

MULTIPROCESADORES

Rubén Gran Tejero y Jesús Alastruey Benedé

Curso 2019-2020

3º curso del Grado en Ingeniería en Informática
Especialidad Ingeniería de Computadores

- Basada en la asignatura “Fundamentos de Arquitecturas Paralelas”, de Ingeniería informática, impartida desde el Curso 1995-96 hasta el 2012-13.
- Algunas transparencias tomadas de cursos del profesor José María Llaboréa, Facultad de Informática de Barcelona, UPC.

INTRODUCCIÓN

1. Clasificación de computadores Paralelos de Flynn

2. Máquina MIMD: Objetivos

A. Prestaciones

- O1: Disminuir latencia (latency)
- O2: Aumentar productividad de ejecución (throughput)

B. Consumo

- O3: Gastar menos energía
- O4: Disipar menos potencia

C. Consumo

- O5: Facilitar modos “críticos” de operación
- O6: Conseguir escalabilidad en infraestructura y prestaciones

D. Productividad de Desarrollo

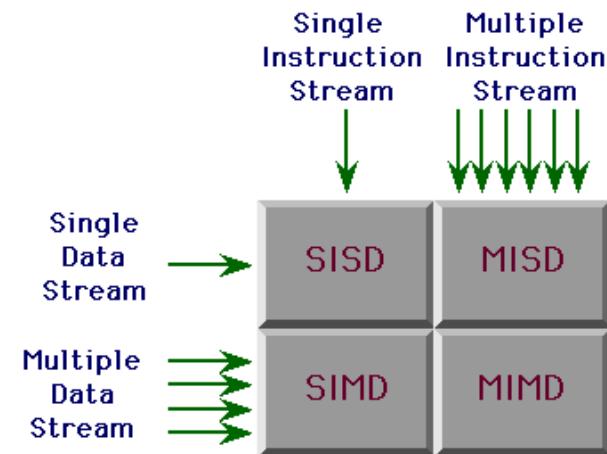
- O7: Ofrecer un buen entorno de programación-depuración
- O2: Disipar menos potencia

3. Objetivo O1, disminuir latencia

- Ej.1: Ley de Amdhal
- Ej.2: Algoritmo Productor – Consumidor
- Ej.3: Algoritmo Genérico “TODOS con TODOS” con N iteraciones

CLASIFICACIÓN DE COMPUTADORES PARALELOS

- Mike Flynn, 1966¹
- **SISD:** Single instruction operates on single data element
 - Conventional processor
- **SIMD:** Single instruction operates on multiple data elements
 - Array processor
 - ILLIAC-IV, BSP, Connection Machine, ICL-DAP, MasPar
 - Vector processors
 - Cray, Convex, Fujitsu, NEC SX
 - Multimedia extensions
 - Intel AVX, ARM NEON, Apple-IBM-Freescale ALTIVec, ...
- **MISD:** Multiple instructions operate on single data element
 - Closest form: systolic array processors, streaming processors

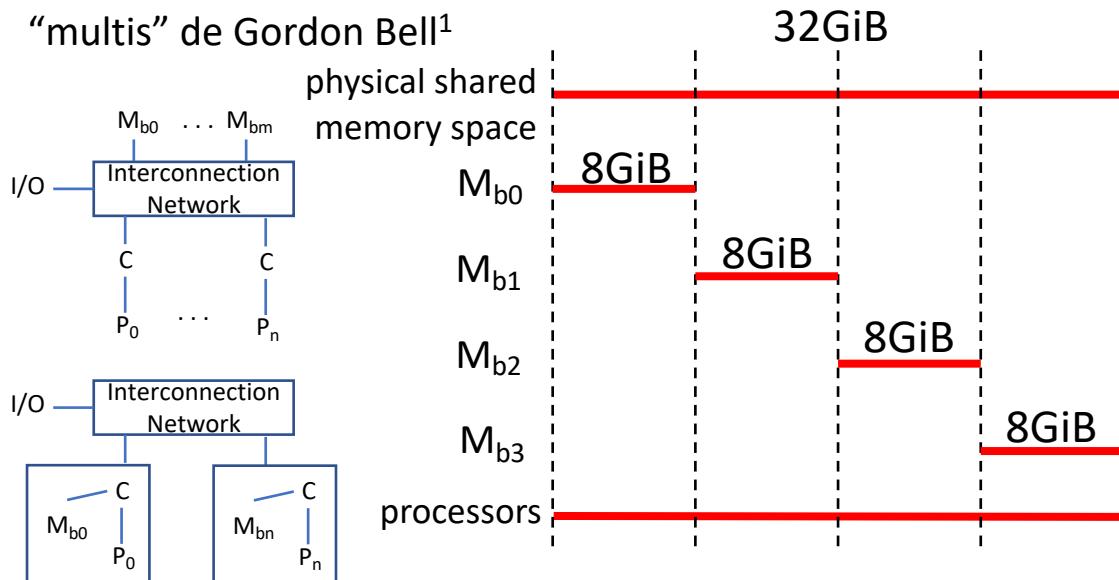


- **MIMD:** Multiple instructions operate on multiple data elements
 - Threads may be independent or not
 - Multithreaded processors
 - **Multiprocessors.** Two big families according to the **unit of work** and the **communication-synchronization method**
 - **Threads** communicate-synchronize by **writing and reading a shared memory**
 - **Processes** communicate-synchronize by **passing messages**

