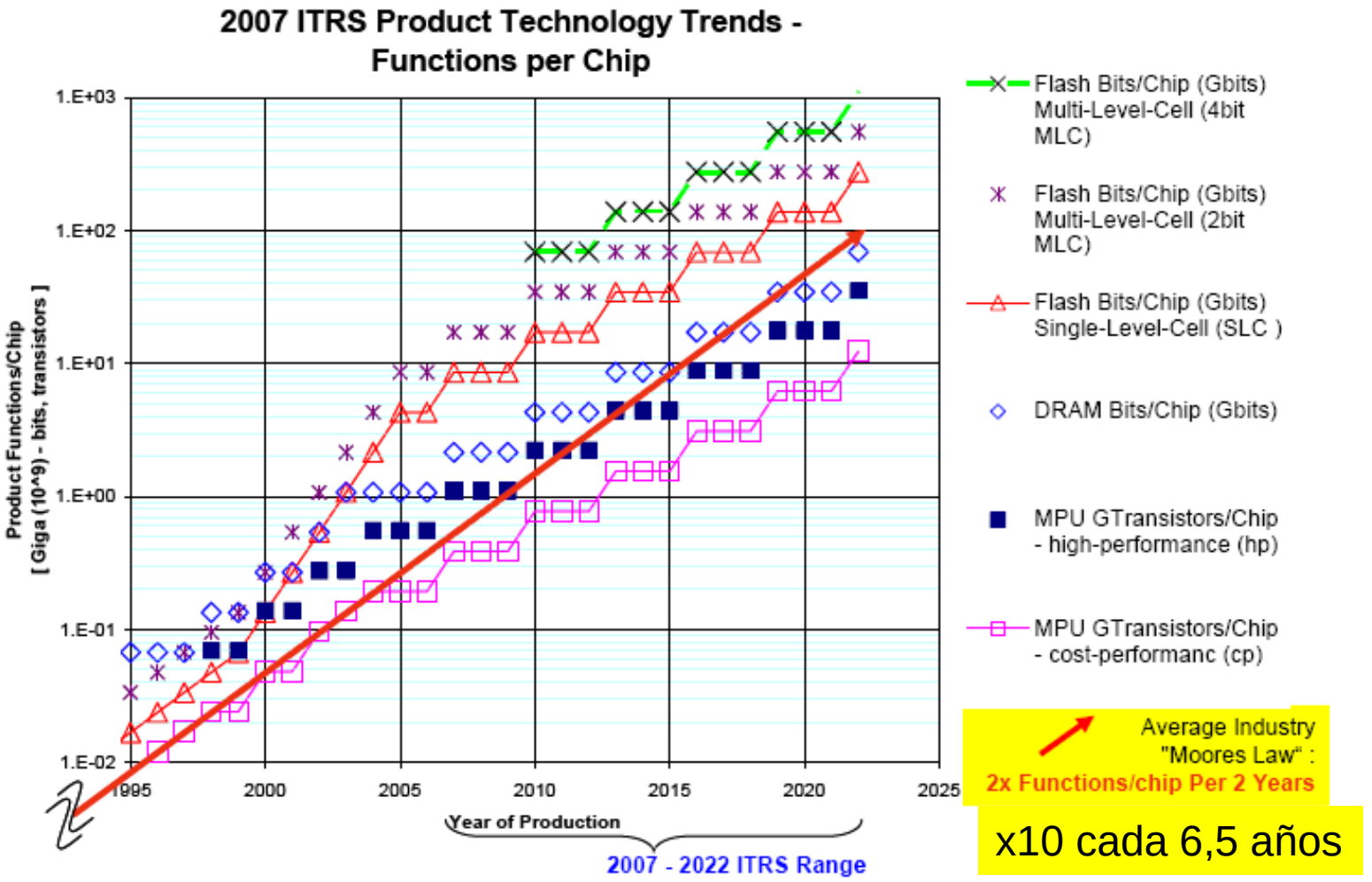


Ejemplo: familia Intel

procesador	introd.	#Trans.
4004	Abr71	2,3K
8008	Abr72	3,5K
8080	Abr74	6K
8086	Jun78	29K
8088	Jun79	29K
80286	Feb82	134K
80386	Oct85	275K
80486	Abr89	1.2M
Pentium	Mar93	3.1M
Pentium Pro	Mar 95	5.5M
Pentium II	May 97	7,5M
¿ Ley de Moore ?		

[1] L. Geppert, "The Media Event: Moore's Law Mania", *IEEE Spectrum*, Enero 1998, pp. 20:21.

Ley de Moore con datos y predicciones de consenso (2/4)



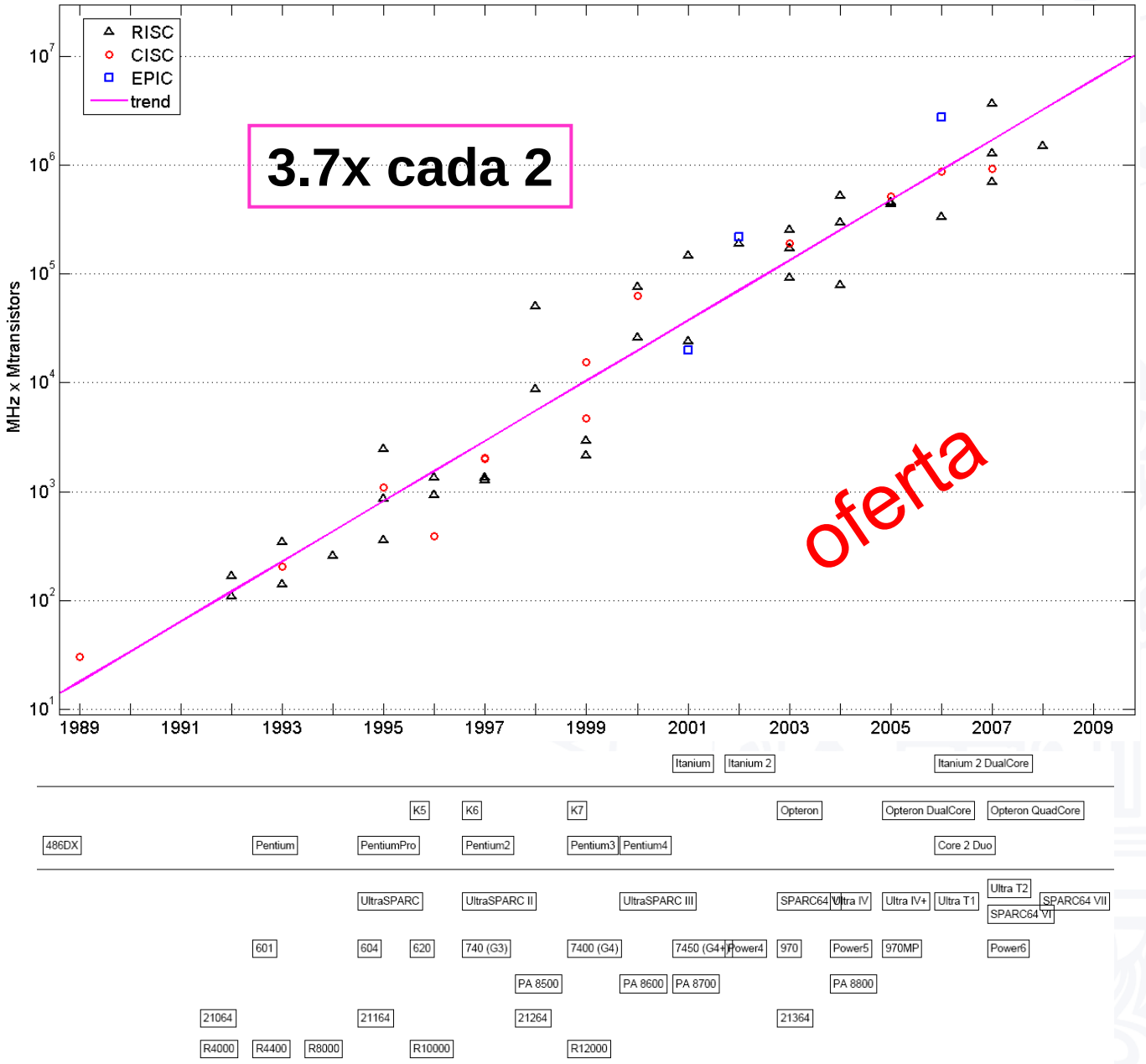
oferta

Semiconductor **I**nternational **A**ssociation:

International **T**echology **R**oadmap for **S**emiconductors. 2008 Update.

Figure ORTC2 ITRS Product Function Size Trends: MPU Logic Gate Size (4-transistor); Memory Cell Size [SRAM (6-transistor); Flash (SLC and MLC), and DRAM (transistor + capacitor)]--Updated

Ley de More y aumento de frecuencia: Transfreq (3/4)



