Arquitectura e Ingeniería de Computadores

Tema 1

Introducción: Tendencias Tecnológicas

Costo/ Rendimiento

DEPARTAMENTO DE ARQUITECTURA DE COMPUTADORES Y AUTOMÁTICA

Curso 2009-2010

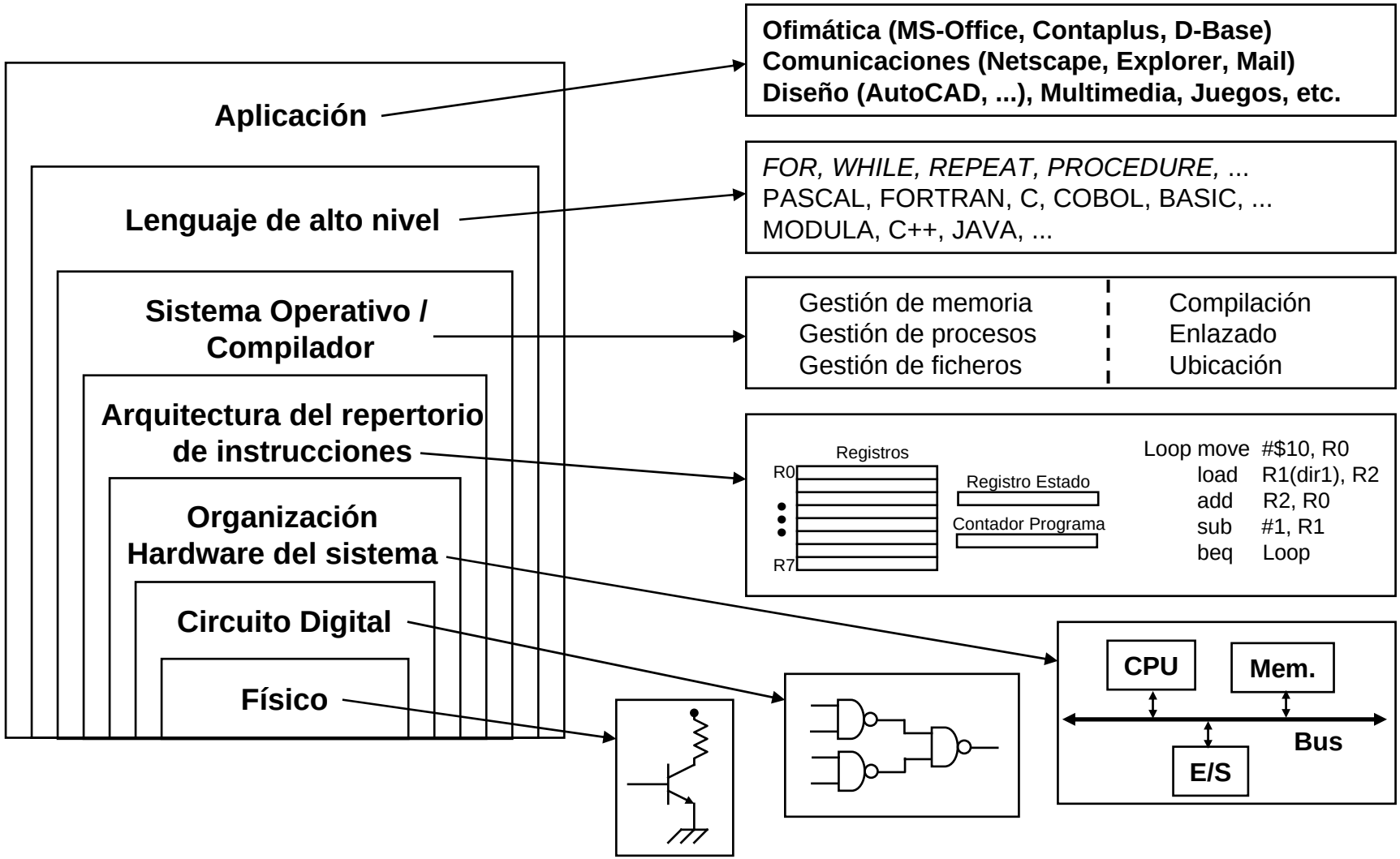
Contenidos

- La asignatura. ¿ Qué estudia?
- El entorno tecnológico
- Rendimiento
- Costo
- Un principio simple
- Bibliografía

Capítulo 1 de [HePa07]
Semiconductor Industry Association. http://public.itrs.net
Standard Performance Evaluation Corporation. http://www.spec.org
Transaction Processing Council. http://www.tpc.org
The Embedded Microprocessor Benchmark Consortium.
http://www.eembc.org

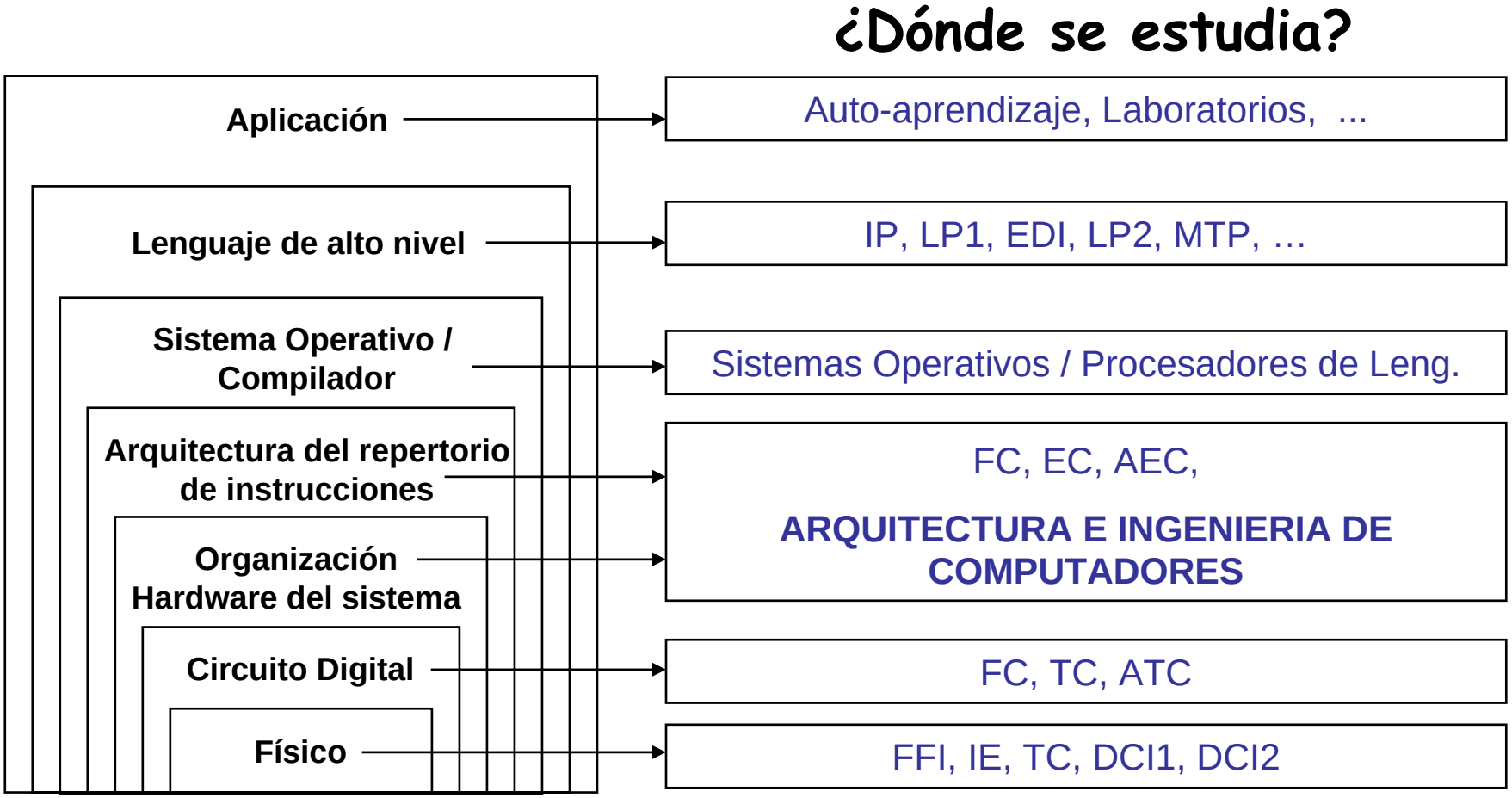
La asignatura

❑ Niveles de descripción y diseño de un computador



La asignatura

❑ Niveles de descripción y diseño de un computador

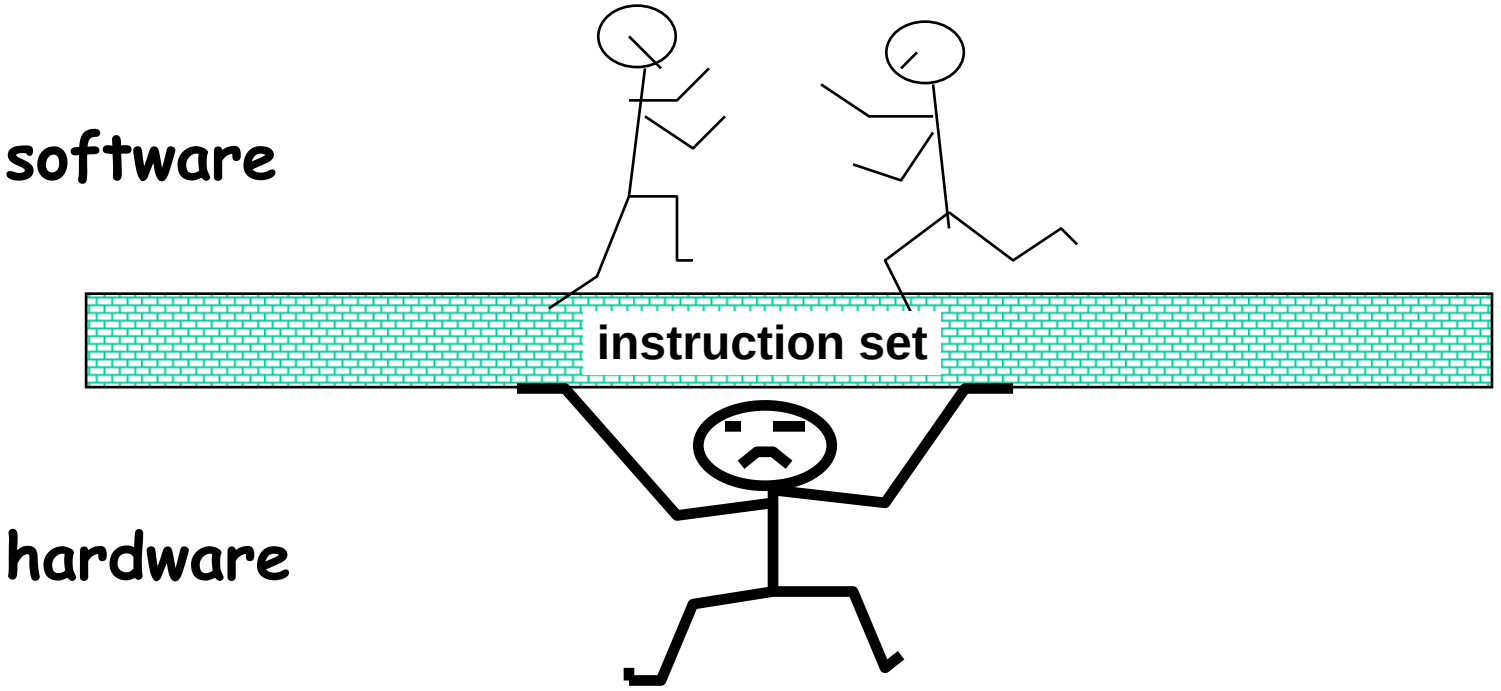


La asignatura

Arquitectura de computadores

- Los atributos de un computador tal y como los ve un programador en lenguaje ensamblador. La estructura conceptual y el modelo funcional (modelo de programación).
  Amdahl, Blaaw, Brooks 1964
- El concepto ha cambiado en el tiempo.
  - Hasta la mitad de los 80. El énfasis era el diseño de juego de instrucciones orientado a los LAN.
  - Desde entonces el énfasis es el diseño de CPU, Jerarquía de memoria, sistema de I/O. Aspectos clave coste-rendimiento-tecnología
- Tres aspectos
  - Arquitectura del juego de instrucciones
  - Organización (diferentes organizaciones P6, Netburst AMD, Core)
  - Implementación (PentiumIII, Celeron, Pentium4, Pentium Xeon, Core2)

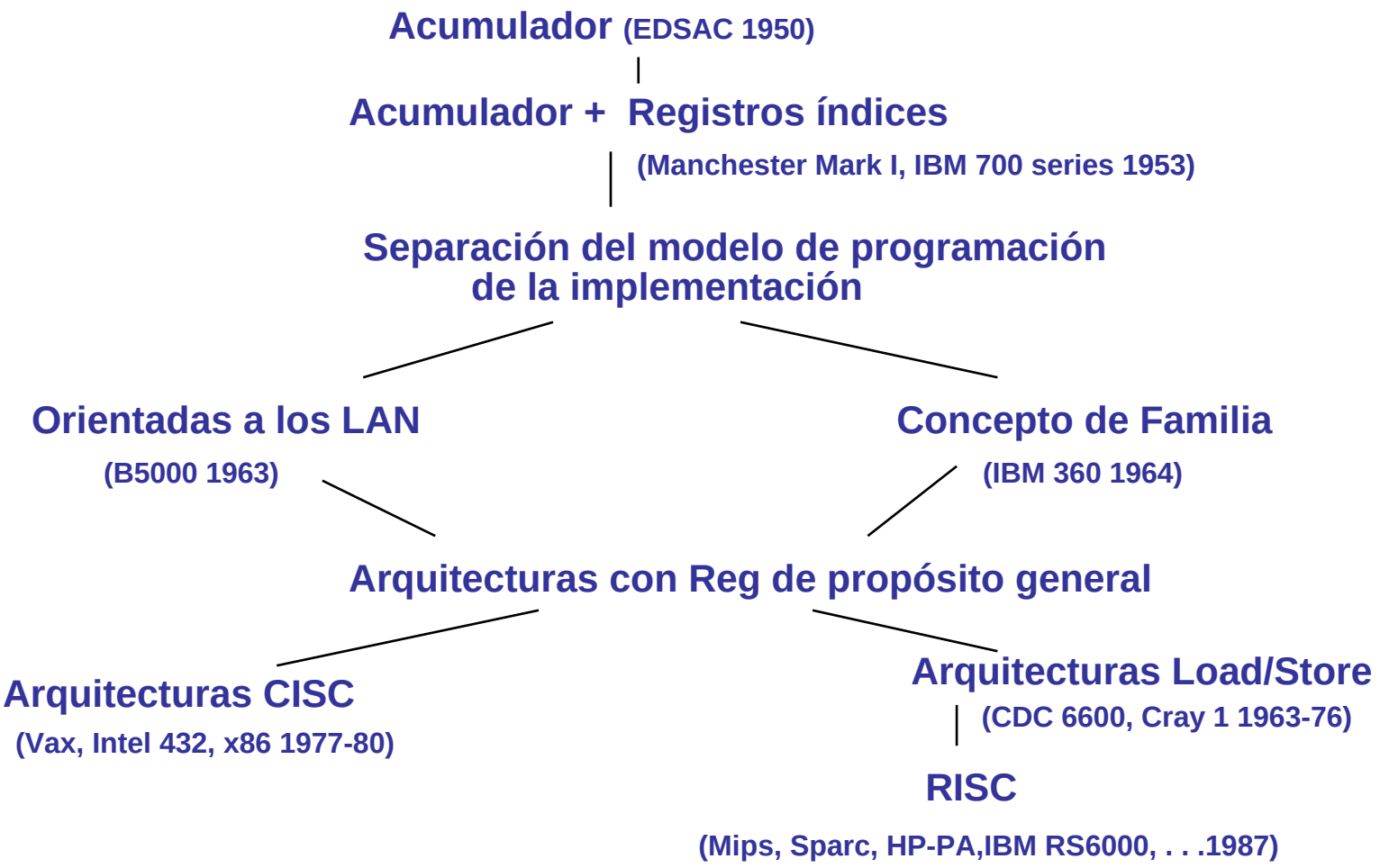
ISA: Interfase Critico



- Propiedades
  - Permanencia con el tiempo / tecnología (portabilidad)
  - Proporciona funcionalidad eficaz a los niveles superiores
  - Permite implementación eficiente en los niveles inferiores

La asignatura

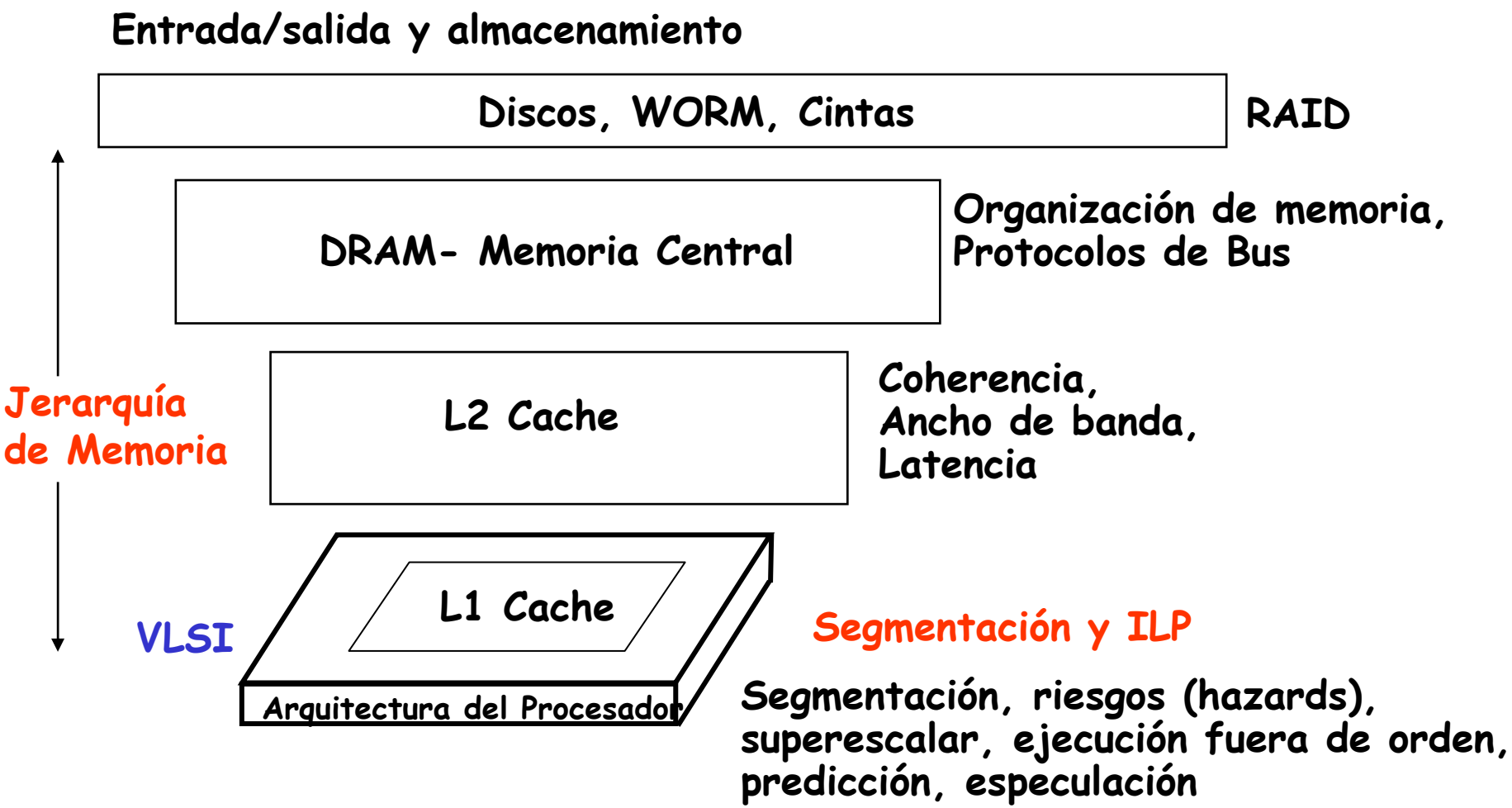
- Evolución de los juegos de instrucciones



Metodología de Diseño

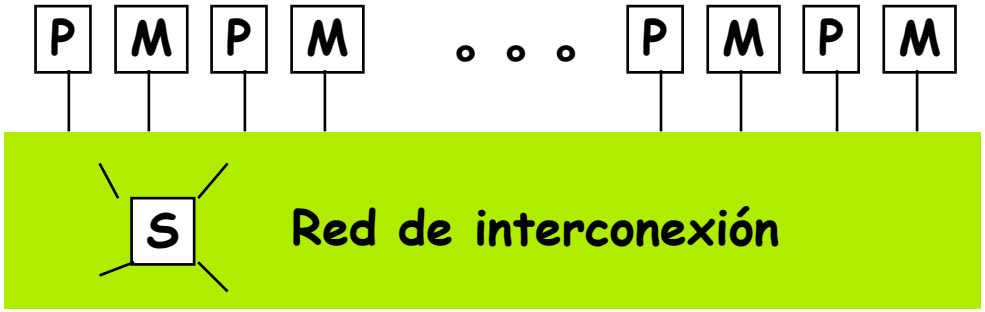
La asignatura

- ¿Qué estudia la asignatura?