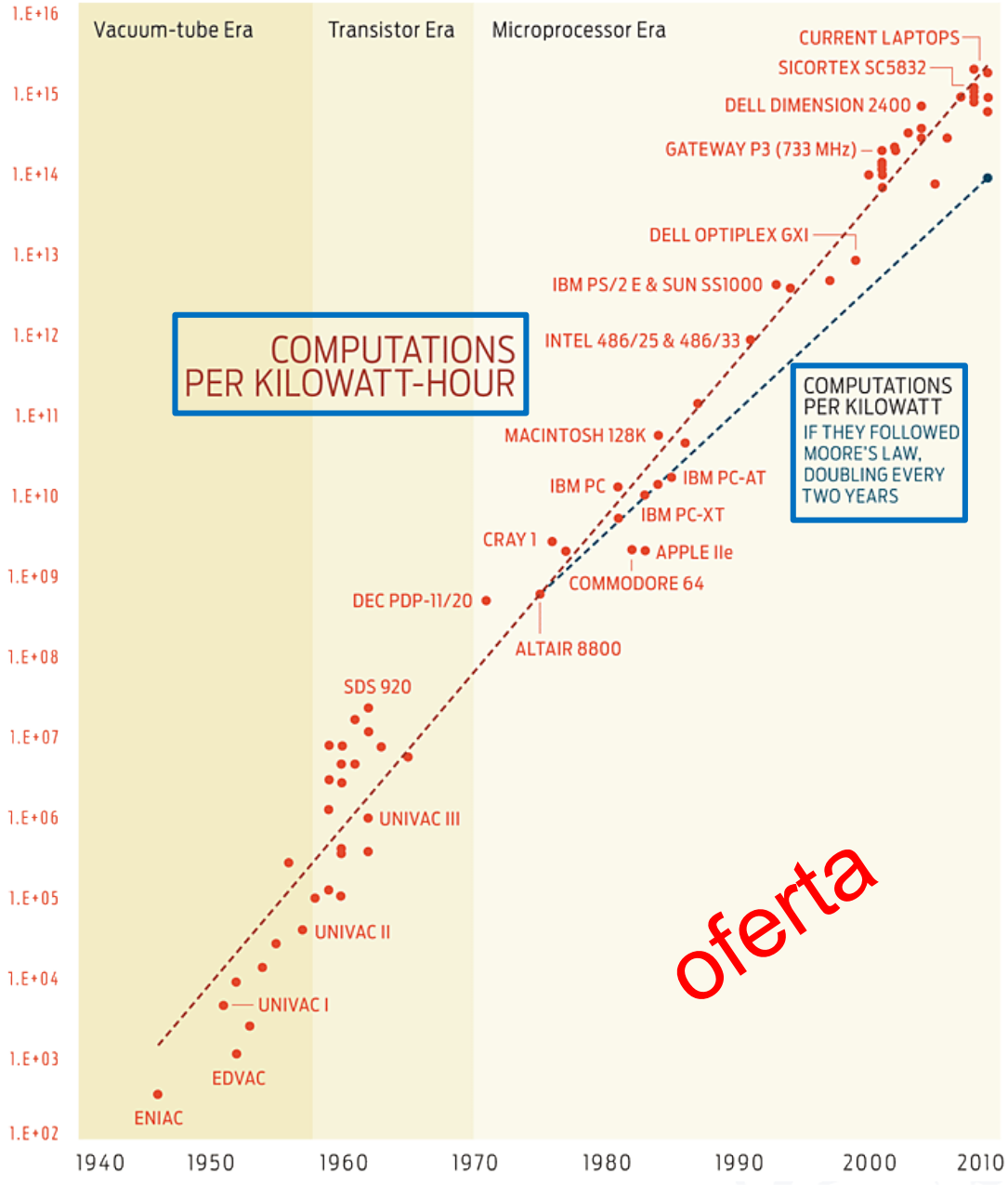
Transfreq =

transistor count
x
clock frequency

Tiene sentido este producto ?
Mtransistores x Mhz

Actualización de:
J. Alastruey, J.L. Briz, P. Ibáñez y V. Viñals. “Software Demand vs. Hardware Supply: SPEC CPU and processor resources”. *IEEE Micro*, IEEE Computer Society 2006.

Ley de Moore y eficiencia energética (4/4)



Outperforming Moore's Law

The number of transistors on a chip has doubled about every two years for decades, a trend that is popularly (but often imprecisely) encapsulated as Moore's Law. What is less well known is that the electrical efficiency in delivering computing performance began following a similar trend long before the microprocessor was invented.

The performance of desktop computers has doubled every 1.5 years, on average, since 1975, and in that time the number of computations per kilowatt-hour has grown about as quickly. But that measure of efficiency had also grown about that fast during the previous three decades, and even more rapidly during the vacuum-tube computing era and during the transition from tubes to transistors.

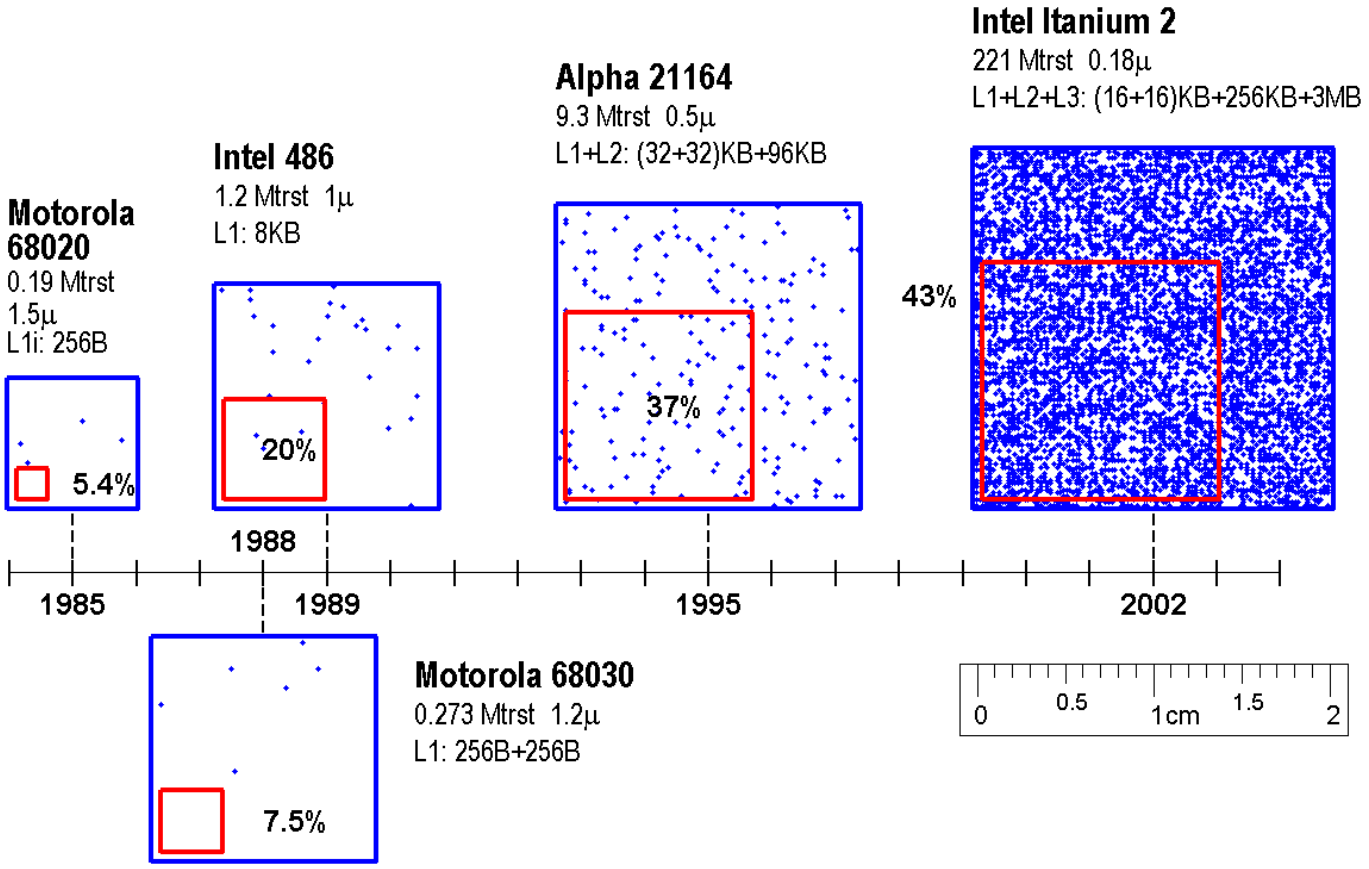
These are just a few of the conclusions drawn from a new study, "Assessing Trends in the Electrical Efficiency of Computation Over Time," prepared by me and my colleagues, researchers affiliated with Intel Corp., Lawrence Berkeley National Laboratory, Microsoft Corp., and Stanford University.

The main technology trends that have improved performance and reduced costs—at first better tubes, and then smaller transistors—also reduce power use, hence the similar improvements in computational performance and electrical efficiency, at similar rates, for such a long time. If these trends continue—and we have every reason to believe they will for at least the next 5 to 10 years—we can expect continued rapid reductions in the size and power requirements of computer-based devices. That should be especially welcome news for users of netbooks, smartphones, cameras, and other mobile devices.

Jonathan G. Koomey, IEEE Spectrum March, 2010, p. 56

Procesador y Memorias Cache

- Integración de **lógica y memoria rápida** (registros, SRAM para Mc)
  - nº transistores/chip: x2 cada 2-3 años (muy al principio, x8 cada 3 !)
    - densidad transistores: a este ritmo
    - superficie chip: estabilizada
    - velocidad conmutación: ≈ a este ritmo
    - retardo "RC" en metal: ≈ fijo



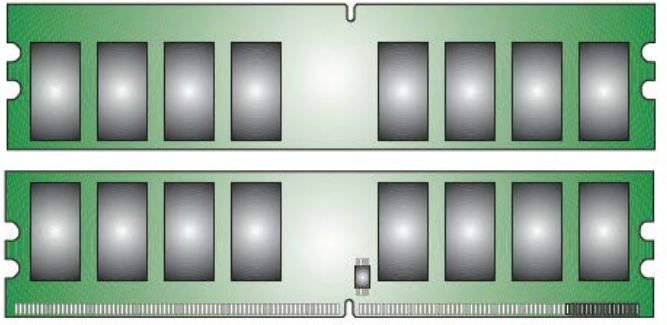
oferta

J. Alastruey, J.L. Briz, P. Ibáñez y V. Viñals "Software Demand vs. Hardware Supply: SPEC CPU and processor resources". *IEEE Micro*, IEEE Computer Society 2006