La asignatura

- ¿Qué estudia la asignatura?



Switch (S) Procesador (P) Memoria (M)

Multiprocesadores
Redes de Interconexión

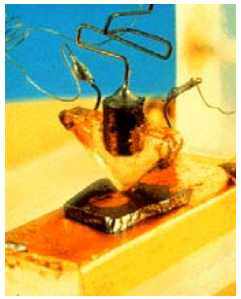
Memoria compartida, paso de mensajes, paralelismo de datos

Red

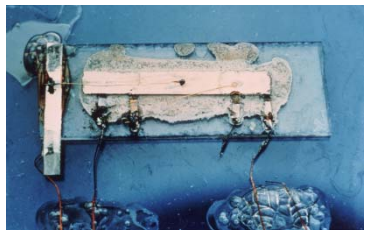
Topología,
Routing,
Ancho de banda,
Latencia,

El entorno

- 1949 EDSAC 10^2 op/seg
- 1957 Transistor: de 10^3 a 10^4 op/seg
  - DEC PDP-1 (1957)
  - IBM 7090 (1960)
- 1965 CI: de 10^5 a 10^6 op/seg
  - IBM System 360 (1965)
  - DEC PDP-8 (1965)
- 1971 Microprocesador
  - Intel 4004
- 2003 más de $3x10^{13}$ op/seg
- 2008 > 10^{16} op/seg un petaflops
  - IBM Roadrunner 1,37Pflops, 122400pc
    - 6562 dual core AMD +12240 Cell 8i
  - MareNostrum 94Tflops 10240pc

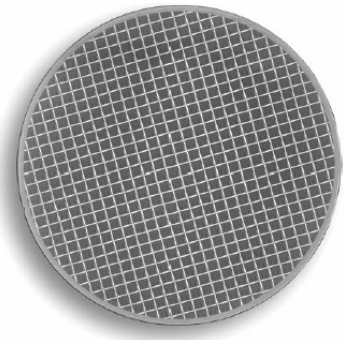


Transistor (47) PN 56



CI (58) PN2000

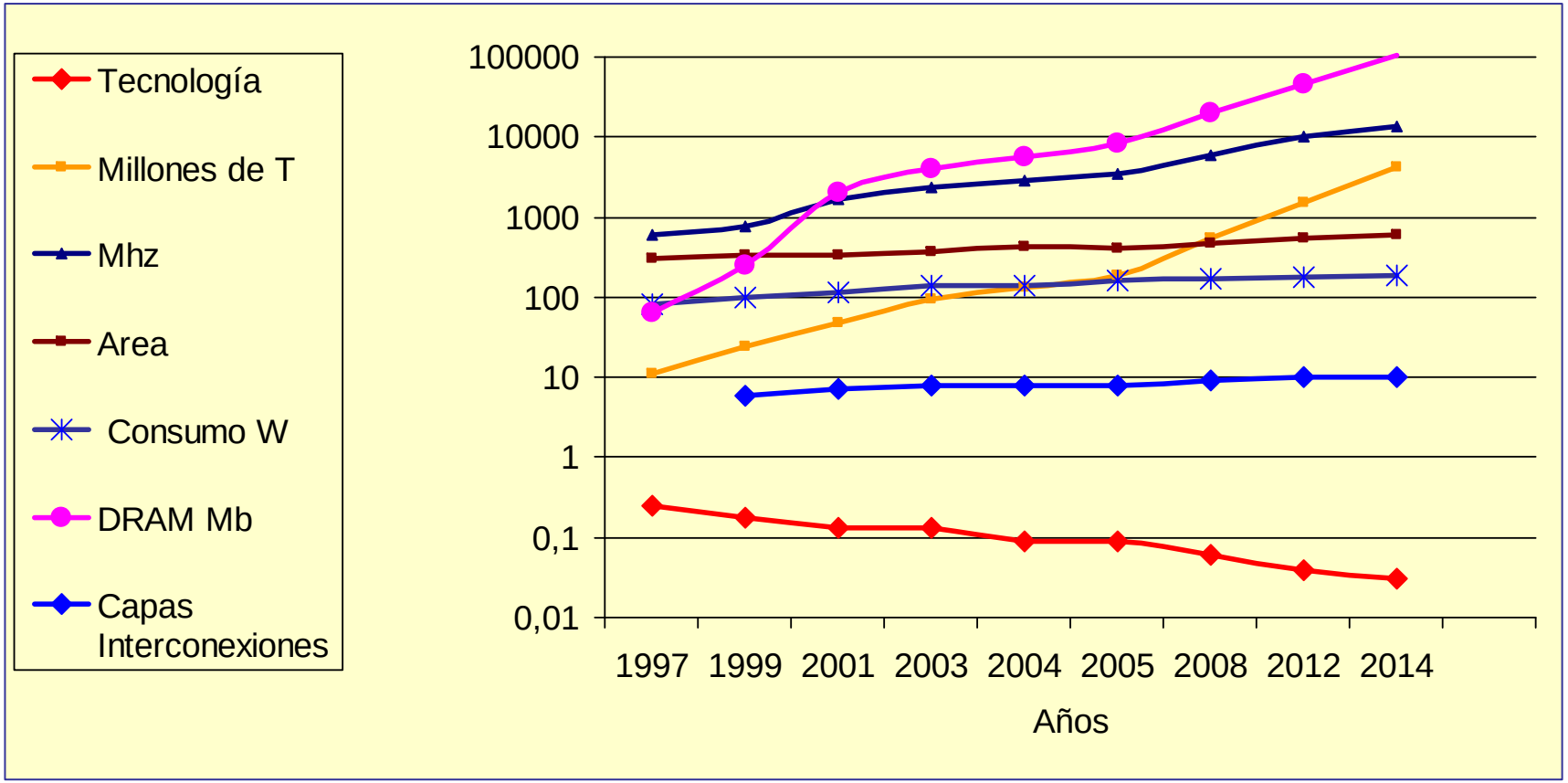
P 4



Oblea (Wafer)

El entorno

- Tecnología: CMOS VLSI domina todas las demás tecnologías
  - (TTL, ECL, AsGa,..) en costo y rendimiento.
  - Arquitectura : Mejoras basadas en la tecnología. La tecnología impulsa la desarrollos arquitectónicos.



❑ La Ley de Moore

Cramming More Components onto Integrated Circuits

GORDON E. MOORE, LIFE FELLOW, IEEE

With unit cost falling as the number of components per circuit rises, by 1975 economics may dictate squeezing as many as 65 000 components on a single silicon chip.

The future of integrated electronics is the future of

Each approach evolved rapidly and converged so that each borrowed techniques from another. Many researchers believe the way of the future to be a combination of the

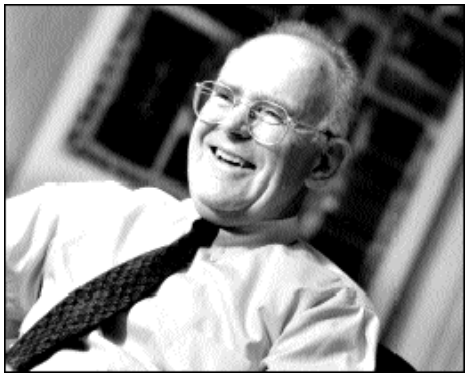


Fig. 2.

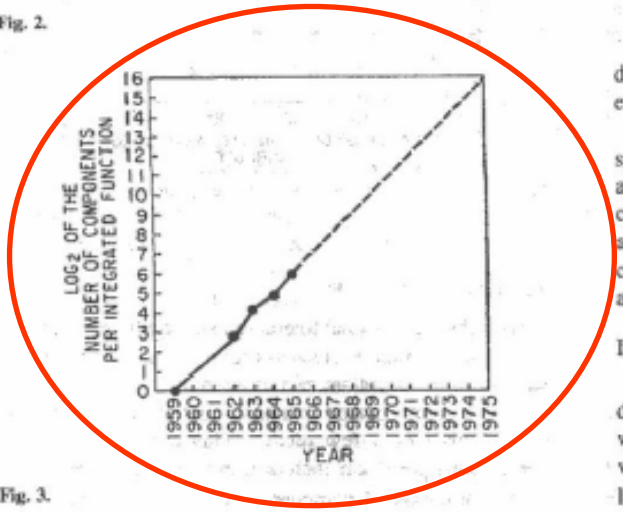


Fig. 3.

diagram to technological realization without any sp
engineering.

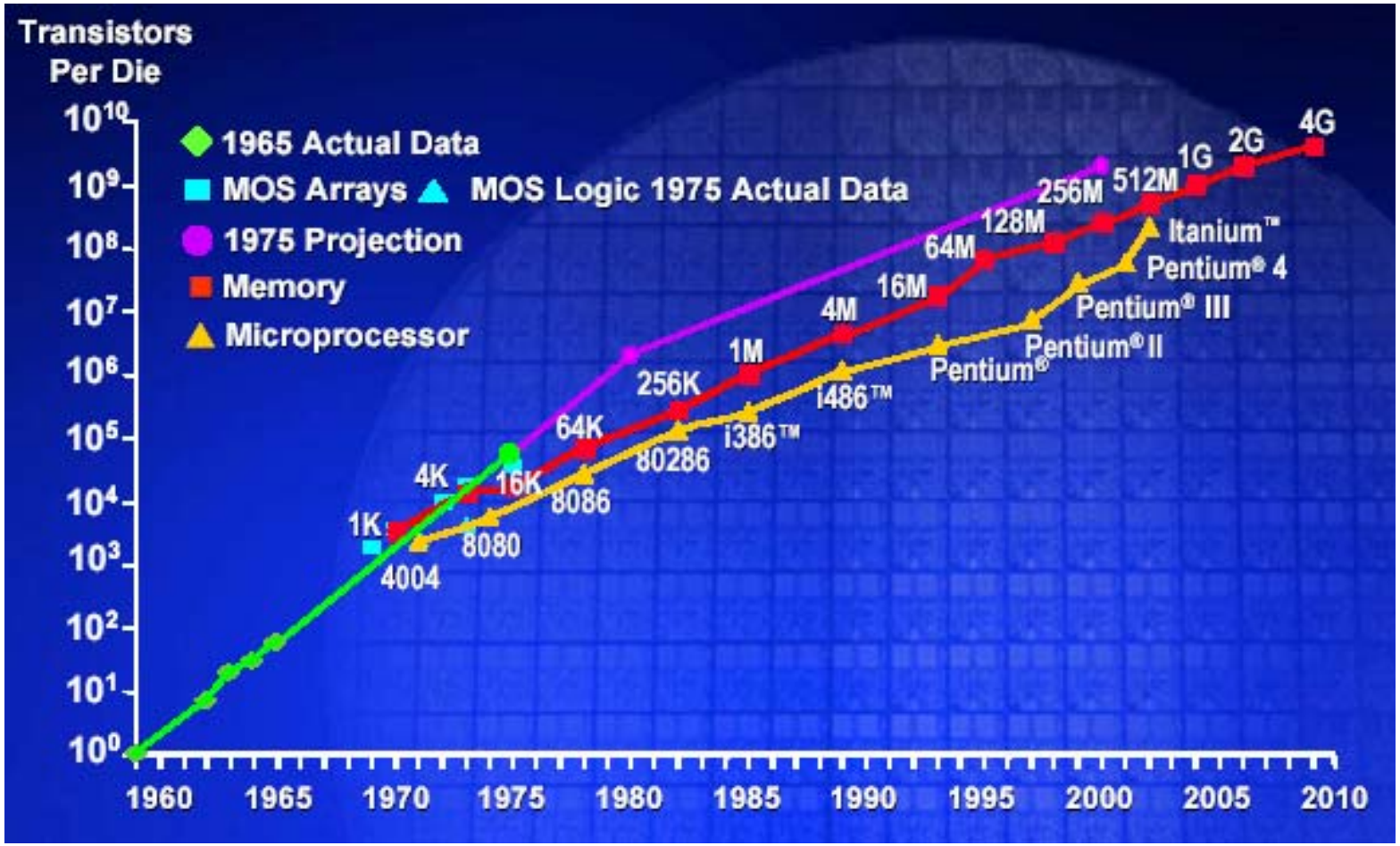
It may prove to be more economical to build
systems out of smaller functions, which are separately p
aged and interconnected. The availability of large funct
combined with functional design and construction, sh
allow the manufacturer of large systems to design
construct a considerable variety of equipment both ra
and economically.

IX. Linear Circuitry

Integration will not change linear systems as radical
digital systems. Still, a considerable degree of integr
will be achieved with linear circuits. The lack of l
value capacitors and inductors is the greatest fundam
limitation to integrated electronics in the linear area.

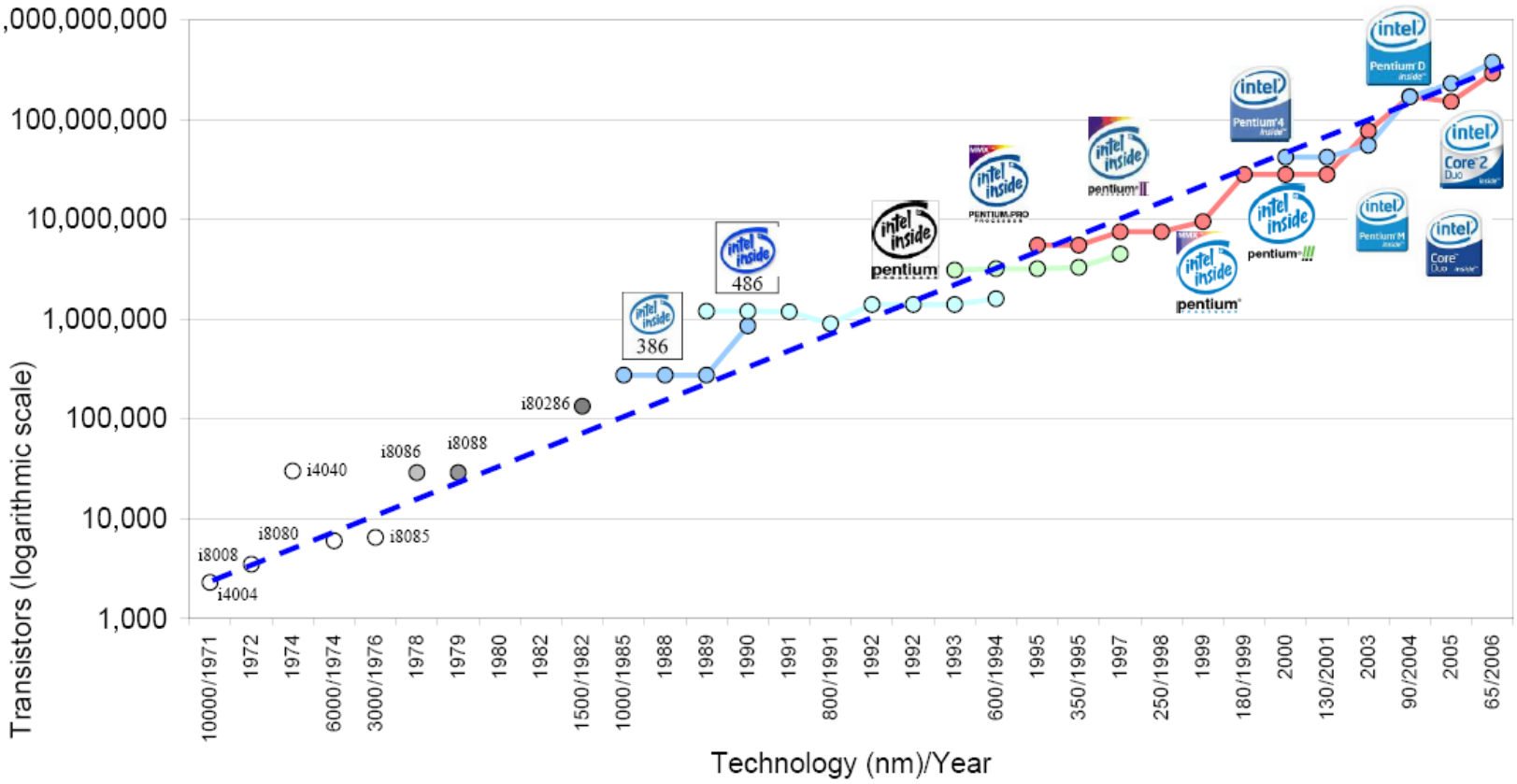
Electronic- Abril1965

❑ La Ley de Moore se ha cumplido



Fuente: Intel Corporation

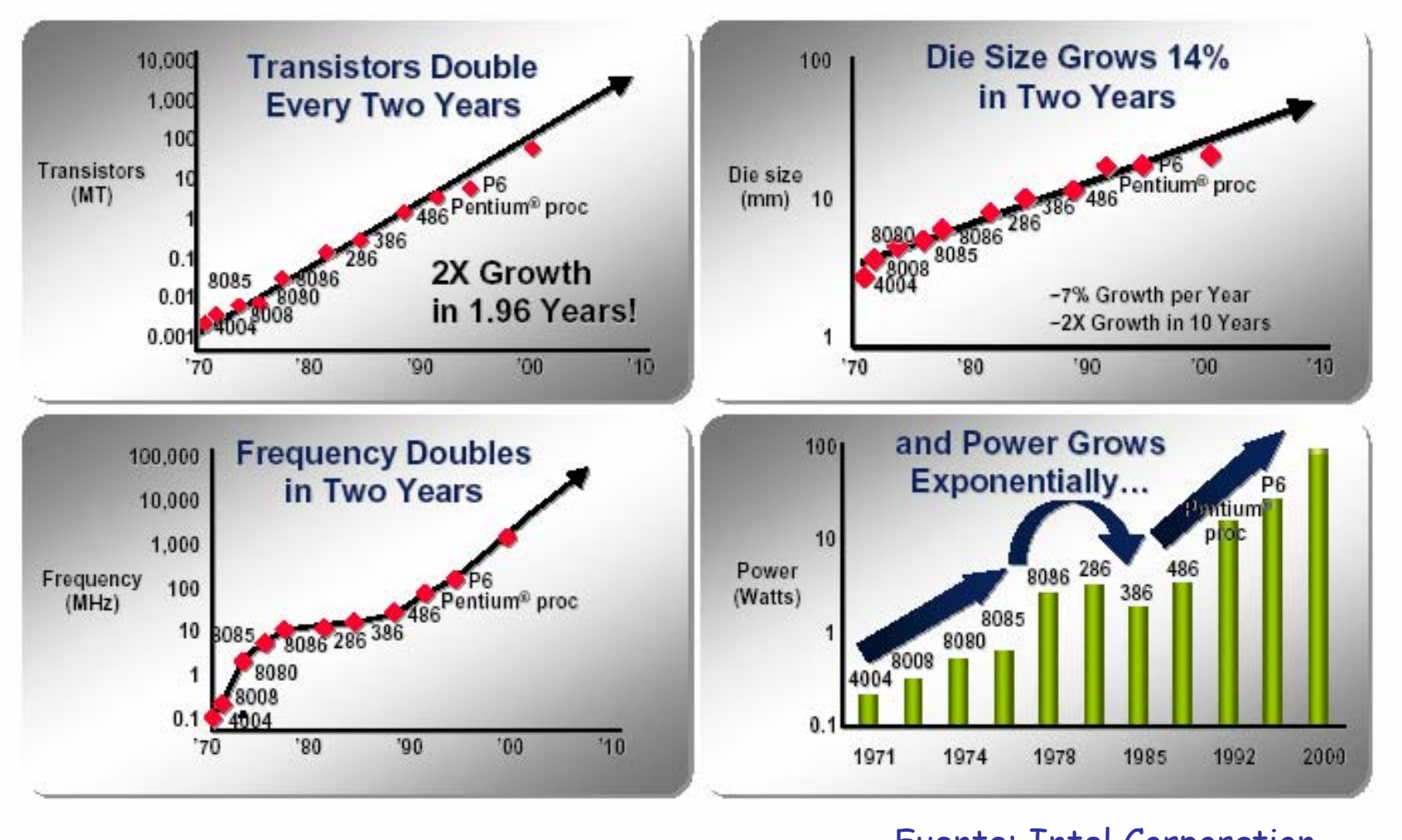
❑ La Ley de Moore se ha cumplido



Fuente: Intel Corporation

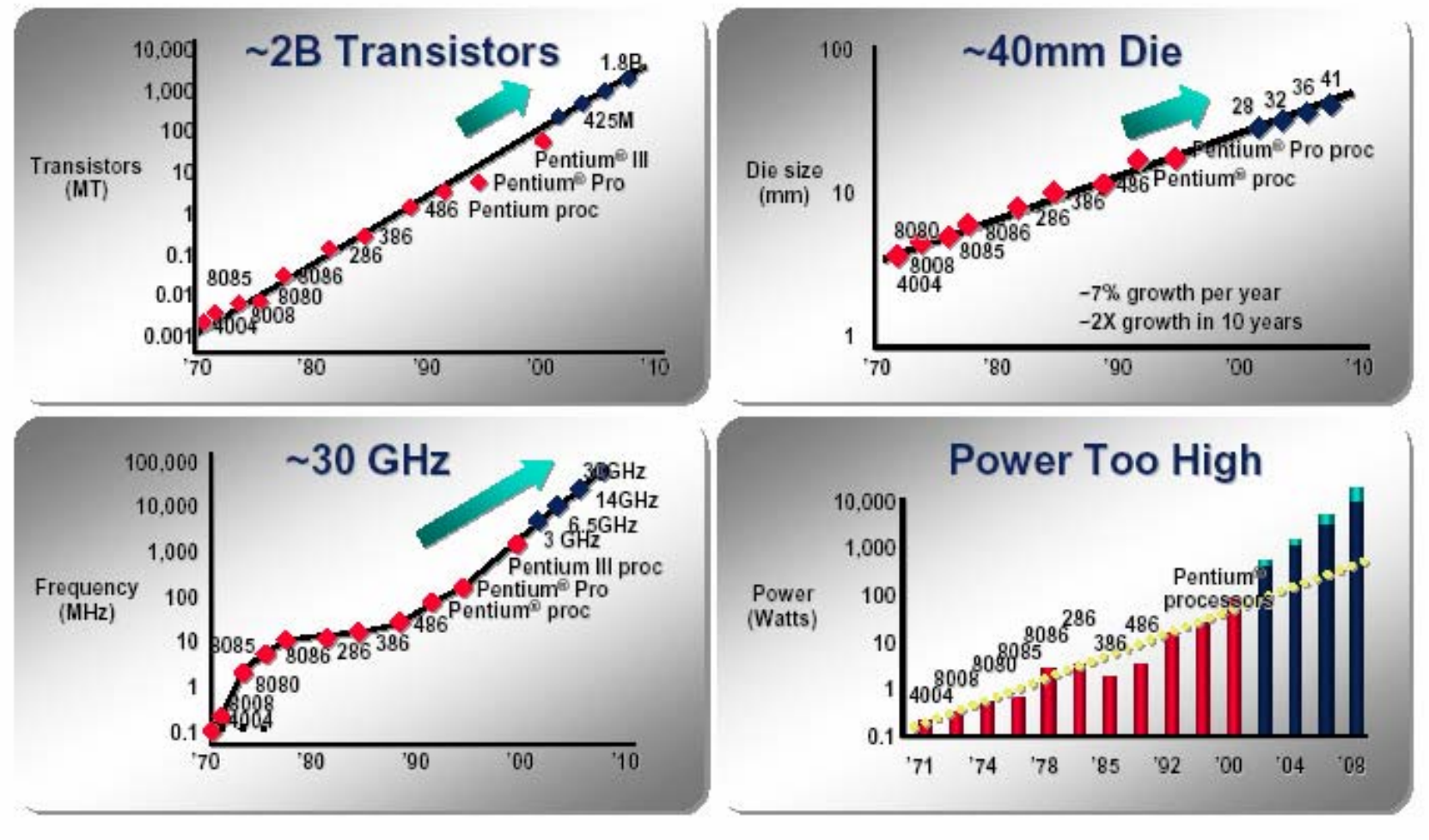
La Ley de Moore

❑ La Ley de Moore se ha cumplido



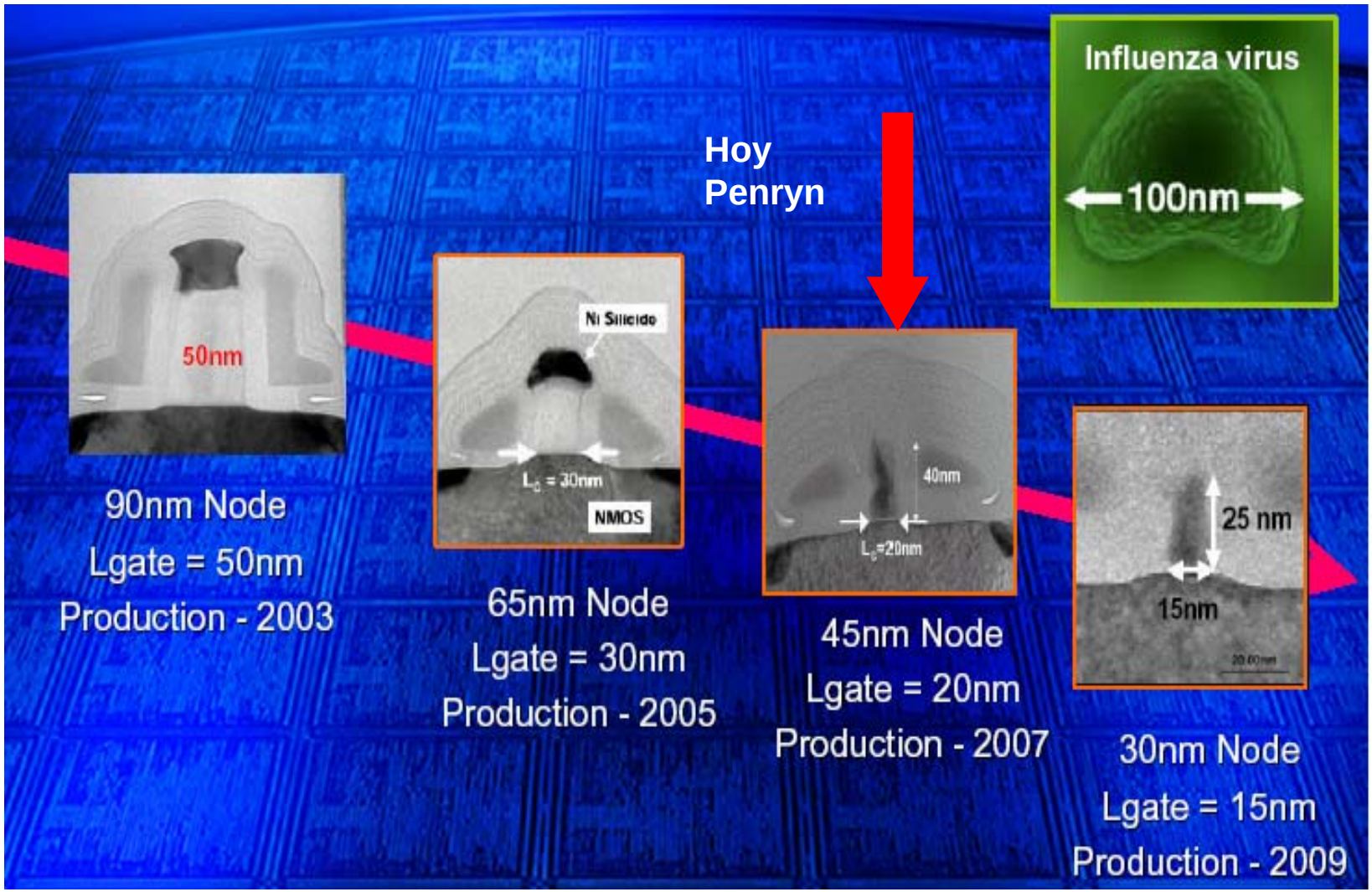
Fuente: Intel Corporation

❑ La Ley de Moore continua



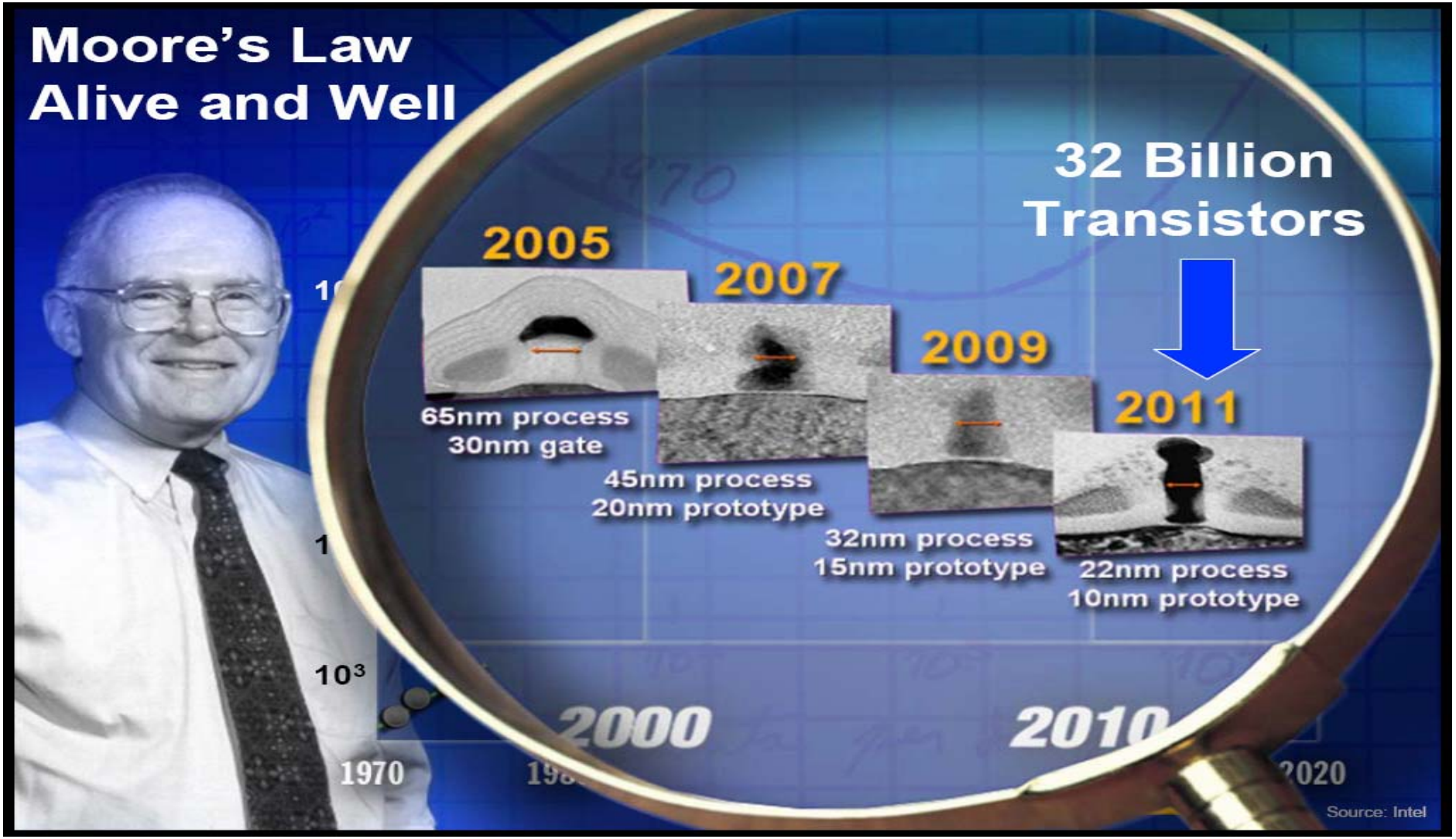
Fuente: Intel Corporation

❑ La Ley de Moore continua



Fuente: Intel Corporation

La Ley de Moore

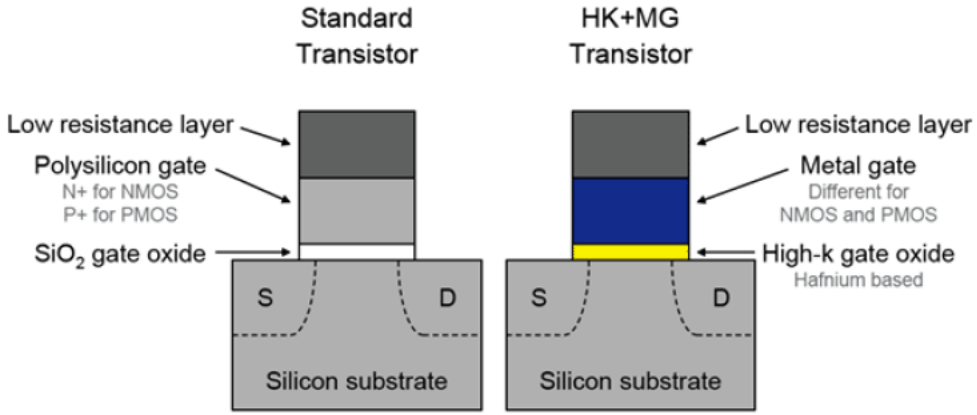


La Ley de Moore

Process Name	P856	P858	Px60	P1262	P1264	P1266	P1268	P1270
1st production	1997	1999	2001	2003	2005	2007	2009	2011
Process	250 nm	180 nm	130 nm	90 nm	65 nm	45 nm	32 nm	22 nm
Wafer size	200	200	200/300	300	300	300	300	300
Inter-connect	Al	Al	Cu	Cu	Cu	Cu	Cu	?
Metal layers	5	6	6	7	8	9	?	?
Channel	Si	Si	Si	Strained Si	Strained Si	Strained Si	Strained Si	Strained Si
Gate Dielectric	SiO_2	SiO_2	SiO_2	SiO_2	SiO_2	High – k	High – k	High – k
Gate electrode	Poly-Si	Poly-Si	Poly-Si	Poly-Si	Poly-Si	Metal	Metal	Metal
Lithography	248 nm	248 nm	248 nm	193 nm	193 nm	193 nm	EUV 13.4 nm	EUV 13.4 nm

(Subject to change)