Memoria DRAM: capacidad y coste

- nº bits/chip
  - ..-2004: x2 cada 1.5 años
  - 2000-04: x2 cada 2 años
  - 2004-22: x2 cada 2-3 años
  - Último hito: 4 gigabits en 2009 (Samsung, 1Q09)
    - Jun-2011: sample shipments of 8Gb DDR3 Chip by Elpida. Based on 3D stacking (TSV, Through Silicon Via Technology)
- Coste/chip
  - No cambia demasiado
  - Feb-2009
    - Módulo de 2 GB DDR2-800
      - 16 chips x 1 gigabit: ~ 20 €
    - Módulo de 4 GB DDR2-800
      - 16 chips x 2 gigabit: ~ 200 €

oferta

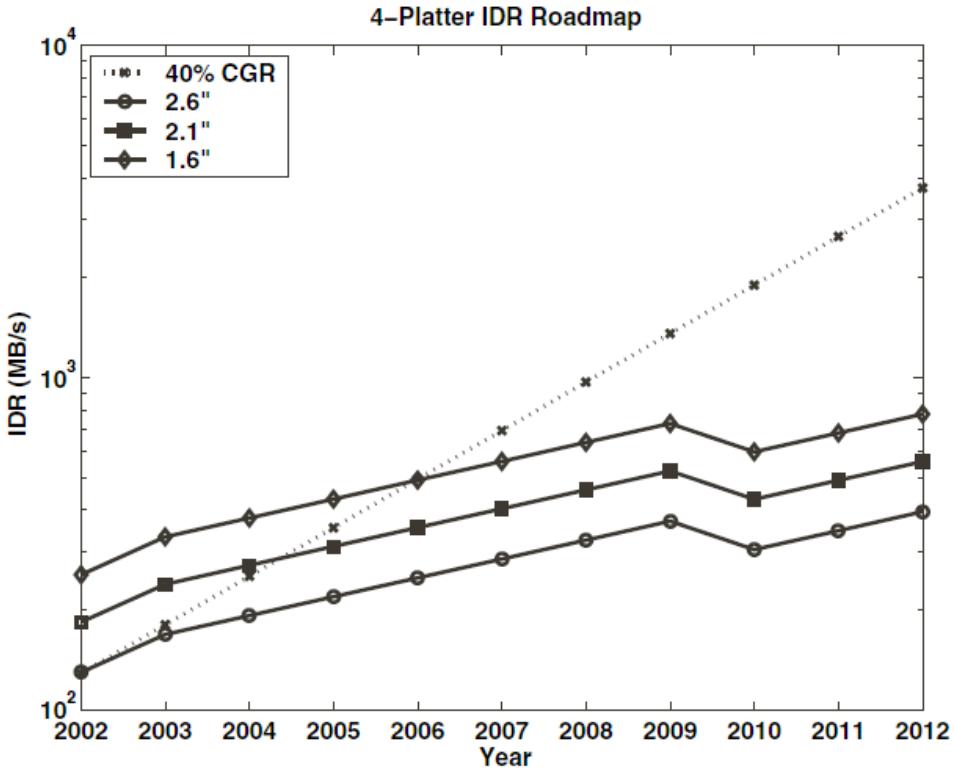


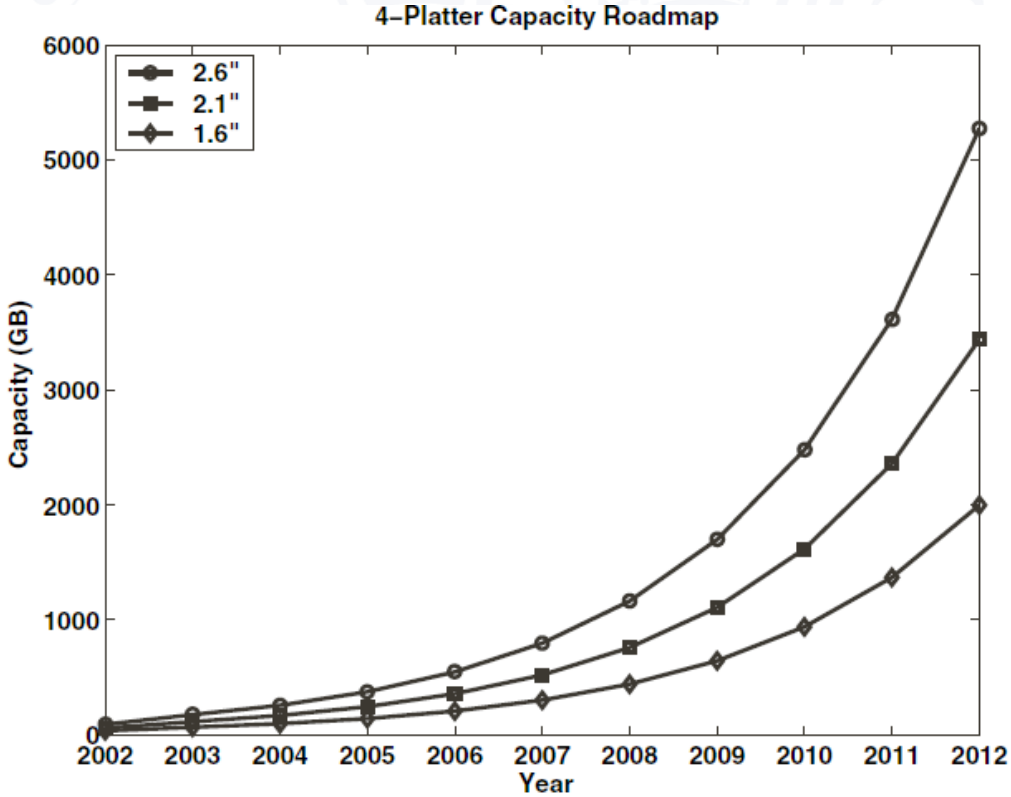
Disco magnético

- nº bits/mm2
  - x1,25 hasta el 90
  - x1,5 ahora
- tiempo de acceso
  - x0,94 cada año

oferta

Gurumurthi et al. “Disk Drive Roadmap from the Thermal Perspective: A Case for Dynamic Thermal Management”. *Int. Symp. on Computer Architecture* (ISCA), 2005



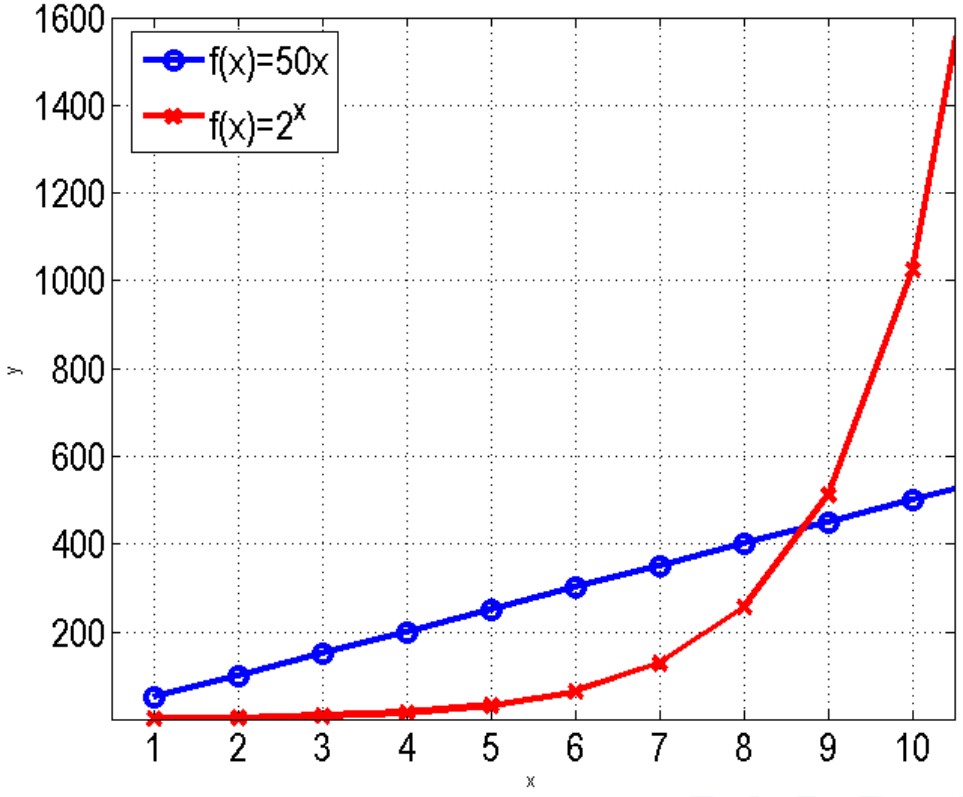


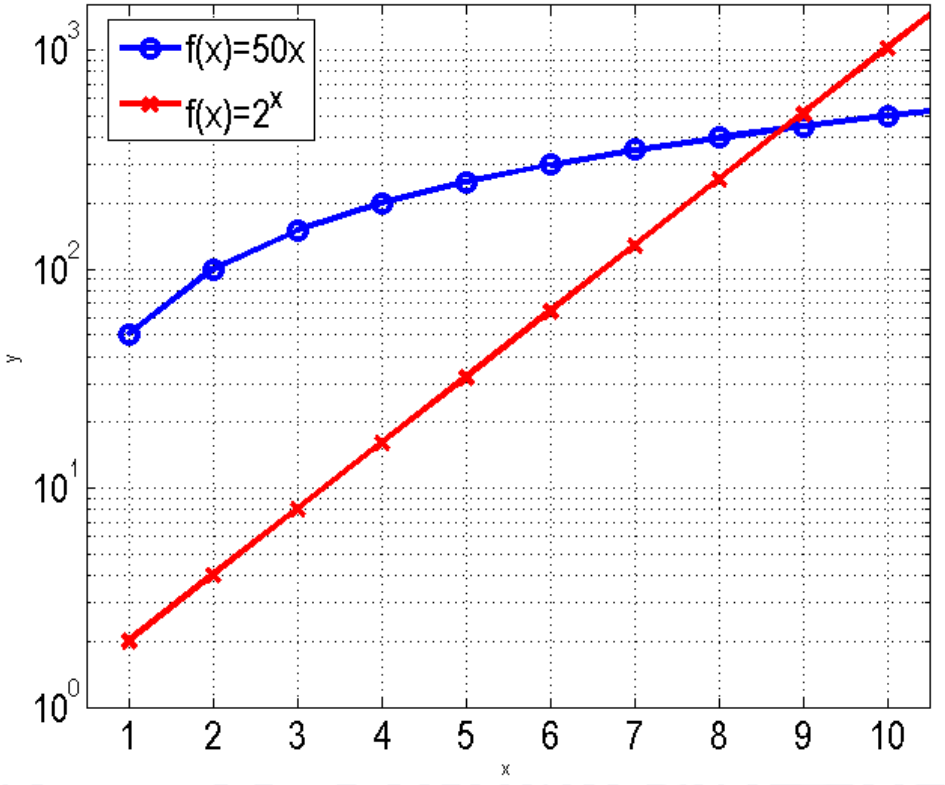
Crecimientos y medida del Rendimiento

- Crecimiento exponencial
  - formulación y ejemplo
  - casos de estudio de evolución temporal (1988-2001)
    - tiempo de ciclo del procesador
    - tiempo de acceso a fila de la memoria DRAM
- Rendimiento: tiempo, velocidad y consumo
- Ley de Amdhal