Manufacturing process details from 1997 to 2011



Fuente: Intel Corporation

La Ley de Moore

- Microelectrónica + Microarquitectura

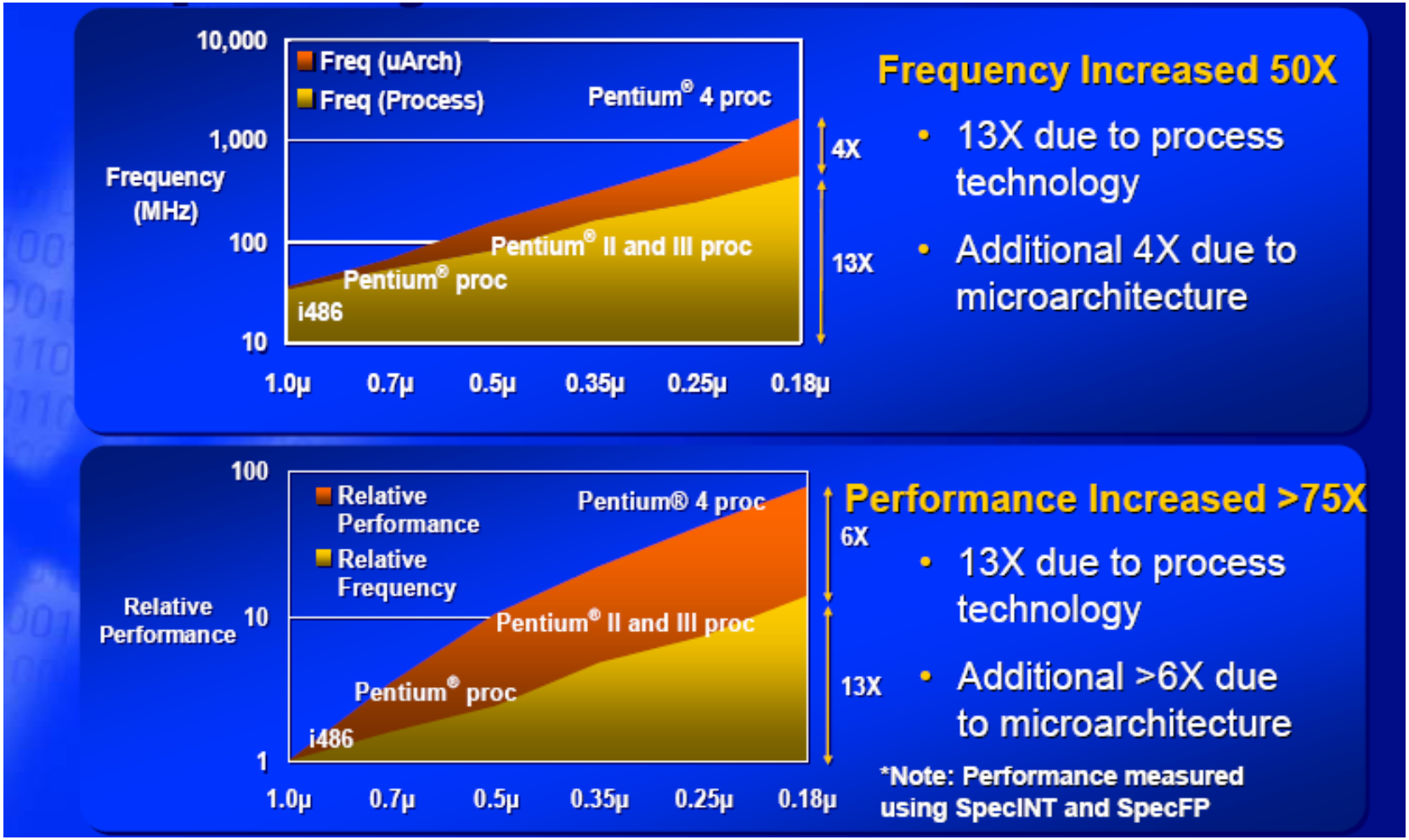
- Una industria con un progreso que no tiene equivalente

- Doblado cada 18 meses (1982-2000):
  - Total de incremento 3,200X
  - Los coches viajarían a 176,000 MPH; y recorrerían 64,000 millas/gal.
  - El viaje: L.A. a N.Y. en 5.5 seg (MACH 3200)

- Doblado cada 24 meses (1971-2001):
  - total de incremento 36,000X
  - Los coches viajarían a 2,400,000 MPH; y recorrerían 600,000 millas/gal.
  - El viaje: L.A. a N.Y. en 0.5 seg (MACH 36,000)

La Ley de Moore

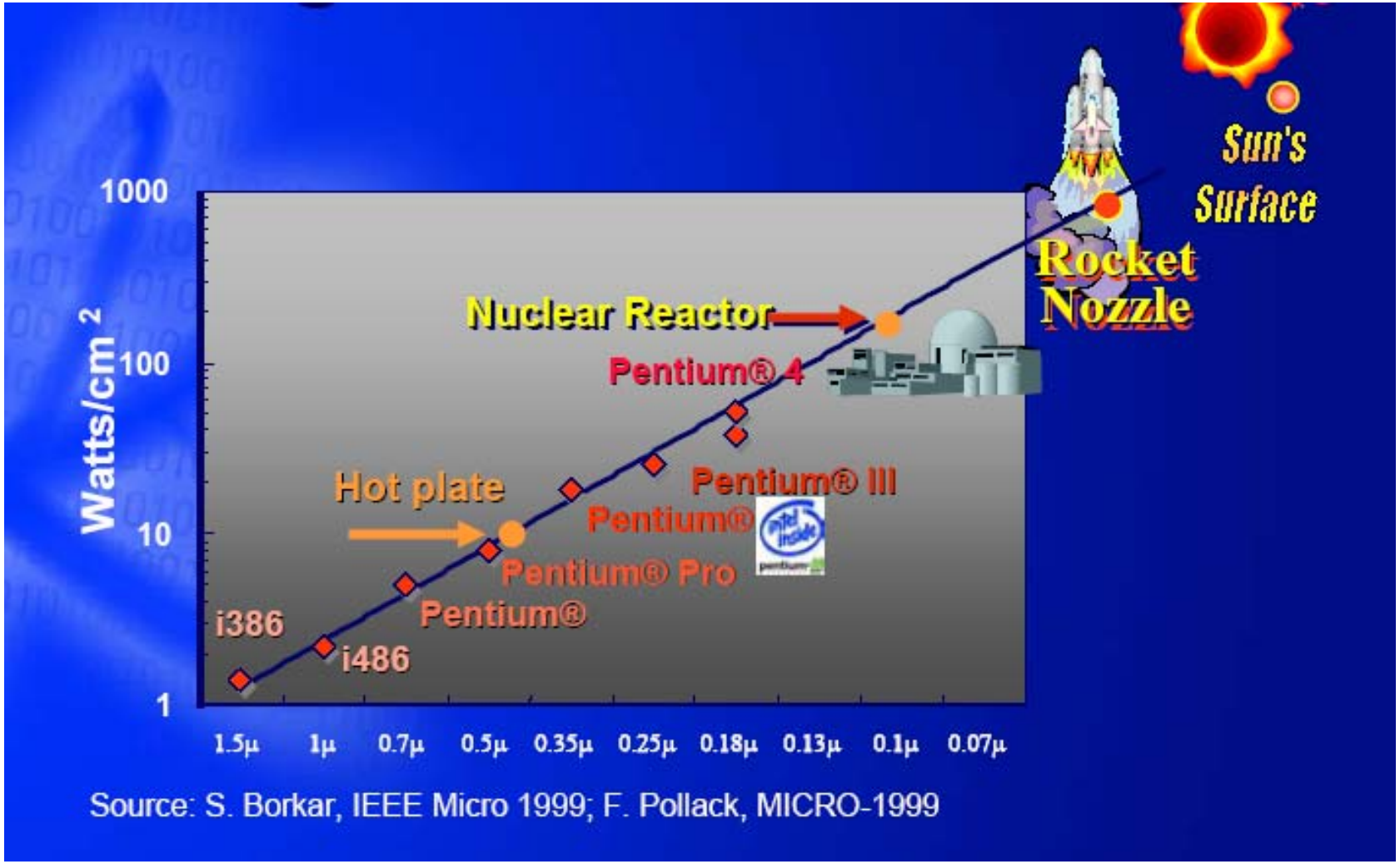
❑ Microelectrónica y microarquitectura



50X in frequency and 75X in performance

La Ley de Moore

❑ Densidad de potencia



El entorno: tendencias

Resumen de evolución en tecnología de implementación

	Capacidad	Velocidad Latencia
Logica	X2 en 3 años	X2 en 3 años
DRAM	X4 en 3 años	X2 en 10 años
Disco	X2 en 3 años	X2 en 10 años

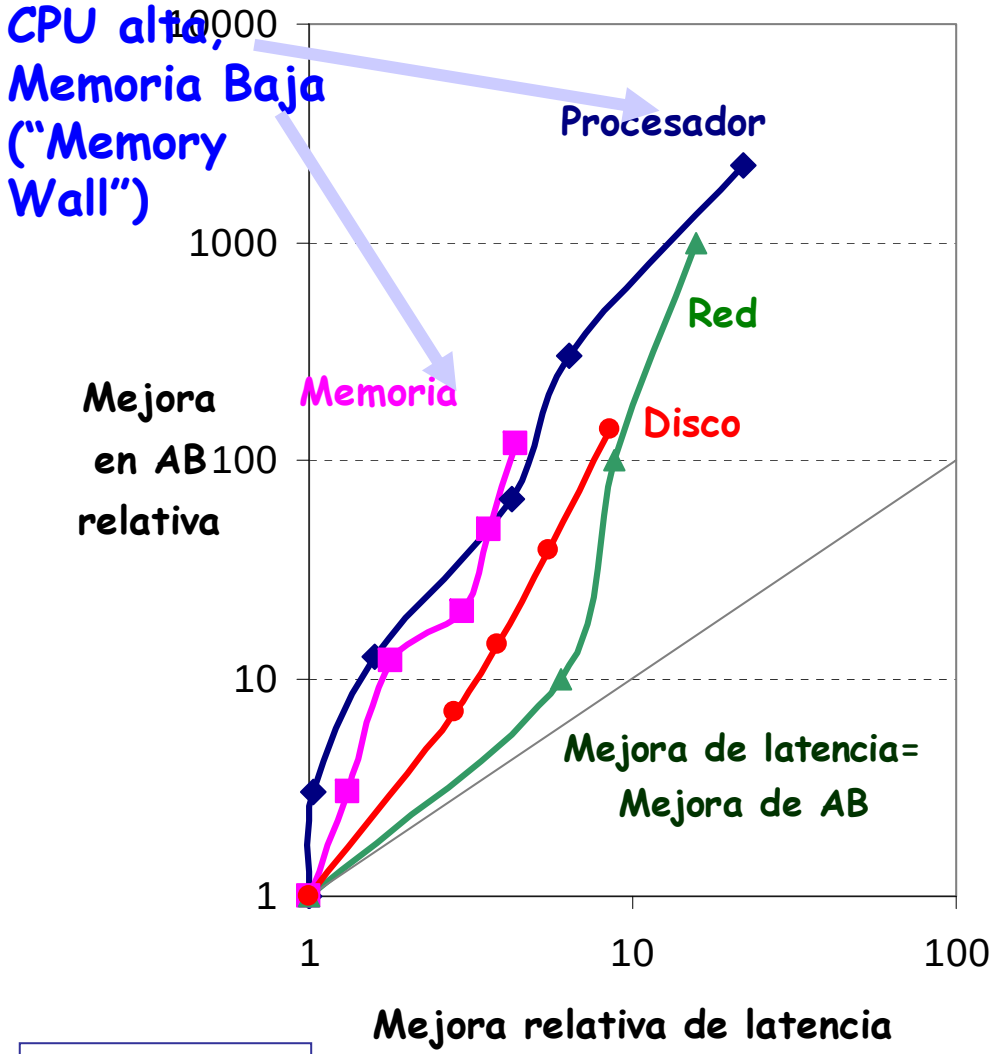
Uso de los computadores

- ✓ La cantidad de memoria necesaria crece entre 1.5 y 2 por año. Más bits para direccionamiento.
- ✓ Programación en LAN. Los compiladores son fundamentales, son el interfase entre las aplicaciones y el computador.

Una arquitectura debe ser diseñada para soportar el paso del tiempo
Cambios en tecnología, Sw y aplicaciones.
Arquitectura IBM360-390 (1964) ,X86 (1978)

El entorno: tendencias

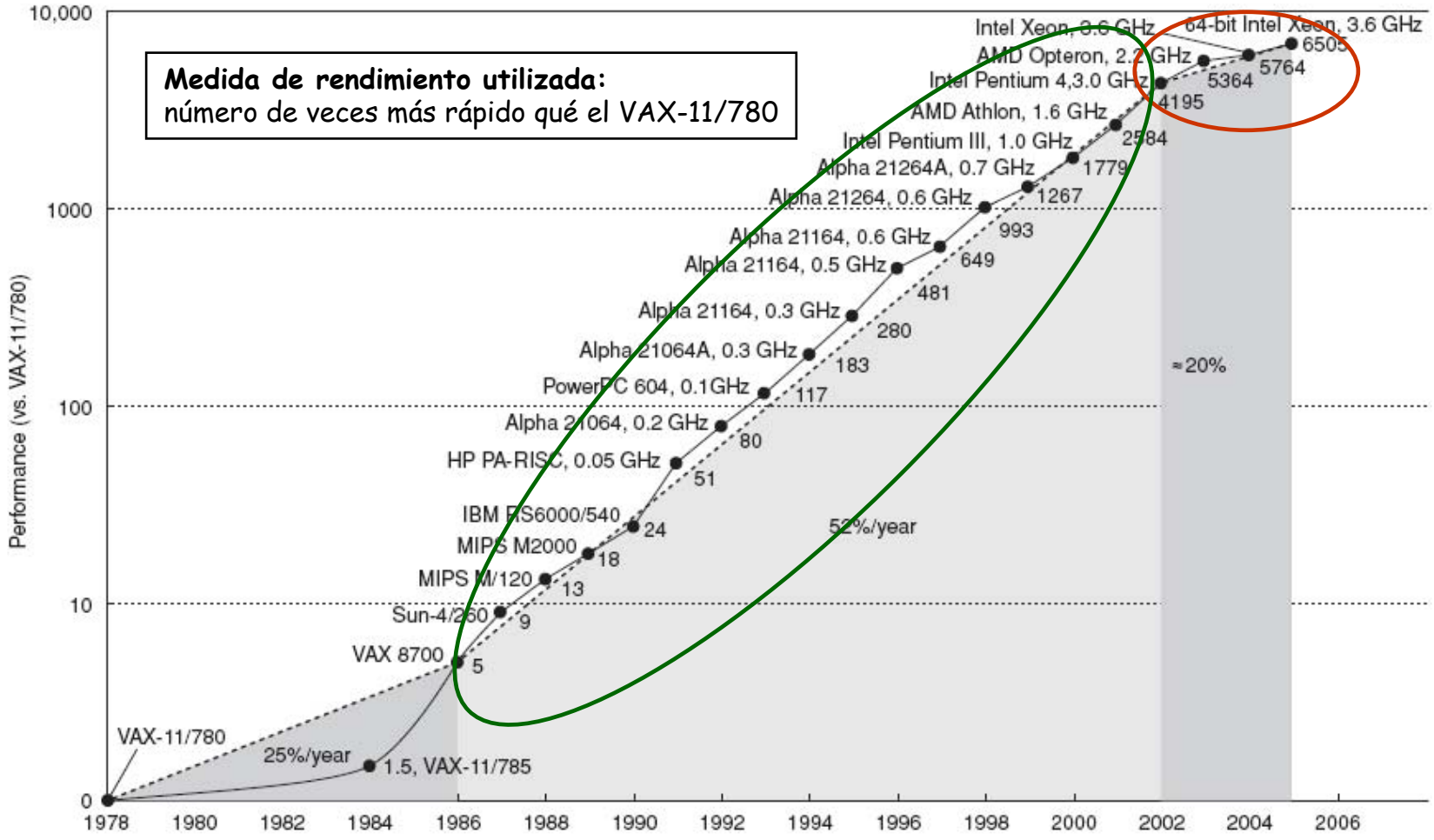
- Latencia y ancho de banda en los últimos 20 años



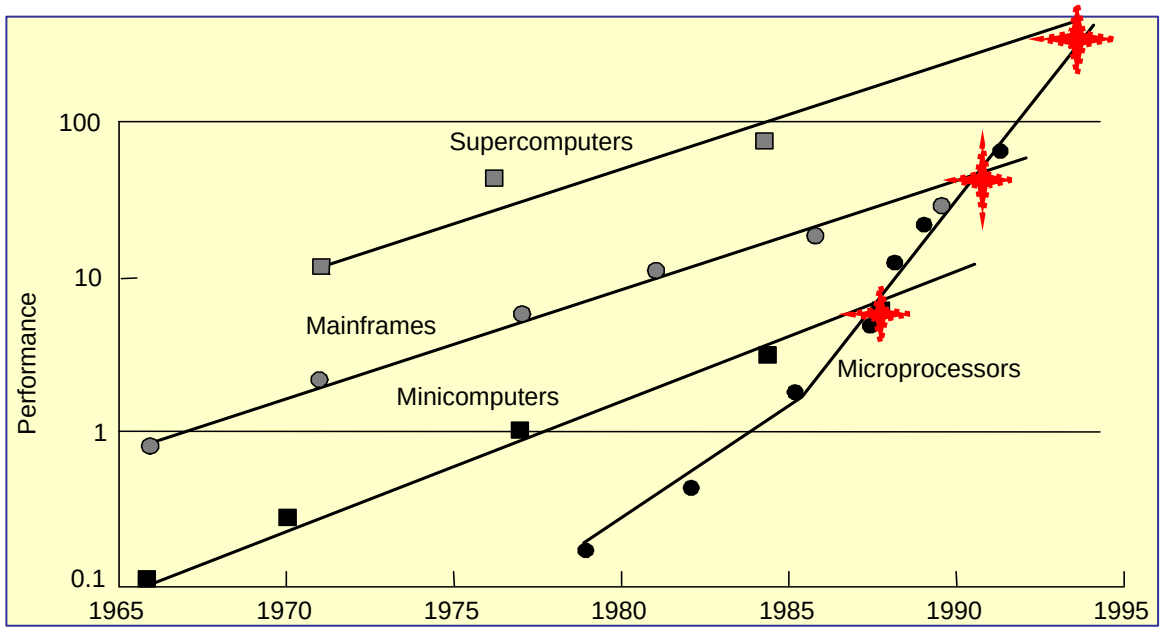
- Procesador: '286, '386, '486, Pentium, Pentium Pro, Pentium 4 (21x,2250x)
- Ethernet: 10Mb, 100Mb, 1000Mb, 10000 Mb/s (16x,1000x)
- Modulo de Memoria: 16bit plain DRAM, Page Mode DRAM, 32b, 64b, SDRAM, DDR SDRAM (4x,120x)
- Disco : 3600, 5400, 7200, 10000, 15000 RPM (8x, 143x)

Rendimiento

- Evolución del rendimiento de los procesadores



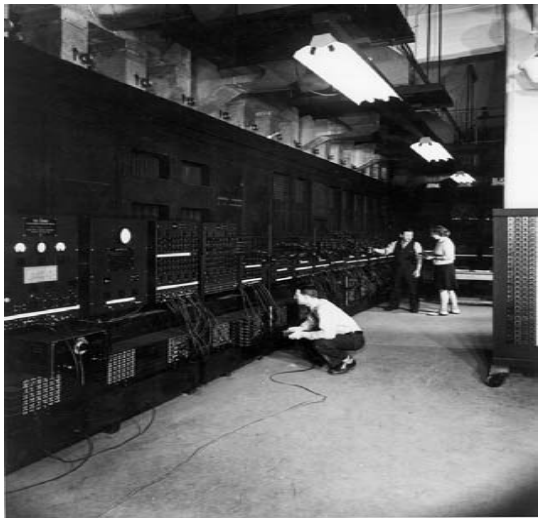
Evolución

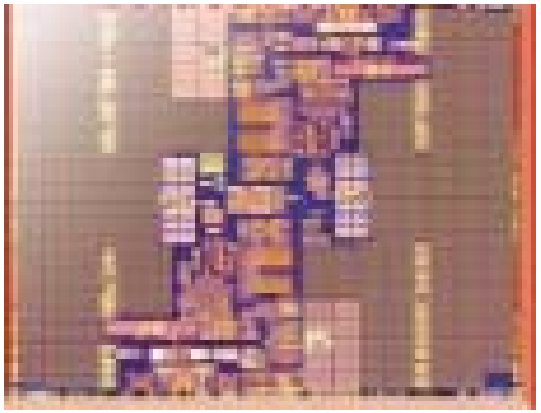


IBM Power2 Ws (1993) más rendimiento que Cray YMP(1988). Costo 50 veces menor

Desktop (precio)
Servidores (disponibilidad)
Servidores HPC (Rendimiento)
Sistemas empotrados
(memoria y consumo)

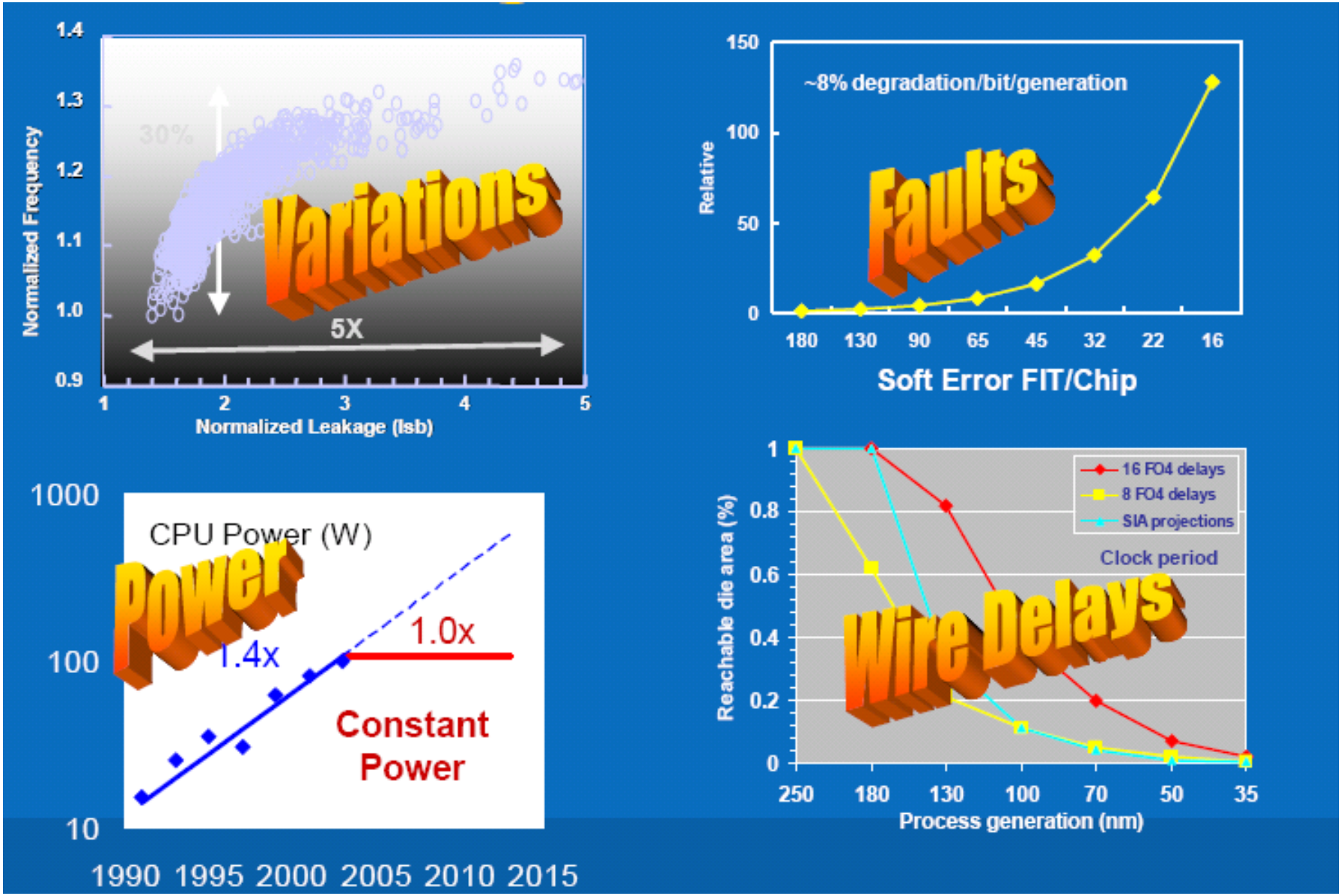
Eniac 1946





Itaniun 9000 Montecito
1700 Mtrans
90 nm, 100w
12MB por core
2 thread-2 cores

Retos 2015


Normalized Frequency
1.4
1.3
1.2
1.1
1.0
0.9
30%
Variations
5X
1
2
3
4
5
Normalized Leakage (lsb)
Relative
150
100
50
0
~8% degradation/bit/generation
Faults
180
130
90
65
45
32
22
16
Soft Error FIT/Chip
1000
100
10
CPU Power (W)
Power
1.4x
1.0x
Constant
Power
1990 1995 2000 2005 2010 2015
Reachable die area (%)
1
0.8
0.6
0.4
0.2
0
16 FO4 delays
8 FO4 delays
SIA projections
Clock period
Wire Delays
250
180
130
100
70
50
35
Process generation (nm)

Rendimiento

❑ Dos conceptos clave

Avión	Wa a París	Velocidad	Pasajeros	Throughput (p.km/h)
Boeing 747	6.5 horas	970 km/h	470	455900
Concorde	3 horas	2160 km/h	132	285120

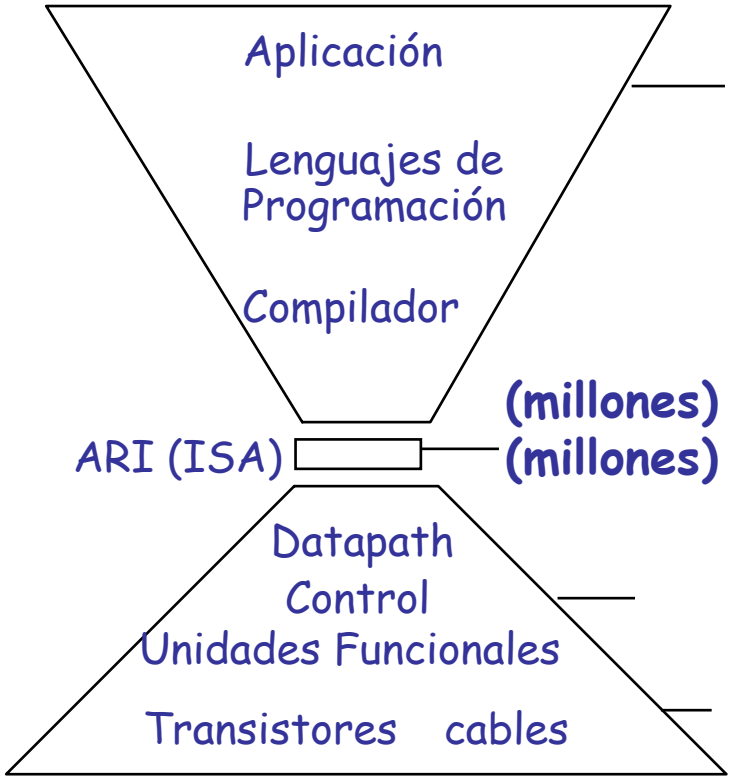
- ✓Tiempo de Ejecución (TEj) : Tiempo que tarda en completarse una tarea
  - ✓ (Tiempo de respuesta, latencia)
- ✓Rendimiento (Performance, Throughput) : tareas por hora, día ,…
- ✓ "X es n veces más rápido que Y" significa

$$\frac{TEj(Y)}{TEj(X)} = \frac{Performance(X)}{Performance(Y)} = n$$

- ✓ Reducir el TEj incrementa el rendimiento

Rendimiento

- Medidas del rendimiento



Respuestas por mes, hora, segundo
Operaciones por segundo TPC

(millones) de Instrucciones por segundo: MIPS
(millones) de (FP) operaciones por segundo: MFLOP/s

Megabytes por segundo

Ciclos por segundo (frecuencia de reloj)

La única medida fiable es el tiempo de ejecución programas reales
Dos aspectos: Rendimiento del computador, Rendimiento del procesador

Rendimiento

- Rendimiento del procesador

$$T_{CPU} = N * CPI * t$$

  - ✓ N: nº de instrucciones (Compiladores y LM)
  - ✓ CPI: (LM, implementación, paralelismo)
  - ✓ t: período de reloj (implementación, tecnología)

- Ciclos medios por instrucción (CPI)

CPI = (T_{CPU} * Frecuencia de reloj) / Numero de Instrucciones
= Ciclos / Numero de Instrucciones

Si asumimos que existen n tipos de instrucciones:

$$T_{CPU} = t * \sum_{j=1}^{n} (CPI_j * I_j)$$ *(n = tipos de instr., I_j = nº instr. tipo j ejecutadas)*

$$CPI = \sum_{j=1}^{n} CPI_j * F_j$$ (donde *Fj* es la frecuencia de aparición de la instrucción tipo J)

Ejemplo : ALU 1 ciclo(50%), Ld 2 ciclos(20%), St 2 ciclos(10%), saltos 2 ciclos(20%)
CPI: ALU 0.5, Ld 0.4, St 0.2, salto 0.4 TOTAL CPI = 1.5

Invertir recursos donde se gasta el tiempo

Rendimiento global del computador : Benchmarks

- La única forma fiable es ejecutando distintos programas reales.
  - Programas "de juguete": 10~100 líneas de código con resultado conocido. *Ej:: Criba de Erastótenes, Puzzle, Quicksort*
  - Programas de prueba (*benchmarks*) sintéticos: simulan la frecuencia de operaciones y operandos de un abanico de programas reales. *Ej:: Whetstone, Dhrystone*
- Programas reales típicos con cargas de trabajo fijas (actualmente la medida más aceptada) SPEC
- Otros
  - HPC:LINPACK, SPEChpc96, Nas Parallel Benchmark
  - Servidores: SPECweb, SPECSFS(File servers), TPC-C
  - Graficos: SPECviewperf(OpenGL), SPECapc(aplicaciones 3D)
  - Winbench, EEMBC

Rendimiento global del computador : SPEC

- Programas reales típicos con cargas de trabajo fijas (actualmente la medida más aceptada)
  - **SPEC89**: 10 programas proporcionando un único valor.
  - **SPEC92**: 6 programas enteros (SPECint92) y 14 en punto flotante (SPECfp92).Sin limites en opciones de compilación
  - **SPEC95**: 8 programas enteros (SPECint95) y 10 en punto flotante (SPECfp95). Dos opciones en compilación: la mejor para cada programa y la misma en todos (base)
  - **SPEC2000** 12 programas enteros y 14 en punto flotante. Dos opciones de compilación (la mejor: spec--, la misma spec--_base
  - **SPEC2006** 12 programas enteros y 17 en punto flotante. Dos opciones de compilación (la mejor: spec--, la misma spec--_base

Rendimiento

- SPEC2006 vesus SPEC2000

  Evolución de la jerarquía de memoria (256KB, 256MB a 4MB, 1GB)
  Más programas más complejos

Benchmark Description	CPU2000			CPU2006		
	Integer	Lng	RT	Integer	Lng	RT
GNU C compiler	176.gcc	C	1,100	403.gcc	C	8,050
Manipulates strings & prime numbers in Perl language	253.perlbmk	C	1,800	400.perlbench	C	9,766
Minimum cost network flow solver (combinatorial optimization)	181.mcf	C	1,800	429.mcf	C	9,120
Data compression utility	256.bzip2	C	1,500	401.bzip2	C	9,644
Data compression utility	164.gzip	C	1,400			
Video compression & decompression				464.h264ref	C	22,235
Artificial intelligence, plays game of Chess	186.crafty	C	1,000	458.sjeng	C	12,141
Artificial intelligence, plays game of Go				445.gobmk	C	10,489
Artificial intelligence used in games for finding 2D paths across terrains				473.astar	C++	7,017
Natural language processing	197.parser	C	1,800			
XML processing				483.xalancbmk	C++	6,869
FPGA circuit placement and routing	175.vpr	C	1,400			
EDA place and route simulator	300.twolf	C	3,000			
Search gene sequence				456.hmmer	C	9,333
Ray tracing	252.eon	C++	1,300			
Computational group theory	254.gap	C	1,100			
Database program	255.vortex	C	1,900			
Library for simulating a quantum computer				462.libquantum	C	20,704
Discrete event simulation				471.omnetpp	C++	6,270
	hours	5.3	19,100	hours	36.6	131,638

Rendimiento

- SPEC2006 vesus SPEC2000

Benchmark Description	CPU2000			CPU2006		
	Floating Pnt	Lng	RTime	Floating Pnt	Lng	RTime
Weather prediction, shallow water model	171.swim	F77	3,100			
Velocity & distribution of pollutants based on temperature, wind	301.apsi	F77	2,600			
Weather modeling (30km area over 2 days)				481.wrf	C/F	11,215
Physics, particle accelerator model	200.sixtrack	F77	1,100			
Parabolic/elliptic partial differential equations	173.applu	F77	2,100			
Multi-grid solver in 3D potential field	172.mgrid	F77	1,800			
General relativity, solves Einstein evolution equations				436.cactusADM	C/F	11,927
Computational electromagnetics (solves Maxwell equations in 3D)				459.GemsFDTD	F	10,583
Quantum chromodynamics	168.wupwise	F77	1,600			
Quantum chromodynamics, gauge field generation with dynamical quarks				433.milc	C	9,180
Fluid dynamics, analysis of oscillatory instability	178.galgel	F90	2,900			
Fluid dynamics, computes 3D transonic transient laminar viscous flow				410.bwaves	F	13,592
Computational fluid dynamics for simulation of astrophysical phenomena				434.zeusmp	F	9,096
Fluid dynamics, large eddy simulations with linear-eddy model in 3D				437.leslie3d	F	9,358
Fluid dynamics, simulates incompressible fluids in 3D				470.lbm	C	13,718
Molecular dynamics (simulations based on newtonian equations of motion)				435.gromacs	C/F	7,132
Biomolecular dynamics, simulates large system with 92,224 atoms				444.namd	C++	8,018
Computational chemistry	188.ammp	C	2,200			
Quantum chemistry package (object-oriented design)				465.tonto	F	9,822
Quantum chemistry, wide range of self-consistent field calculations				416.gamess	F	19,575
Computer vision, face recognition	187.facerec	F90	1,900			
Speech recognition system				482.sphinx3	C	19,528
3D graphics library	177.mesa	C	1,400			
Neural network simulation (adaptive resonance theory)	179.art	C	2,600			
Earthquake modeling (finite element simulation)	183.equake	C	1,300			
Crash modeling (finite element simulation)	191.fma3d	F90	2,100			
Number theory (testing for primes)	189.lucas	F90	2,000			
Structural mechanics (finite elements for linear & nonlinear 3D structures)				454.calculix	C/F	8,250
Finite element analysis (program library)				447.dealII	C++	11,486
Linear programming optimization (railroad planning, airlift models)				450.soplex	C++	8,338
Image ray tracing (400x400 anti-aliased image with abstract objects)				453.povray	C++	5,346
	hours	8.0	28,700	hours	52	186,164