Crecimientos

- Lineal: el sistema no guarda memoria
  - Kilómetros de autopistas construidos por año
- Exponencial: cada valor se calcula a partir del anterior
  - Dinero en un banco, cuando los intereses se añaden al capital





Crecimiento exponencial

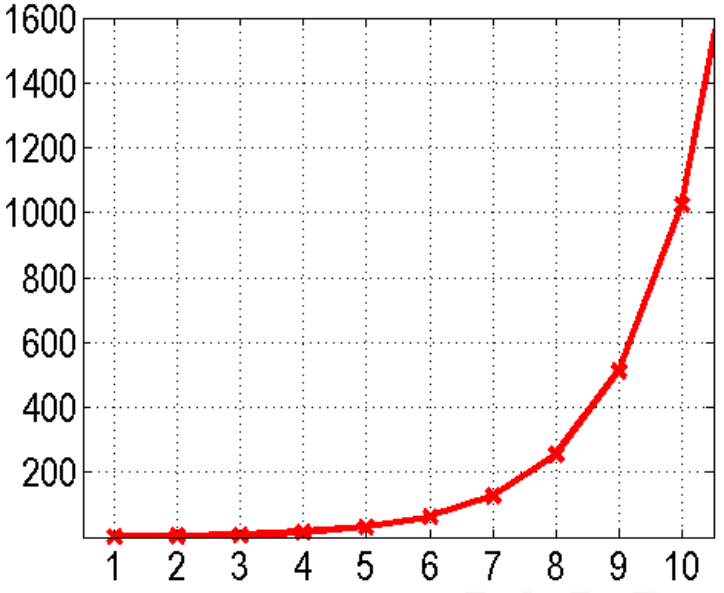
- En caso de dominios discretos con intervalos iguales
  - Crecimiento geométrico
- Fórmula 1: **factor** de multiplicación anual (*K*)

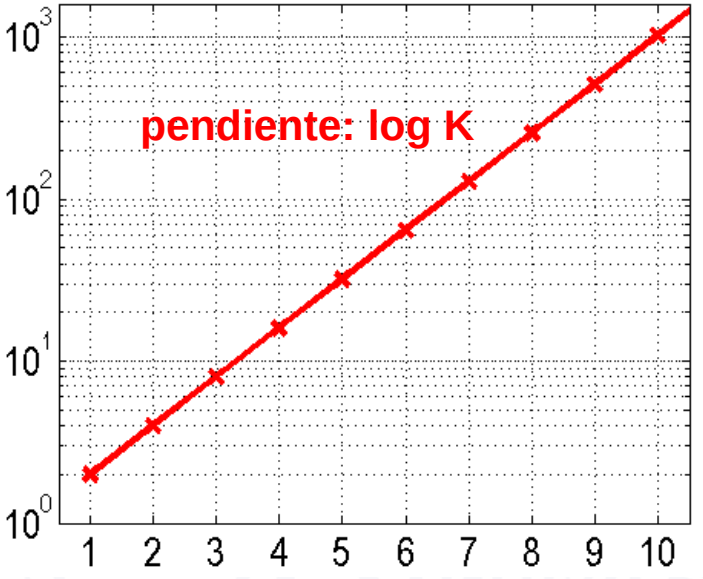
$$I(t) = I_0 * K^{\Delta t}$$

I_0 = índice en el año de referencia (t_0)

K = factor de crecimiento anual

Δt = diferencia en años ($t-t_0$)





Crecimiento exponencial

- Fórmula 2: **fracción** de crecimiento anual (ΔI), o tanto por uno ($\Delta I = \Delta I_{\%}/100$)

$$I(t) = I_0 * K^{\Delta t}$$

$$K = 1 + \Delta I$$

$$I(t) = I_0 * (1+\Delta I)^{\Delta t}$$

- Fórmula 3: **factor** de multiplicación (K_d) cada D años

$$I(t) = I_0 * K^{\Delta t}$$

$$K_d = K^D$$

$$K = K_d^{1/D}$$

$$I(t) = I_0 * (K_d^{1/D})^{\Delta t} = I_0 * K_d^{\Delta t/D}$$

Ejemplos de crecimiento

Dos formas de expresar
el mismo incremento exponencial:

- Crecimiento anual en %
  - "Las prestaciones de los sistemas informáticos en los años 60 y 70 crecían un **26% cada año**"
- Factor de multiplicación K_d cada D años
  - "Las prestaciones de los sistemas informáticos en los años 60 y 70 se **duplicaban cada 3 años**"
- Misma realidad con dos fórmulas:

$$1{,}26^3 = 2$$

Tiempo de ciclo del procesador

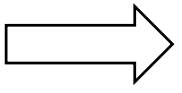
- Frecuencias y tiempo de ciclo de procesadores, 1988 -2001

año	procesador	frecuencia (MHz)	ciclo (nsg.)
1988	motorola 88000	25	40,0
1990	intel i860	40	25,0
1991	HP 730	66	15,0
1991	mips R4000	100	10,0
1992	Alpha 21064	150	6,7
1993	Alpha 21064	200	5,0
1993	HP PA7100	100	10,0
1994	Alpha 21064	275	3,6
1994	HyperSparc	100	10,0
1995	PowerPC 604	133	7,5
1995	Alpha 21164	333	3,0
1996	Alpha 21164	500	2,0

año	procesador	frecuencia (MHz)	ciclo (nsg.)
1996	HP PA8000	180	5,6
1997	pentium II	300	3,3
1997	Alpha 21164	600	1,7
1998	Alpha 21264	800	1,3
1998	mips R12000	300	3,3
1999	pentium III	733	1,4
1999	Sparc Ultra II	300	3,3
2000	pentium III	1130	0,9
2000	HP PA8600	550	1,8
2001	pentium 4	2000	0,5
2001	PowerPC 74XX	733	1,4

Motorola
Intel
PA-Risc
Mips

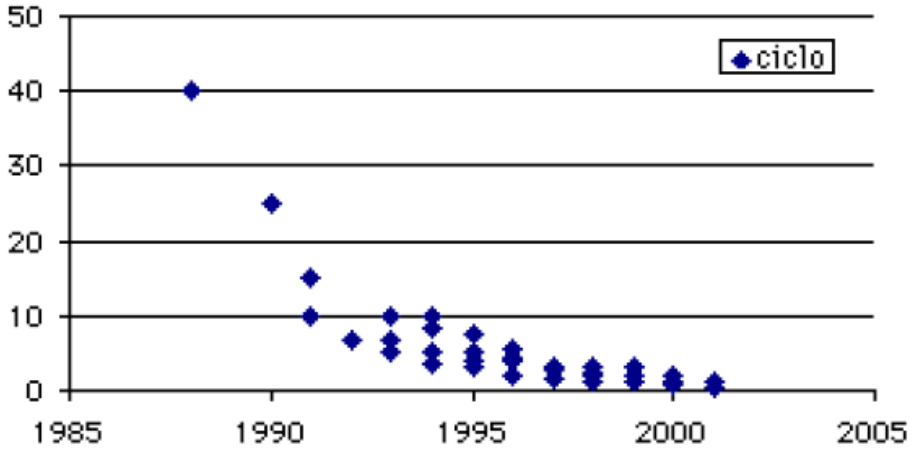
Alpha
Sparc
PowerPC

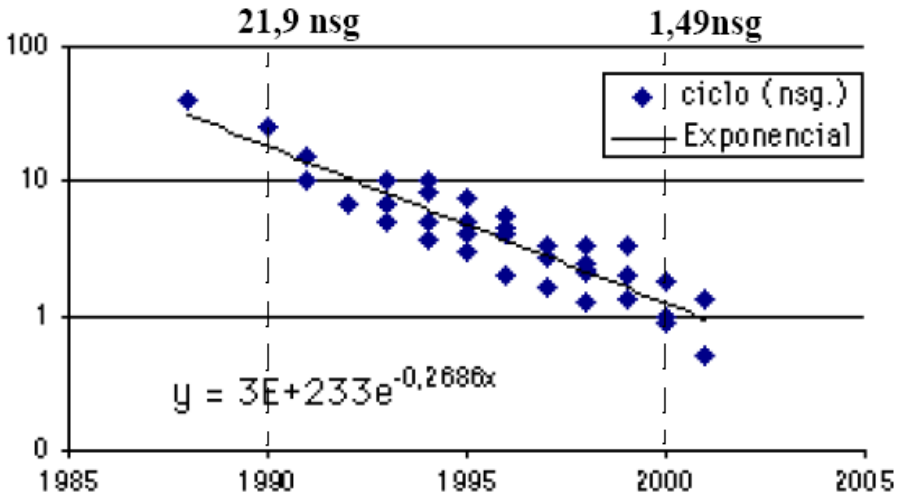


7 lenguajes máquina

Tiempo de ciclo del procesador

- Tiempo de ciclo de procesadores,1988-2001





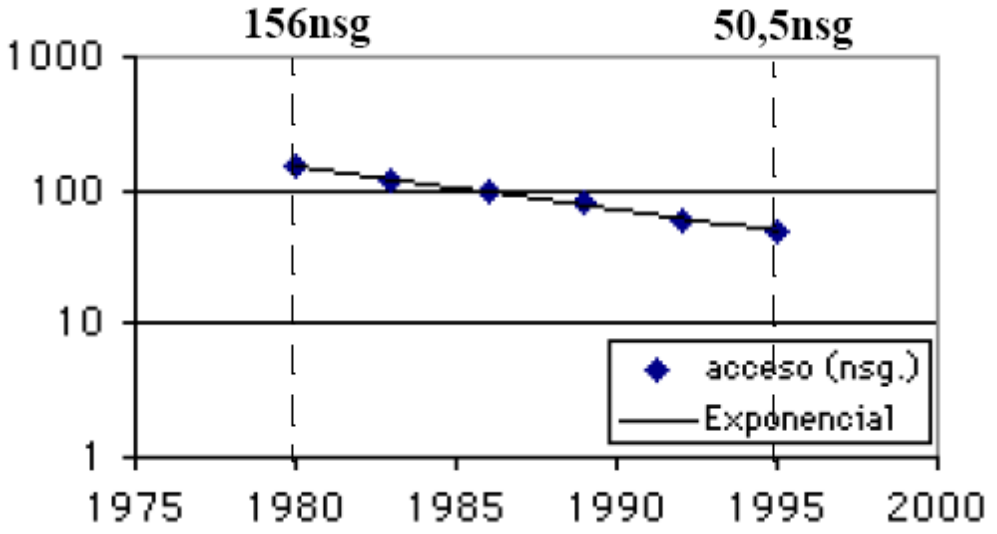
- Expresión para Tiempo de ciclo (Tc) en función del año:

?

Tiempo de acceso a memoria DRAM

- Tiempo de acceso a fila, 1980-1995

DRAM: tiempo de acceso RAS	
año	tiempo de acceso
1980	150
1983	120
1986	100
1989	80
1992	60
1995	50



- Expresión para Tiempo de acceso (Ta) en función del año:

?

Rendimiento: tiempo, velocidad y consumo (1)

- Tiempo de ejecución de un trabajo (t)
  - Tiempo de ejecución, tiempo de respuesta, latencia
- Trabajos (T) por unidad de tiempo: día, segundo, ns ...(p)
  - Productividad *(throughput)*, ancho de banda (*bandwidth), velocidad*
  - IPS, MIPS, MFLOPS

$$e = v * t \qquad v = \frac{e}{t} \qquad p = \frac{T}{t}$$

- Consumo
  - Energía consumida (Julios)
  - Energía consumida por unidad de tiempo: potencia (Watios)

Rendimiento: formas de comparar (2)

- Tiempo: segundos
- Productividad = velocidad =
  - cálculos / tiempo
  - bytes / tiempo
- **x** es **n**% más rápido que **y**

$$p(x) = p(y) + \frac{n}{100} p(y)$$

$$\frac{t(y)}{t(x)} = \frac{p(x)}{p(y)} = 1 + \frac{n}{100}$$

$$n = \frac{p(x) - p(y)}{p(y)} * 100$$

- **x** es **r** veces más rápido que **y**

$$\frac{t(y)}{t(x)} = \frac{p(x)}{p(y)} = r$$

Ejemplo: IPS, MIPS, MFLOPS, EPI, EPFLOP

`Real*8 A(1000), B(1000), C(1000)`
`DO i=1, 1000`
`  A(i) = B(i) * C(i) + 3.0`
`ENDDO`
	9 instrucciones de LM
	2 operaciones PF
	1 μs todo el bucle
	50 W

- IPS = instrucciones por segundo =
- GIPS = IPS / 10^9 =
- GFLOPS
  = operaciones coma flotante por segundo / 10^9 =
  FLOP S
- EPI = energía por instrucción (nJ/inst) =
- EPFLOP = energía por FLOP (nJ/FLOP) =

Ley del Hierro: Tiempo de ejecución

$$T_{ex} = I \times CPI \times Tc$$

- ¿Bajo qué condiciones puedo usar MIPS como métrica de comparación entre dos computadores?
- ¿Bajo qué condiciones puedo usar xFLOPS?
- ¿Bajo qué condiciones Tex?

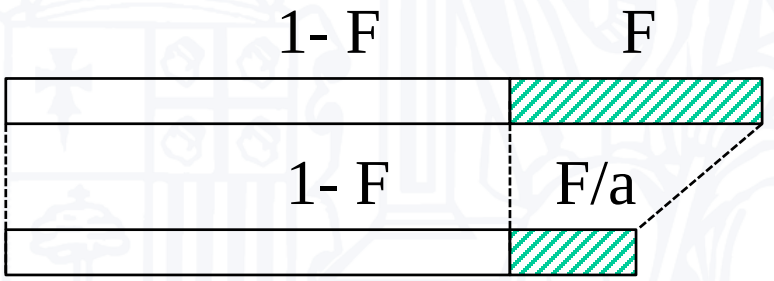
Ley de Gene Amdahl

- *Speedup*: incremento de prestaciones
  al introducir una mejora **E** en el sistema

$$\textbf{speedup(E)} = \frac{\text{prestaciones del nuevo sistema (con E)}}{\text{prestaciones del sistema viejo (sin E)}} = \frac{\text{Tiempo (sin E)}}{\text{Tiempo (con E)}}$$

- La ley de Amdahl formula el *speedup* en función de:
  - la fracción de tiempo (**F**) donde opera la mejora E
  - ganancia (factor **a**) durante ese tiempo

$$S_{up} = ?$$



Ley de Amdahl: ejercicio

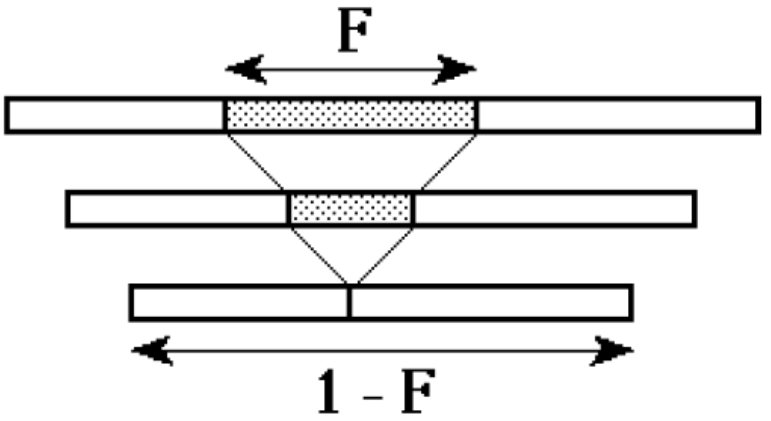
- Cambio de servidor web ?
  - estadísticas: 40% del tiempo haciendo cálculos, 60% E/S
  - nuevo procesador 10 veces más rápido haciendo cálculos
  - mantenemos el subsistema de E/S (discos + red)

Speed-up =

Ley de Amdahl: principios cuantitativos

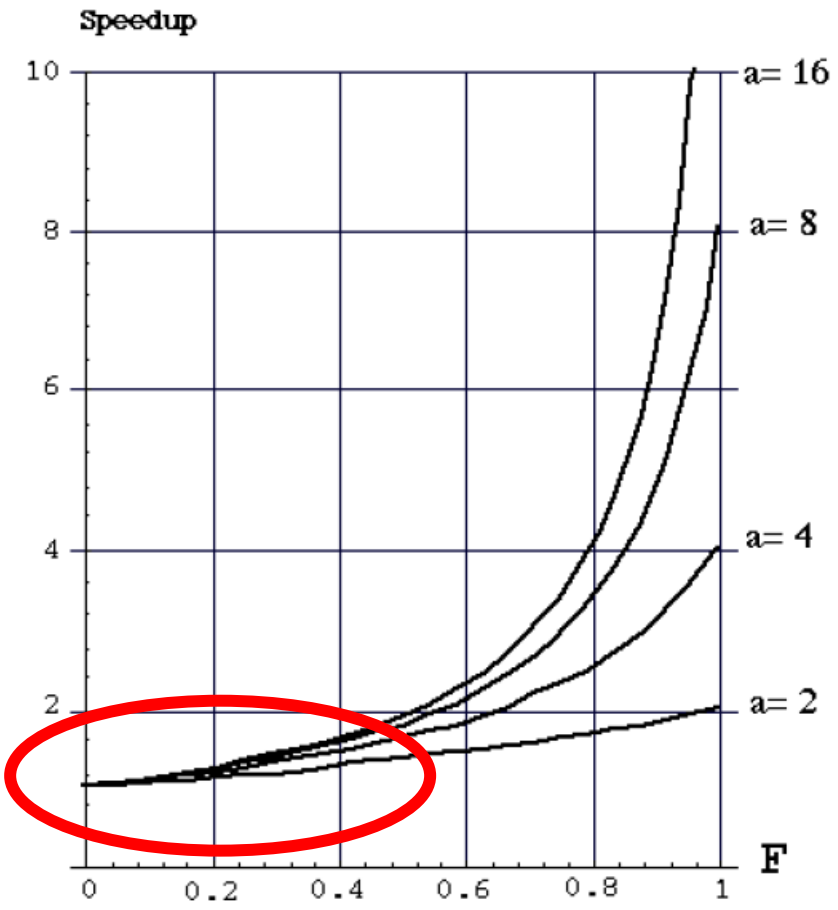
- Disminución del ratio Speedup / Coste
  - Si sólo se actúa sobre una parte del sistema, cada nueva mejora añadida trabaja sobre una fracción de tiempo F menor
- Límite de speedup
  - inverso de fracción de tiempo no mejorada