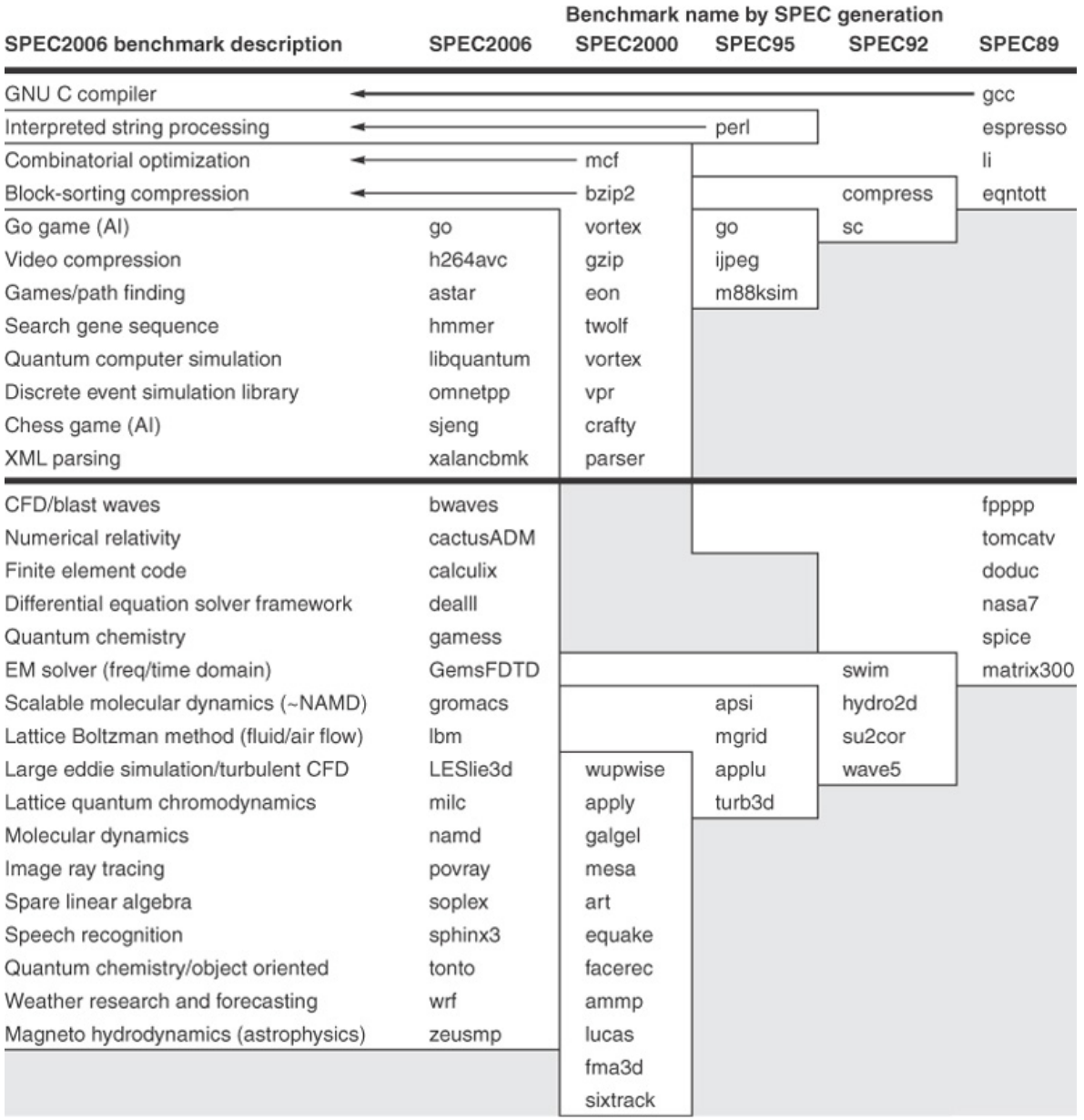
Rendimiento

➢ Evolución de los SPEC

Year	Iteration	Suites	Languages	Measures	Reference Machine
1989	SPEC CPU	10 SPEC programs RT: 18.66 hours (scores not rounded)	C(4) & Fortran(5) & C/Fortran(1)	SPECmark SPECthruput	Vax 11/780 5 MHz 8K cache off-chip memory N/A
1992	SPEC CPU92	6 CINT92 programs RT: 6.21 hours 14 CFP92 programs RT: 41.27 hours (scores rounded to 10s place)	C(6) C(2) & Fortran(12)	SPECint92 SPECfp92 SPECint_rate92 SPECfp_rate92	same as SPEC89
1995	SPEC CPU95	8 CINT95 programs RT: 5.25 hours 10 CFP95 programs RT: 11.00 hours (scores rounded to 100s place)	C(8) Fortran(10)	SPECint95 SPECint_base95 SPECfp95 SPECfp_base95 SPECint_rate95 SPECint_rate_base95 SPECfp_rate95 SPECfp_rate_base95	SPARCstation 10/40 40 MHz SuperSPARC I 20K/16K I/D L1 on-chip no L2 cache 128MB memory
2000	SPEC CPU2000	12 CINT2000 programs RT: 5.31 hours 14 CFP2000 programs RT: 7.97 hours (scores rounded to 100s place)	C(11) & C++(1) C(4) & Fortran77(6) & Fortran90(4)	same set of 8 measures defined for SPEC CPU95	Ultra 5 model 10 300 MHz UltraSPARC IIi 16K/16K I/D L1 on-chip 2MB L2 cache off-chip 256MB memory
2006	SPEC CPU2006	12 CINT2006 programs RT: 36.57 hours 17 CFP2006 programs RT: 51.71 hours (scores not rounded)	C(9) & C++(3) C(3) & C++(4) & Fortran(6) & C/Fortran(4)	same set of 8 measures defined for SPEC CPU95	Ultra Enterprise 2 296 MHz UltraSPARC II 16K/16K I/D L1 on-chip 2MB L2 cache off-chip 1GB memory

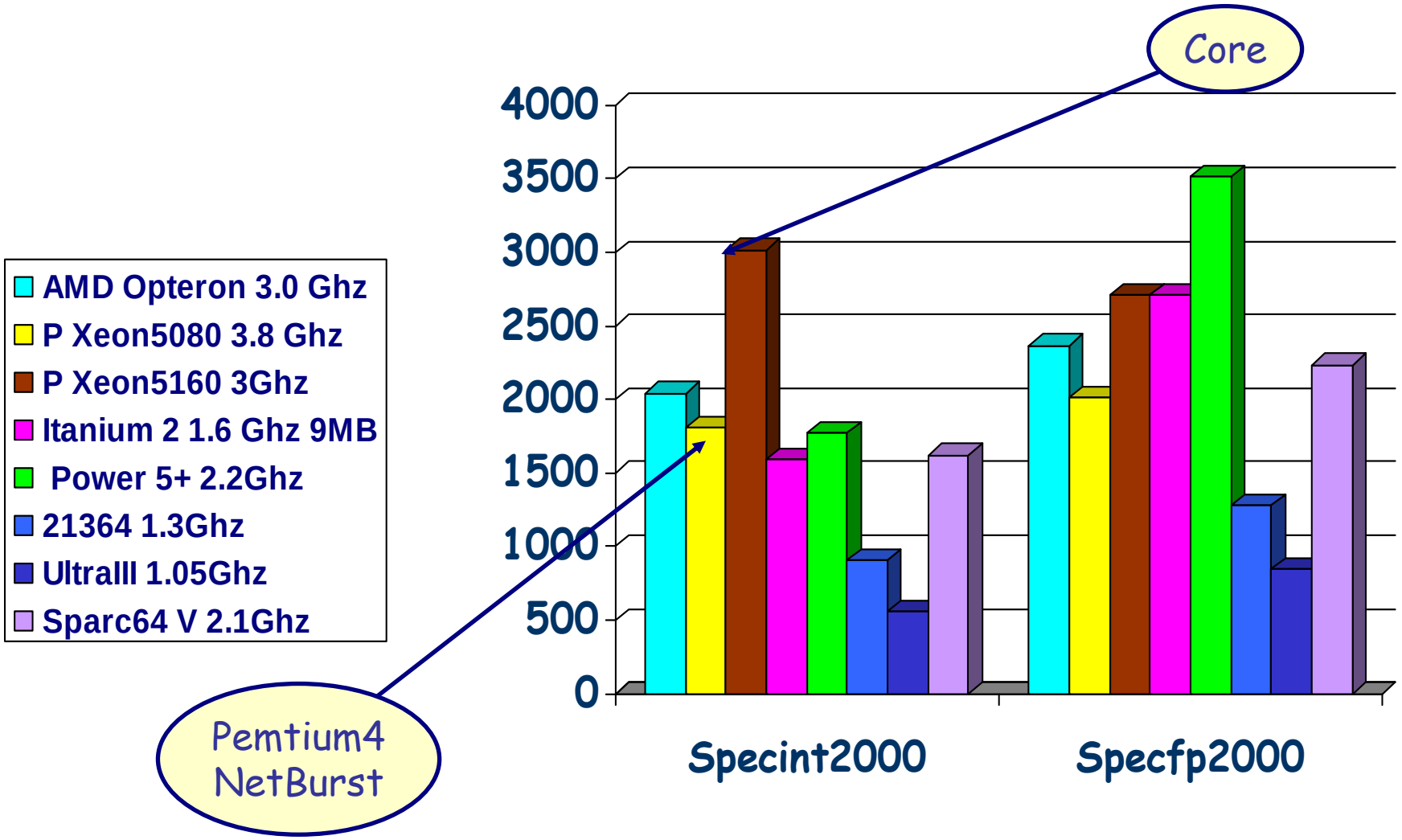
Rendimiento

Evolución de los Spec

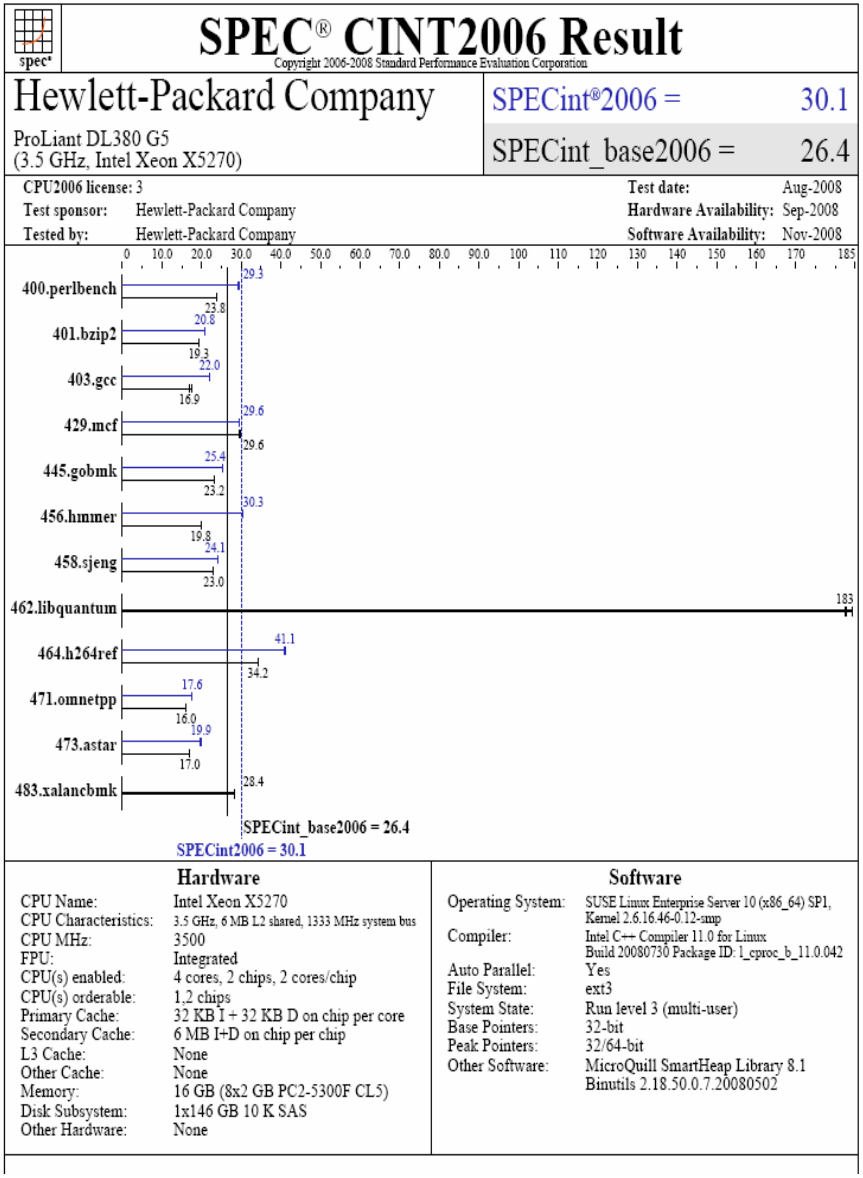


SPEC2006 benchmark description	Benchmark name by SPEC generation: SPEC2006	SPEC2000	SPEC95	SPEC92	SPEC89
GNU C compiler					gcc
Interpreted string processing			perl		espresso
Combinatorial optimization		mcf			li
Block-sorting compression		bzip2		compress	eqntott
Go game (AI)	go	vortex	go	sc	
Video compression	h264avc	gzip	ijpeg		
Games/path finding	astar	eon	m88ksim		
Search gene sequence	hmmer	twolf			
Quantum computer simulation	libquantum	vortex			
Discrete event simulation library	omnetpp	vpr			
Chess game (AI)	sjeng	crafty			
XML parsing	xalancbmk	parser			
CFD/blast waves	bwaves				fpppp
Numerical relativity	cactusADM				tomcatv
Finite element code	calculix				doduc
Differential equation solver framework	dealII				nasa7
Quantum chemistry	gamess				spice
EM solver (freq/time domain)	GemsFDTD			swim	matrix300
Scalable molecular dynamics (~NAMD)	gromacs		apsi	hydro2d	
Lattice Boltzman method (fluid/air flow)	lbm		mgrid	su2cor	
Large eddie simulation/turbulent CFD	LESlie3d	wupwise	applu	wave5	
Lattice quantum chromodynamics	milc	apply	turb3d		
Molecular dynamics	namd	galgel			
Image ray tracing	povray	mesa			
Spare linear algebra	soplex	art			
Speech recognition	sphinx3	equake			
Quantum chemistry/object oriented	tonto	facerec			
Weather research and forecasting	wrf	ammp			
Magneto hydrodynamics (astrophysics)	zeusmp	lucas			
		fma3d			
		sixtrack			

- SPEC de los últimos procesadores (SPEC2000)
  - Retirados febrero 2007



Rendimiento



SPEC® CINT2006 Result

Copyright 2006-2008 Standard Performance Evaluation Corporation

Hewlett-Packard Company

ProLiant DL380 G5
(3.5 GHz, Intel Xeon X5270)

SPECint®2006 = 30.1

SPECint_base2006 = 26.4

CPU2006 license: 3
Test sponsor: Hewlett-Packard Company
Tested by: Hewlett-Packard Company

Test date: Aug-2008
Hardware Availability: Sep-2008
Software Availability: Nov-2008

400.perlbench 29.3 23.8
401.bzip2 20.8 19.3
403.gcc 22.0 16.9
429.mcf 29.6 29.6
445.gobmk 25.4 23.2
456.hmmer 30.3 19.8
458.sjeng 24.1 23.0
462.libquantum 183
464.h264ref 41.1 34.2
471.omnetpp 17.6 16.0
473.astar 19.9 17.0
483.xalancbmk 28.4

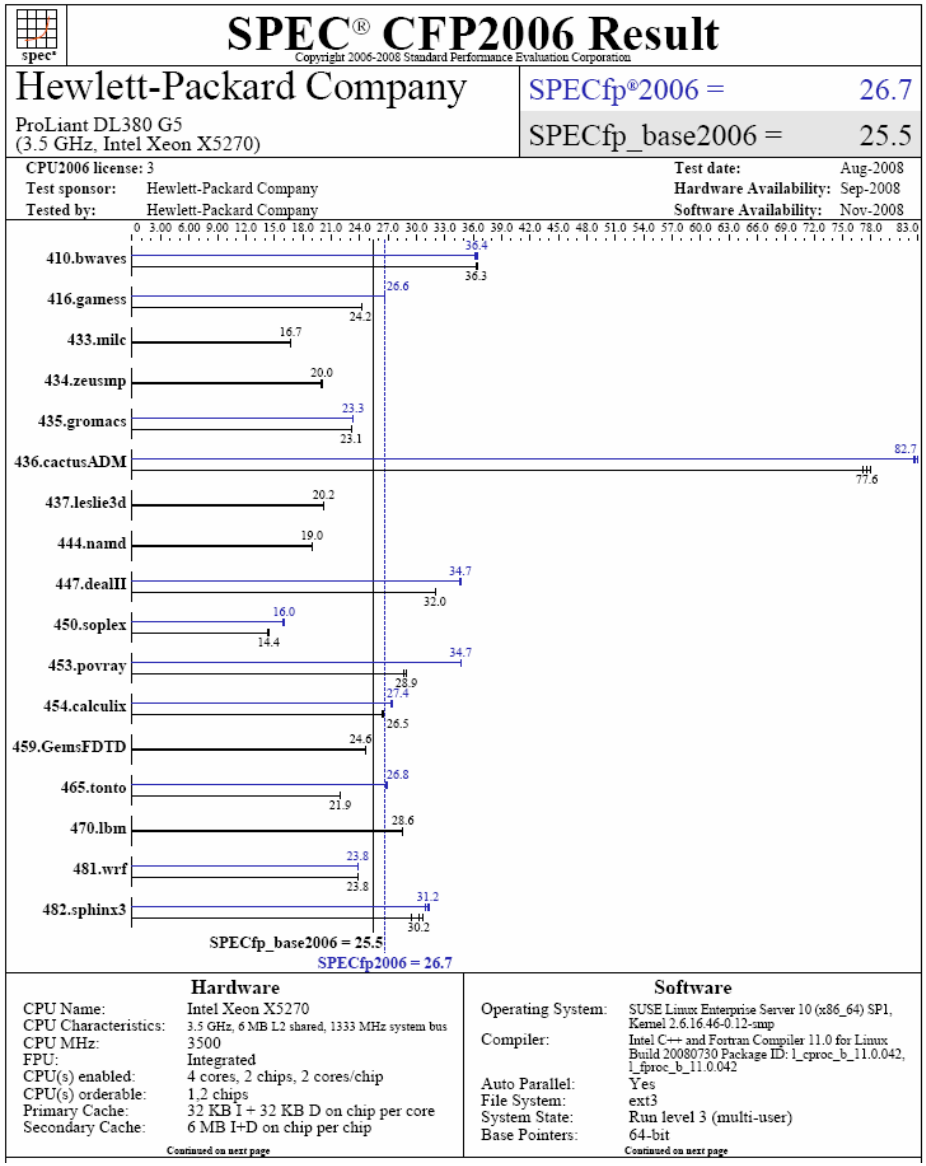
SPECint_base2006 = 26.4
SPECint2006 = 30.1

Hardware

CPU Name:	Intel Xeon X5270
CPU Characteristics:	3.5 GHz, 6 MB L2 shared, 1333 MHz system bus
CPU MHz:	3500
FPU:	Integrated
CPU(s) enabled:	4 cores, 2 chips, 2 cores/chip
CPU(s) orderable:	1,2 chips
Primary Cache:	32 KB I + 32 KB D on chip per core
Secondary Cache:	6 MB I+D on chip per chip
L3 Cache:	None
Other Cache:	None
Memory:	16 GB (8x2 GB PC2-5300F CL5)
Disk Subsystem:	1x146 GB 10 K SAS
Other Hardware:	None

Software

Operating System:	SUSE Linux Enterprise Server 10 (x86_64) SP1, Kernel 2.6.16.46-0.12-smp
Compiler:	Intel C++ Compiler 11.0 for Linux Build 20080730 Package ID: l_cproc_b_11.0.042
Auto Parallel:	Yes
File System:	ext3
System State:	Run level 3 (multi-user)
Base Pointers:	32-bit
Peak Pointers:	32/64-bit
Other Software:	MicroQuill SmartHeap Library 8.1 Binutils 2.18.50.0.7.20080502



SPEC® CFP2006 Result

Copyright 2006-2008 Standard Performance Evaluation Corporation

Hewlett-Packard Company

ProLiant DL380 G5
(3.5 GHz, Intel Xeon X5270)

SPECfp®2006 = 26.7

SPECfp_base2006 = 25.5

CPU2006 license: 3
Test sponsor: Hewlett-Packard Company
Tested by: Hewlett-Packard Company

Test date: Aug-2008
Hardware Availability: Sep-2008
Software Availability: Nov-2008

410.bwaves 36.4 36.3
416.gamess 26.6 24.2
433.milc 16.7
434.zeusmp 20.0
435.gromacs 23.3 23.1
436.cactusADM 82.7 77.6
437.leslie3d 20.2
444.namd 19.0
447.dealII 34.7 32.0
450.soplex 16.0 14.4
453.povray 34.7 28.9
454.calculix 27.4 26.5
459.GemsFDTD 24.6
465.tonto 26.8 21.9
470.lbm 28.6
481.wrf 23.8 23.8
482.sphinx3 31.2 30.2