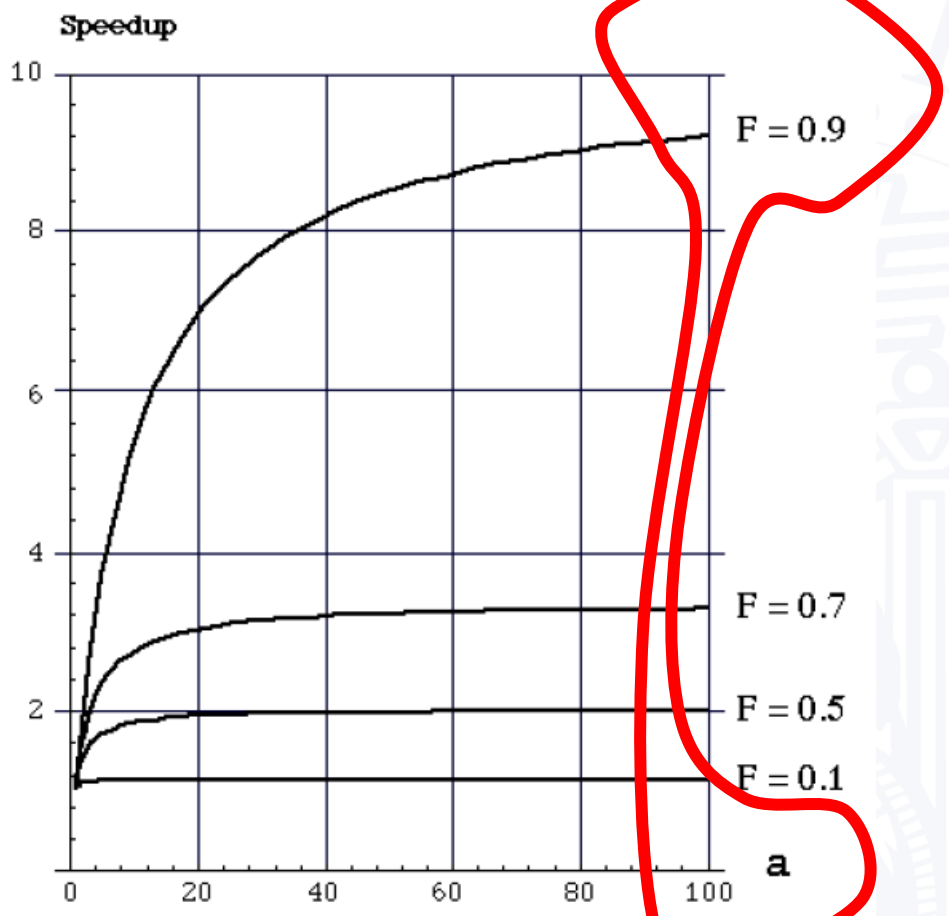
$$\text{máximo speedup(E)} = \frac{1}{1 - F}$$



- Se debe actuar siempre sobre el **caso más común**, el que mayor F consume
- El caso más **simple** suele ser el **más común**

Ley de Amdahl: ejemplo



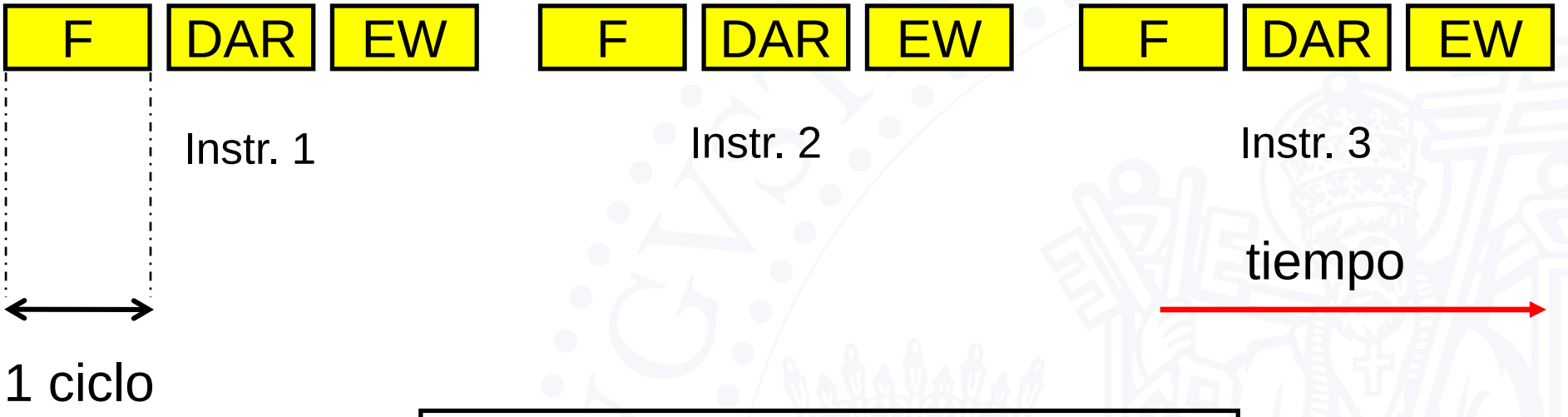


No importa el valor de **a** si **F** es pequeña

Con valores grandes **a**, el límite de Speedup es **1/(1-F)**

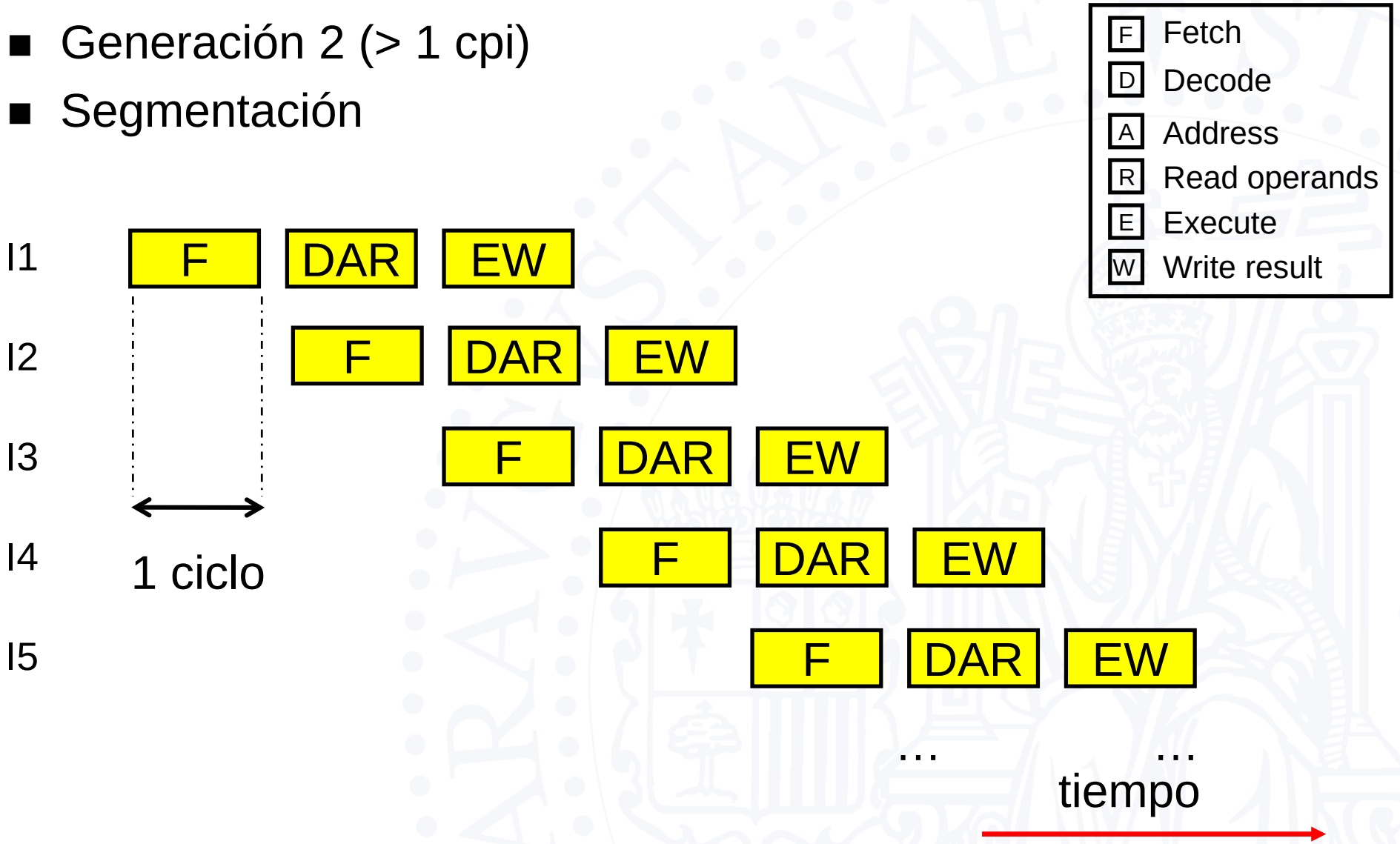
Modelo de ejecución y rendimiento

- Generación 1 (> 3 **c**iclos **p**or **i**nstrucción)