SPECfp_base2006 = 25.5
SPECfp2006 = 26.7

Hardware

CPU Name:	Intel Xeon X5270
CPU Characteristics:	3.5 GHz, 6 MB L2 shared, 1333 MHz system bus
CPU MHz:	3500
FPU:	Integrated
CPU(s) enabled:	4 cores, 2 chips, 2 cores/chip
CPU(s) orderable:	1,2 chips
Primary Cache:	32 KB I + 32 KB D on chip per core
Secondary Cache:	6 MB I+D on chip per chip

Continued on next page

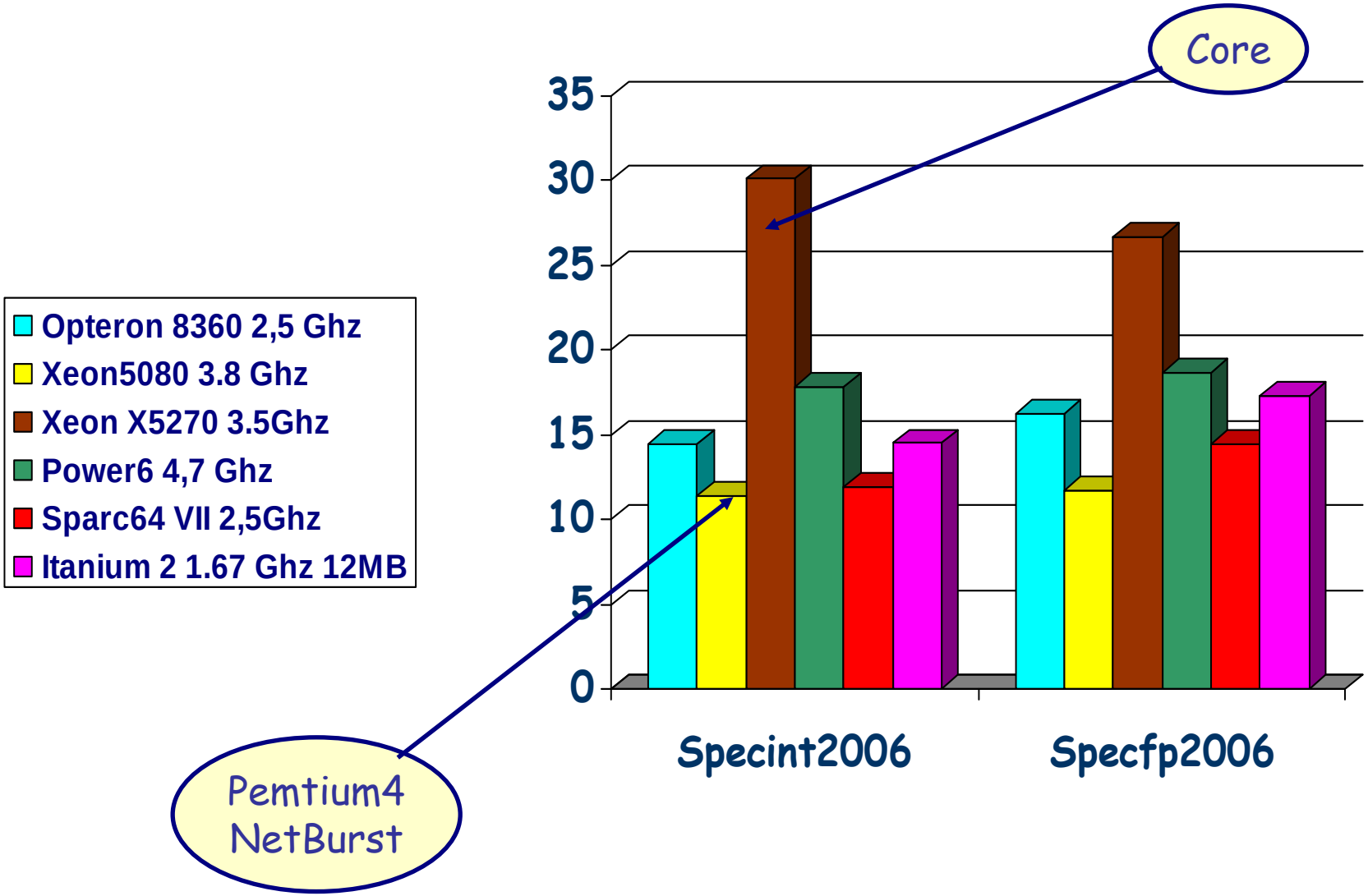
Software

Operating System:	SUSE Linux Enterprise Server 10 (x86_64) SP1, Kernel 2.6.16.46-0.12-smp
Compiler:	Intel C++ and Fortran Compiler 11.0 for Linux Build 20080730 Package ID: l_cproc_b_11.0.042, l_fproc_b_11.0.042
Auto Parallel:	Yes
File System:	ext3
System State:	Run level 3 (multi-user)
Base Pointers:	64-bit

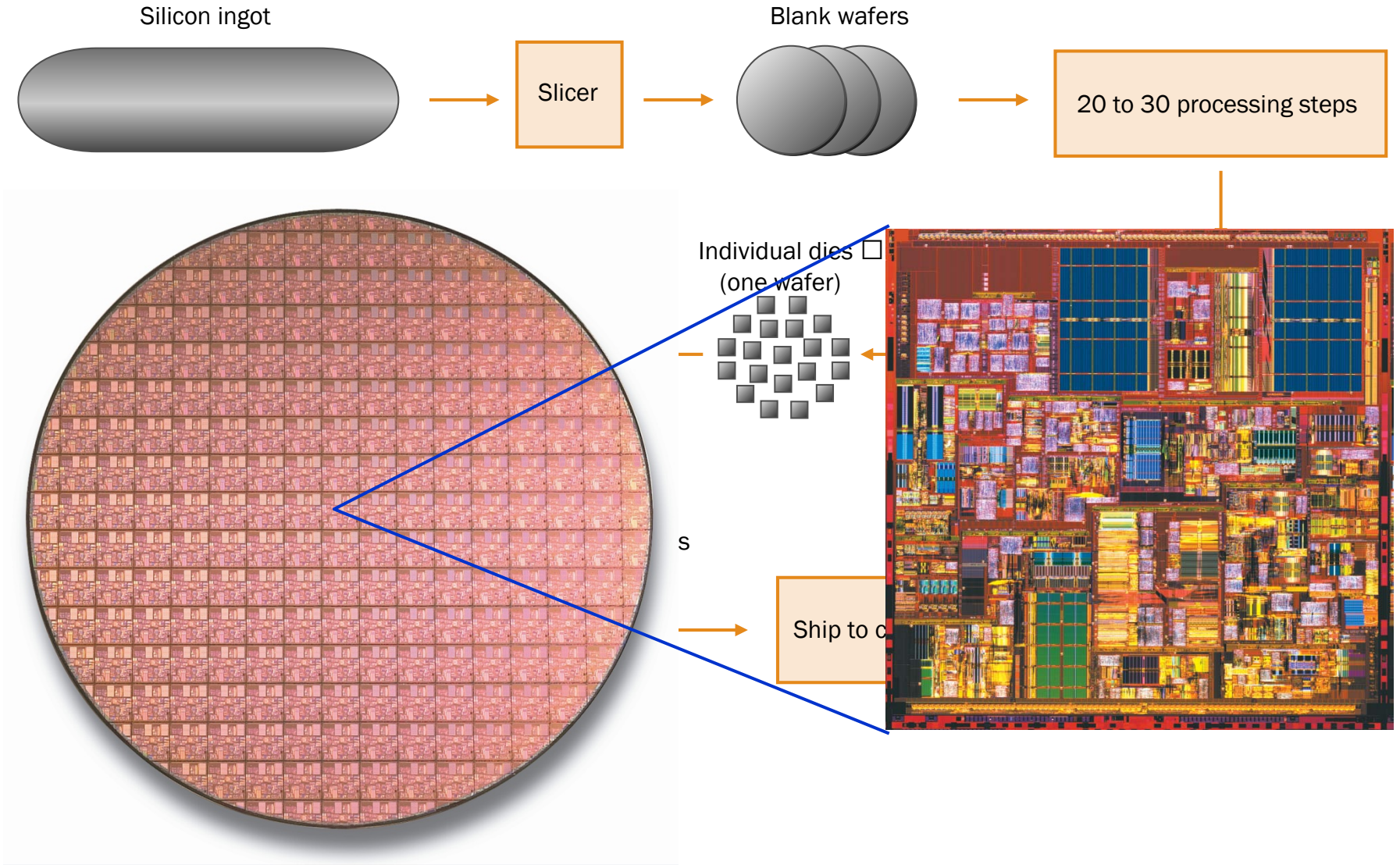
Continued on next page

Rendimiento

- SPEC de los últimos procesadores (SPEC2006)



Fabricación de un CI

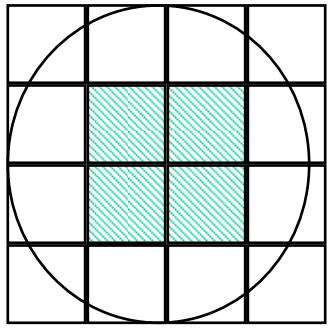


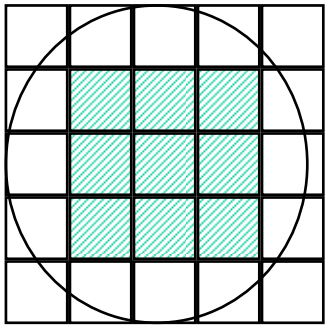
Coste

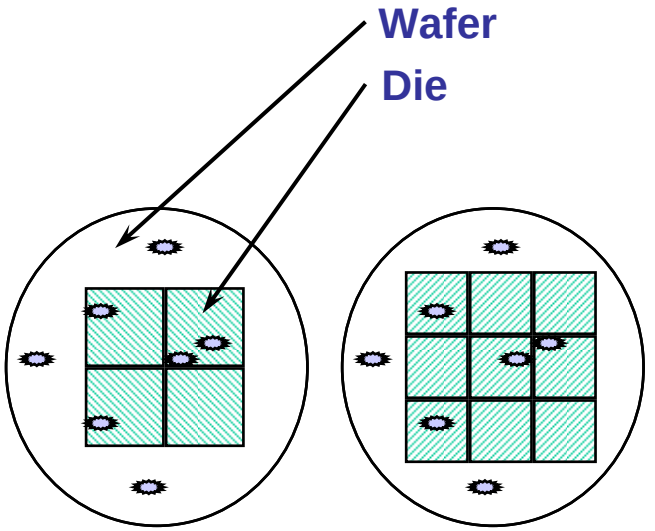
- Coste : El fundamental, el coste del CI

$$\text{coste de CI} = \frac{\text{Die coste} + \text{Testing coste} + \text{Packaging coste}}{\text{Final test yield}}$$

$$\text{Die coste} = \frac{\text{coste del Wafer}}{\text{Dies por Wafer} * \text{Die yield}}$$



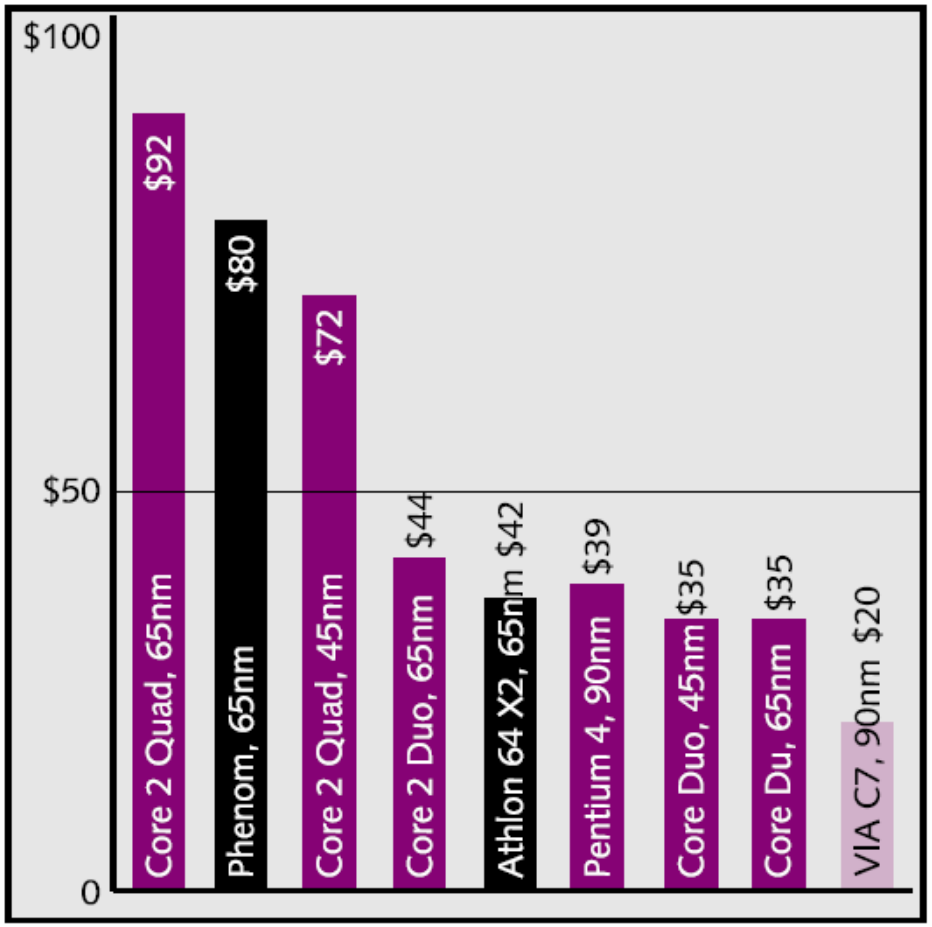




El costo de CI (Die) ≈ f(área del die)²

Coste

- Algunos ejemplos reales

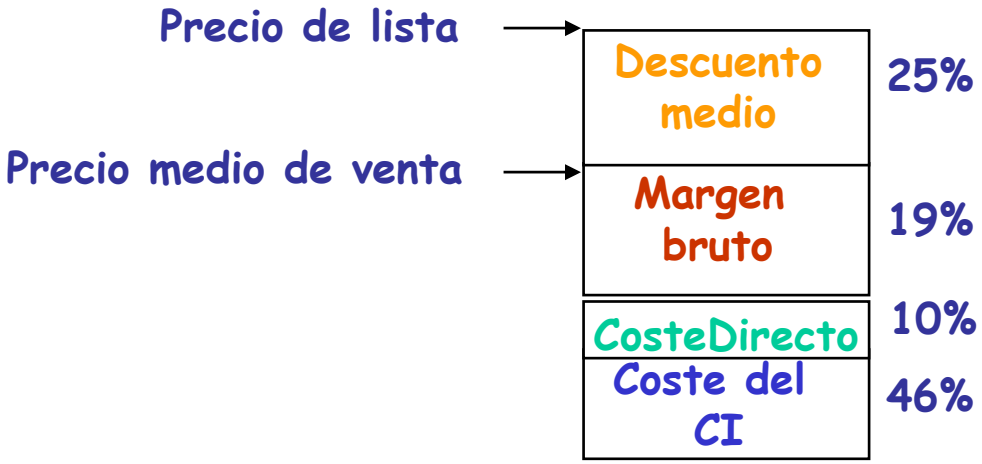


* Processor core change from previous process generation.
** Not mainstream processors, but provided for comparison purposes.

Fuente Microprocessor Report

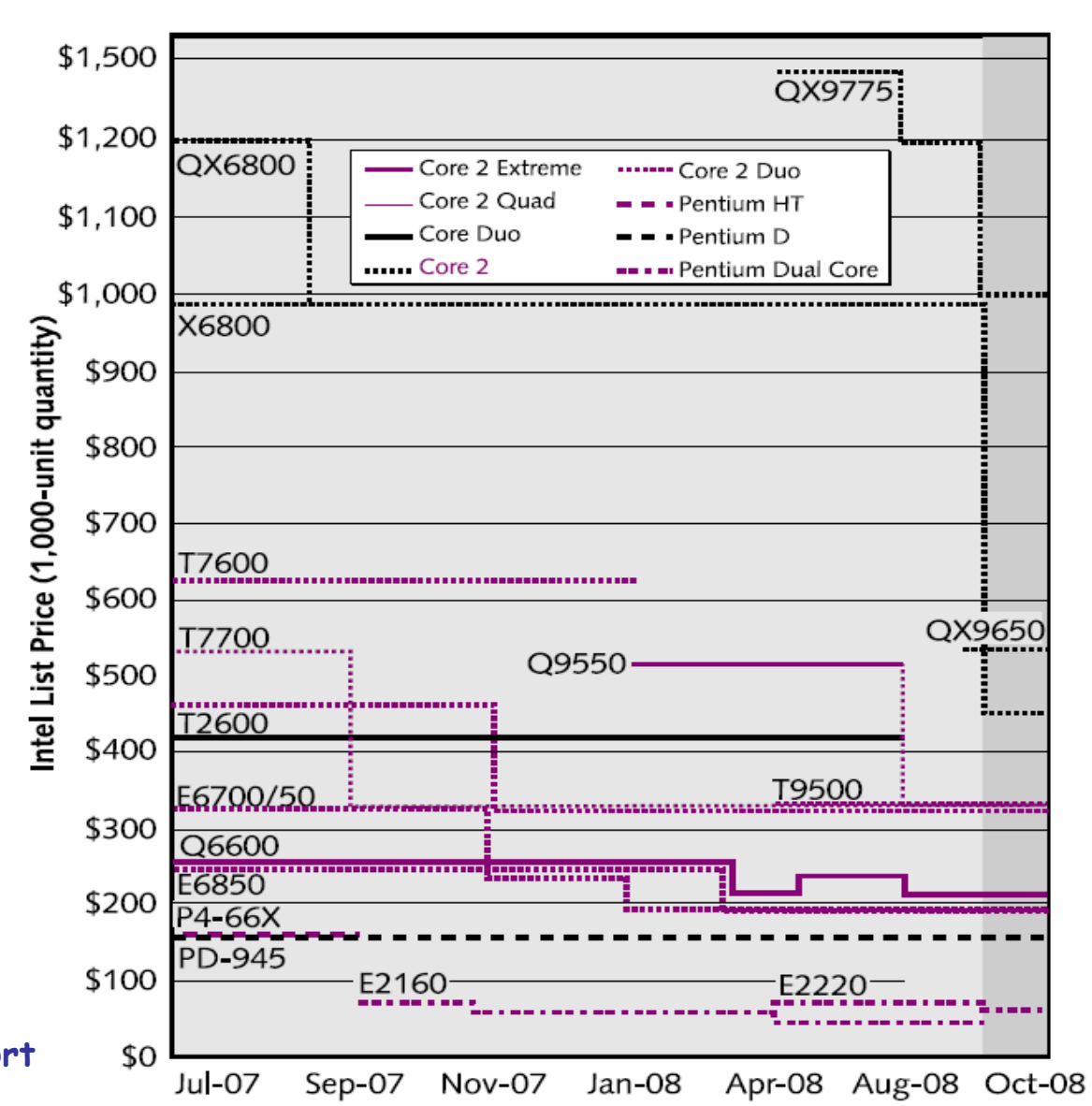
❑ Componentes del coste final (Precio)