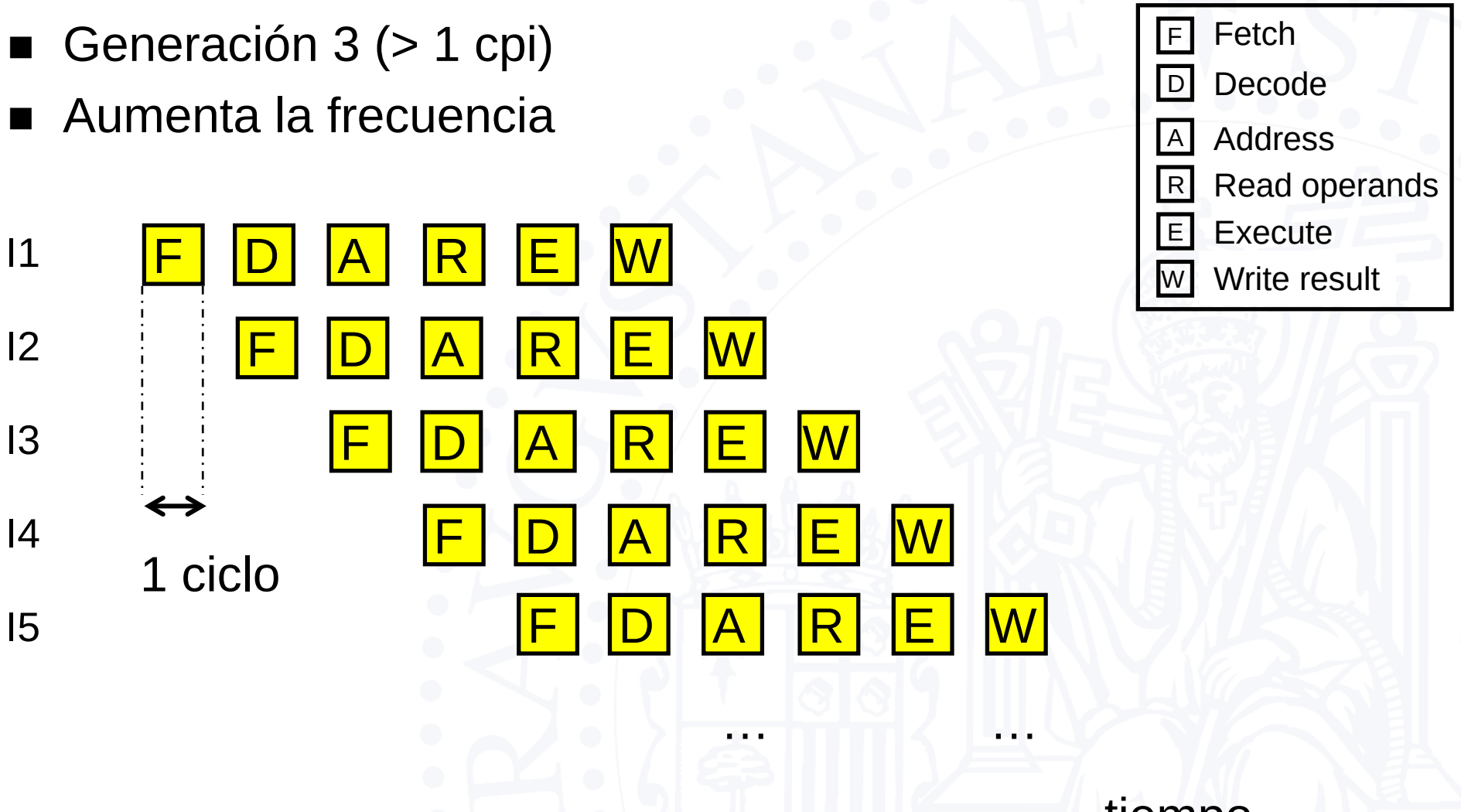
Evolución en el modelo de ejecución

- Generación 2 (> 1 cpi)
- Segmentación



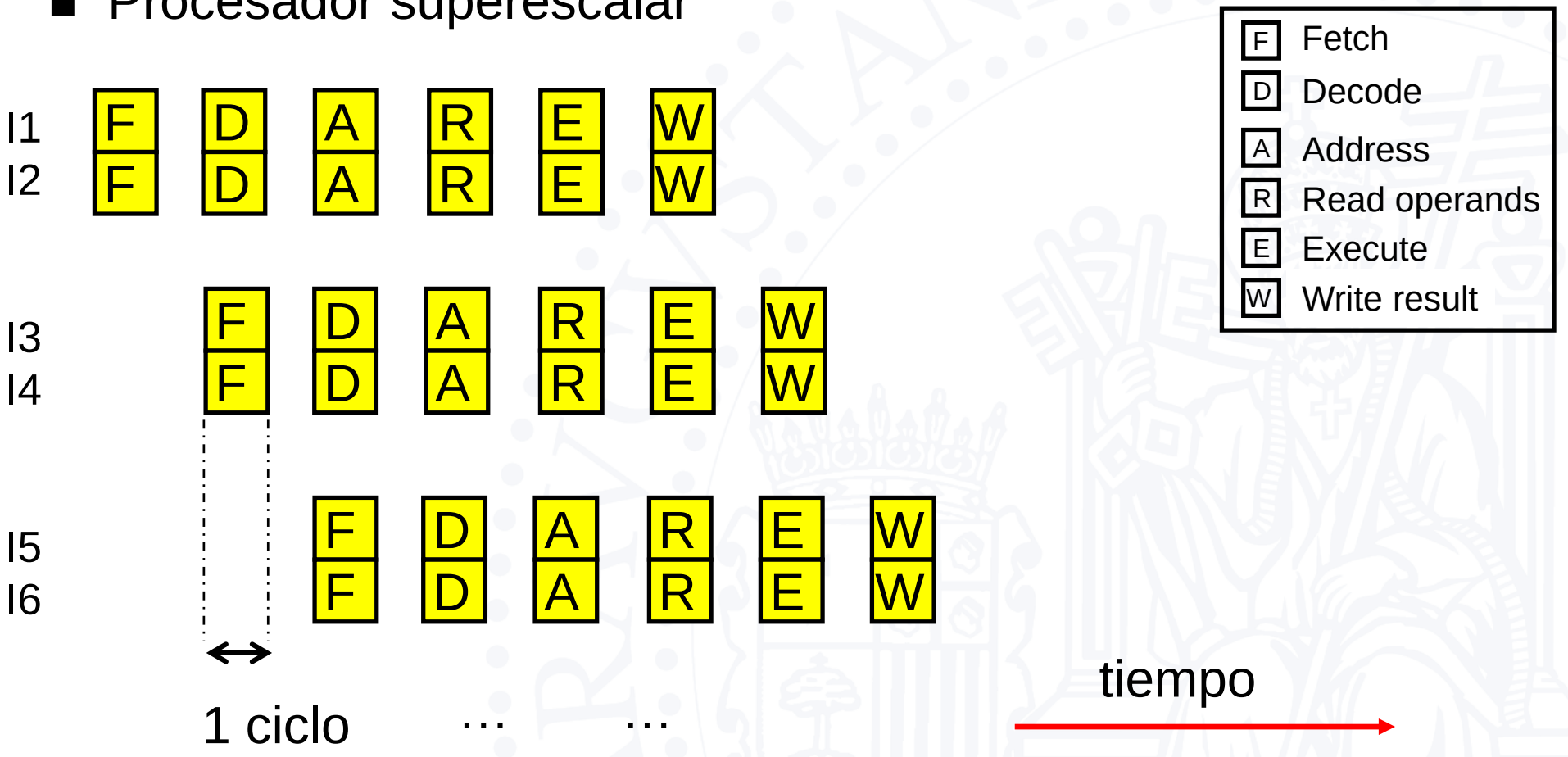
Evolución en el modelo de ejecución

- Generación 3 (> 1 cpi)
- Aumenta la frecuencia



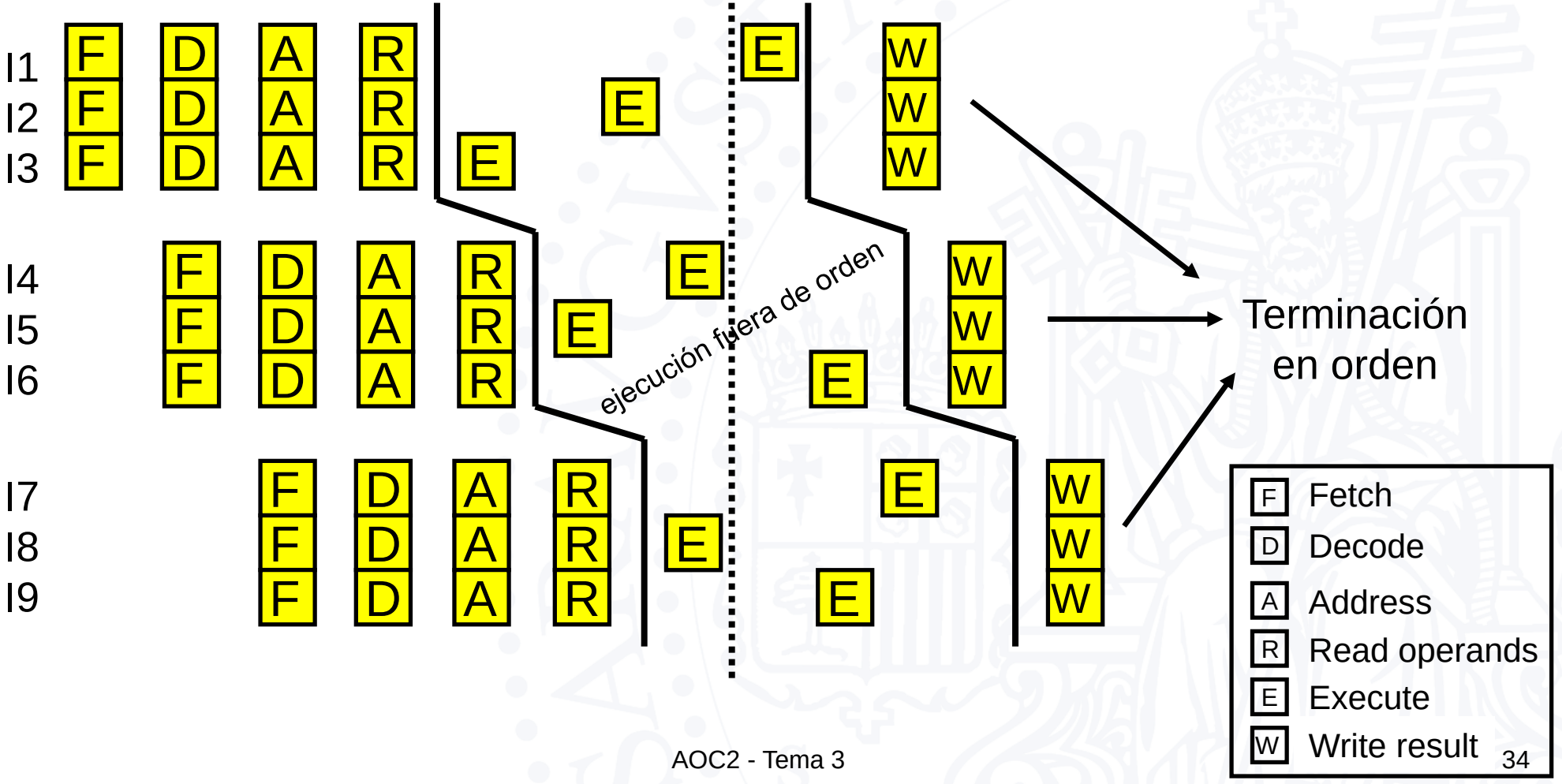
Evolución en el modelo de ejecución

- Generación 4 (> 0.5 cpi) – en μ a finales de los 80 –
- Procesador superescalar