- Coste del CI
- Costo Directo: costes recurrentes: mano de obra, compras,
- Margen bruto: costes no recurrentes, I&D, marketing, ventas, equipamiento, costes financieros, beneficio, impuestos
- Descuento



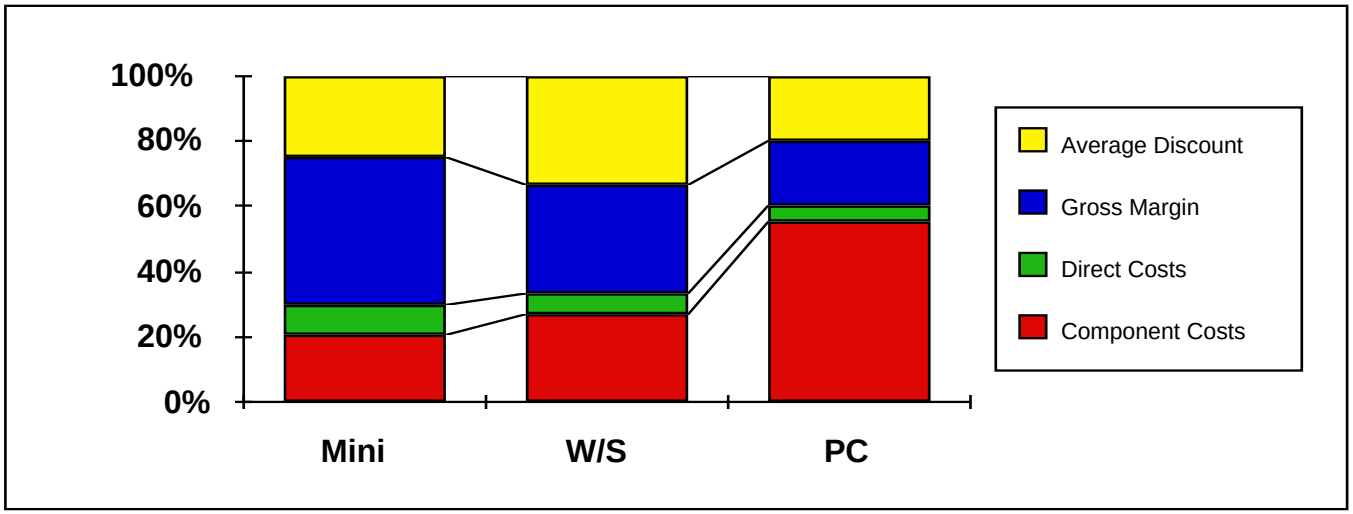
Coste

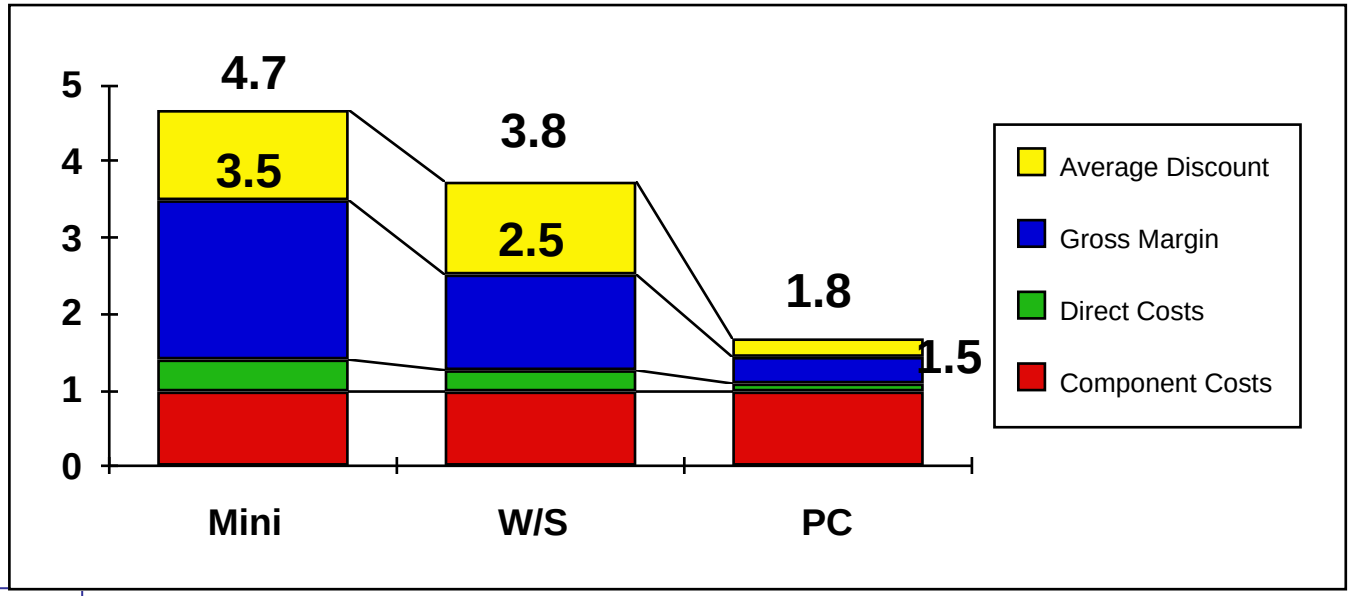
- Evolución en la vida comercial



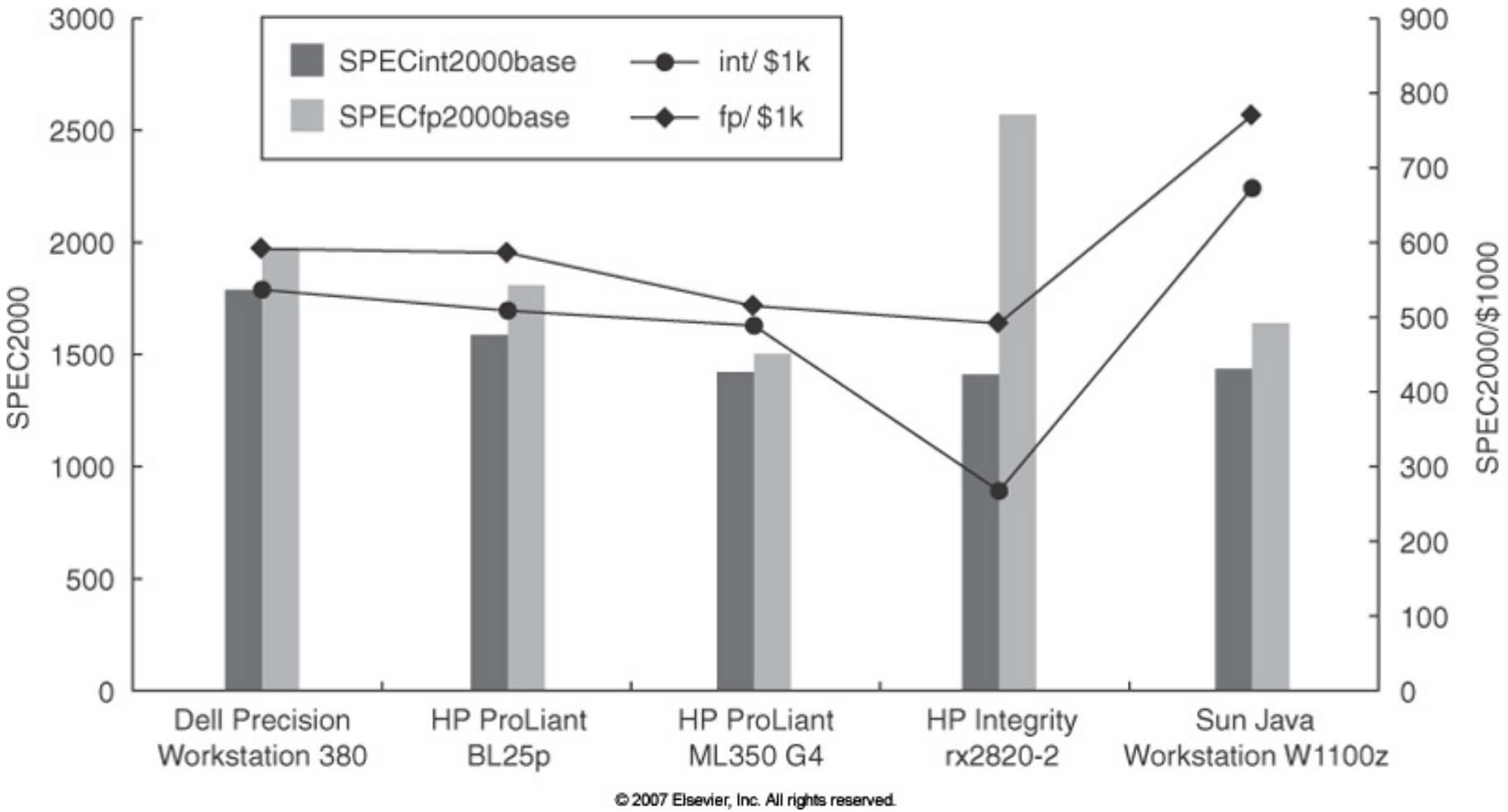
Fuente Microprocessor Report

❑ Diferentes segmentos de mercado

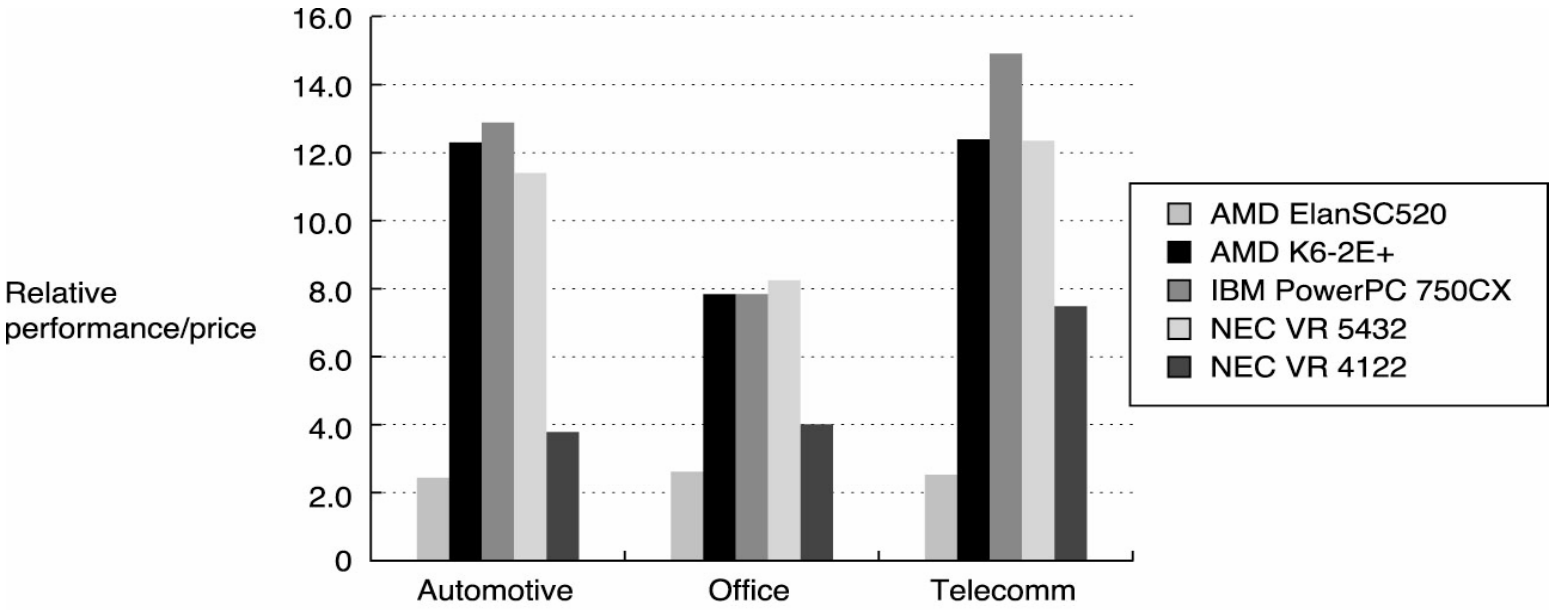


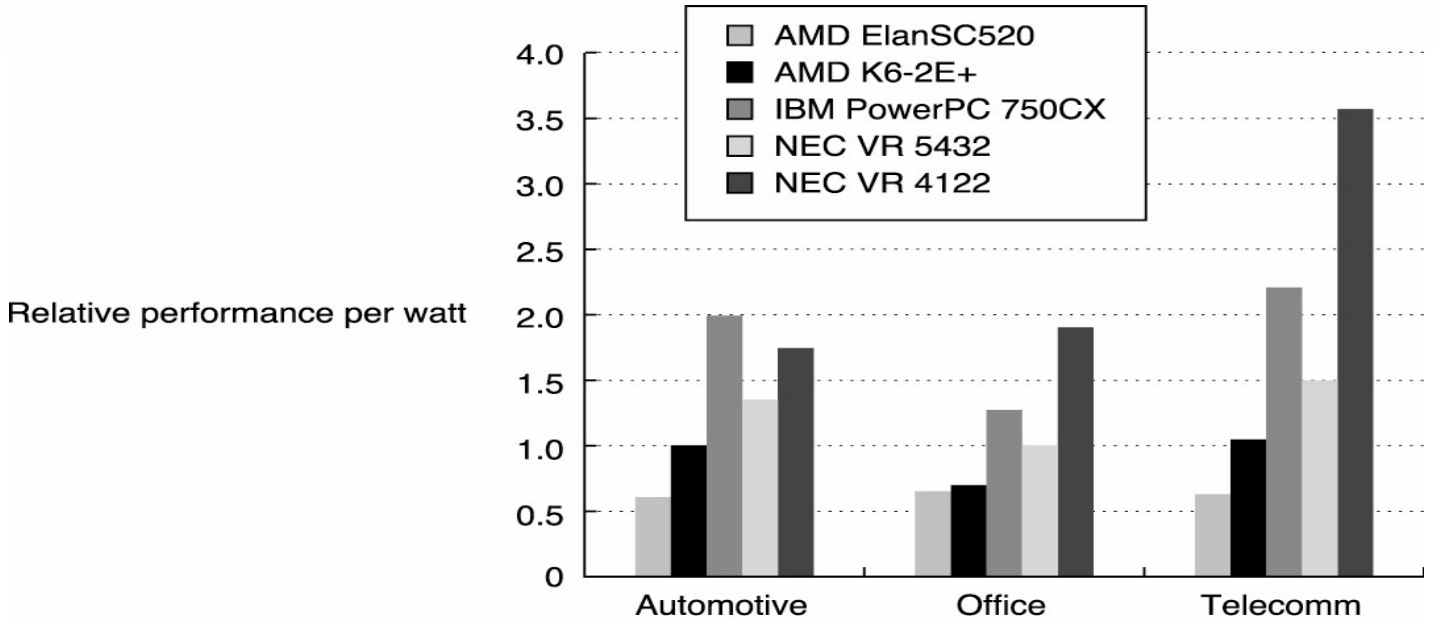


Coste-Rendimiento



Coste-Rendimiento-Consumo





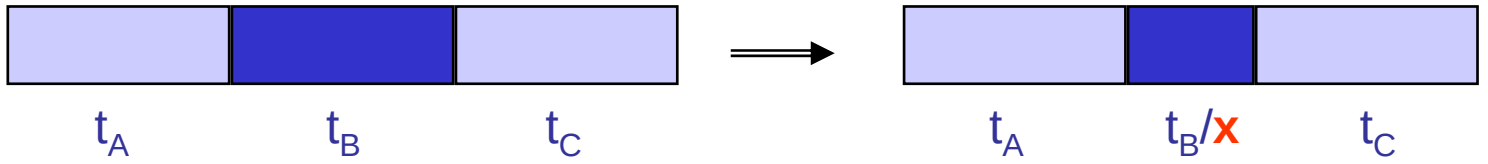
Un principio simple

- Un principio básico: Hacer rápidas las funciones frecuentes.
  Gastar recursos donde se gasta el tiempo.

- **Ley de Amdahl**: Permite caracterizar este principio

Permite la evaluación del speedup que se obtendrá al aplicar una cierta mejora, M, que permite ejecutar una parte del código **x** veces más rápido.

$$\textbf{Def:}\quad \text{Speedup(E)} = \frac{\text{TEj sin M}}{\text{TEj con M}} = \frac{\text{Performance con M}}{\text{Performance sin M}}$$



Si la mejora sólo acelera la ejecución de un fracción F de la tarea, el tiempo de ejecución del resto permanece sin modificación. Por tanto es muy importante el porcentaje de la tarea que es acelerada.

$$F = \frac{t_B}{t_A + t_B + t_C}$$

Un principio simple

❑ La Ley Amdahl

$$TEj_{nuevo} = TEj_{antiguo} \times \left[(1 - Fraccion_{mejora}) + \frac{Fraccion_{mejora}}{x} \right]$$

$$Speedup = \frac{TEj_{antiguo}}{TEj_{nuevo}} = \frac{1}{(1 - Fraccion_{mejora}) + \frac{Fraccion_{mejora}}{x}}$$

Ejemplo 1:. El 10% del tiempo de ejecución de mi programa es consumido por operaciones en PF. Se mejora la implementación de la operaciones PF reduciendo su tiempo a la mitad

$TEj_{nuevo} = TEj_{antiguo} \times (0.9 + 0.1 / \mathbf{2}) = 0.95 \times TEj_{antiguo}$ $\qquad$ $Speedup = \frac{1}{0.95} = 1.053$

Mejora de sólo un 5.3%

Ejemplo 2: Para mejorar la velocidad de una aplicación, se ejecuta el 90% del trabajo sobre 100 procesadores en paralelo. El 10% restante no admite la ejecución en paralelo.

$TEj_{nuevo} = TEj_{antiguo} \times (0.1 + 0.9 / \mathbf{100}) = 0.109 \times TEj_{antiguo}$ $\qquad$ $Speedup = \frac{1}{0.109} = 9.17$

El uso de 100 procesadores sólo multiplica la velocidad por 9.17