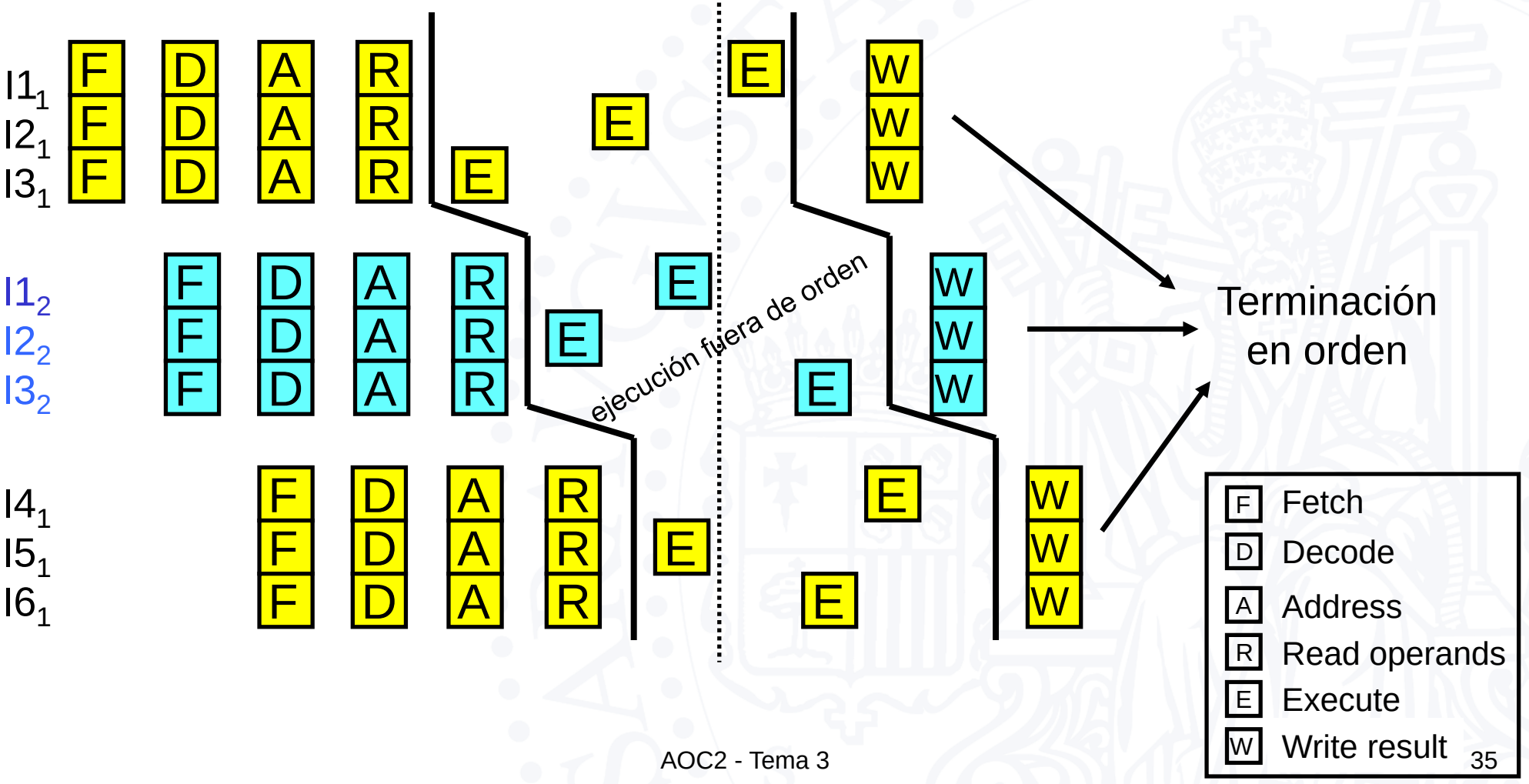
Evolución en el modelo de ejecución

- Generación 5 (≈ 0.3 cpi) – en μ a principios de los 90 –
- Ejecución en desorden



Evolución en el modelo de ejecución

- Generación 6 (≈ 0.25 cpi) – en μ en los años 2000 –
- Multithreading

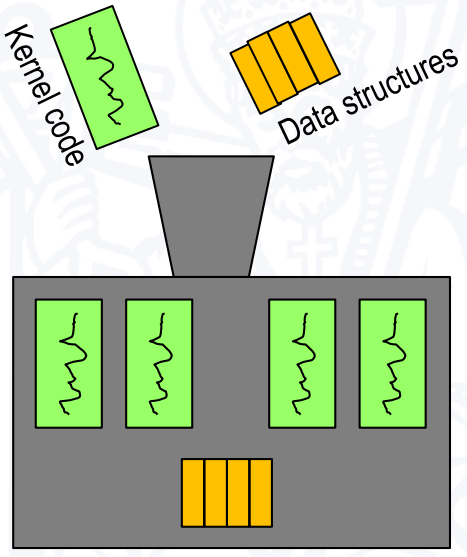


Otros modelos de ejecución

- SIMD / SIMP (CUDA/OpenCL)
  - Ejemplo: C = A + B

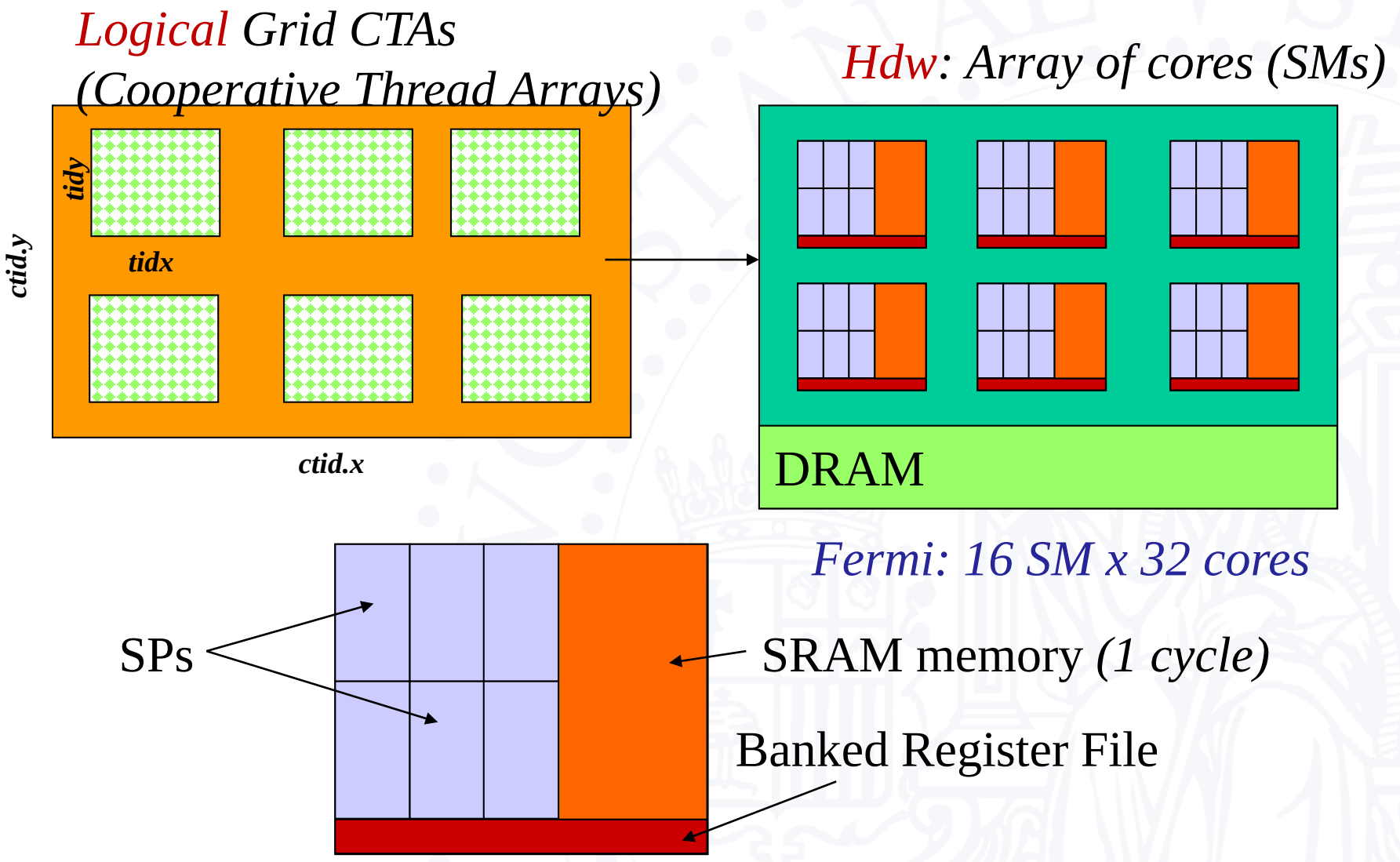
```
__global__ void matAdd(float A[N][N], float B[N][N],float C[N][N])
{
        int i = threadIdx.x;
        int j = threadIdx.y;
        C[i][j] = A[i][j] + B[i][j];
}

int main()
{
// Allocate structures in device
// Transfer data to device
        …
// Set grid parameters and number of threads
        dim3 dimBlock(N, N);
// Kernel invocation
        matAdd<<<1, dimBlock>>>(A, B, C);
// Transfer results from device
…
}
```



Nvidia device

Programming model and hdw



Hacia dónde vamos ahora en AOC2

- Incremento del rendimiento por segmentación
  - Diseño de la ruta de datos segmentada
  - Diseño del control
  - Problemas para conseguir el CPI = 1: riesgos y soluciones
    - Estructurales
    - De datos
    - De control