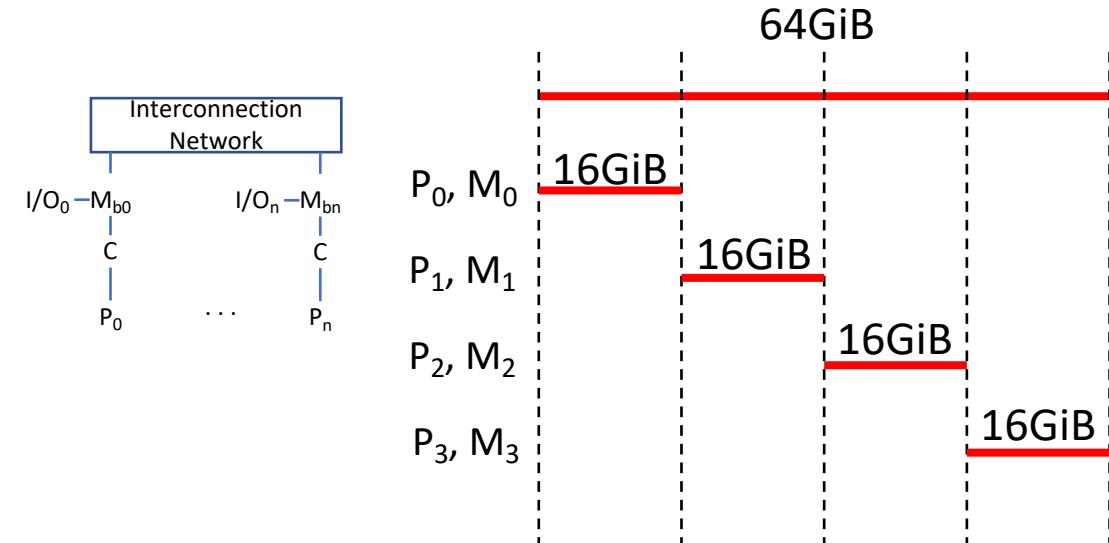
1) M. J. Flynn, "Very High-Speed Computing Systems," Proceedings of the IEEE, volume 54, issue 12, p. 1901-1909, December 1966.

CLASIFICACIÓN DE COMPUTADORES PARALELOS (II)

- **Threads** comm-sync by writing and reading a shared memory
 - Shared-memory multiprocessors, or tightly coupled multiprocessors
 - Symmetric multiprocessors (SMPs)
 - “multis” de Gordon Bell¹



- **Processes** comm-sync by passing messages
 - Loosely coupled multiprocessors
 - Local-private memory multiprocessors
 - “multicomputers” de Gordon Bell¹



1. C. G. Bell, “Multis: a new class of multiprocessor computers”, Science vol. 228, pp. 462-467, April 26, 1985. Gordon Bell is a principal researcher in the Microsoft Research Silicon Valley Laboratory, working in the San Francisco Laboratory. His interests include extreme lifelogging, digital lives, preserving everything in cyberspace, and cloud computing as a new computer class and platform. He proselytizes Jim Gray’s Fourth Paradigm of Science. research.microsoft.com

MÁQUINAS MIMD: OBJETIVOS

A. PRESTACIONES

O1. Disminuir latencia

- Disminuir el tiempo de ejecución de una aplicación
- Supercomputación -> volvemos enseguida

O2. Aumentar la productividad de ejecución (throughput)

- Aumentar el número de trabajos terminados por unidad de tiempo
- Servidores transaccionales

1. ejemplo: batería de 1200 mAh@3V
= fuente de 3V dando 120mA durante 10 horas
= 360 mW durante 10 horas = 3,6 Wh = 12,96 KJ

B. CONSUMO

O3. Gastar menos energía ($E \propto V^2 \cdot \sum C_i + P_{est} \cdot t$)

- Disminuir huella de carbono
- Aumentar duración de batería¹

O2. Disipar menos potencia ($E \propto V^2 \cdot \sum C_i \cdot f + P_{est}$)

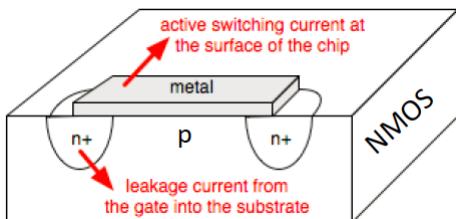
- En régimen estacionario, a temperatura constante, potencia que consume el chip = calor a retirar por unidad de tiempo
- Disminuir el coste del sistema de disipación/refrigeración
 - Radiador de aletas
 - Radiador de aletas con ventilador
 - ...
 - Enfriar agua y bombearla a unos radiadores (i.e. Mare Nostrum)

MÁQUINAS MIMD: OBJETIVOS: CONSUMO (I)

- Un trabajo paralelizable puede terminarse en el mismo tiempo T_{ex} con un procesador, o con dos a $\frac{1}{2}$ de F y de V, *con menos consumo*

$$E(1 \text{ core}, f, V) = P_{1\text{core}} \cdot T_{ex}$$

$$P_{1\text{core}} = P_{\text{din}} + P_{\text{est}}$$



Por otra parte, f y V están relacionados de forma lineal,
 $\uparrow f \rightarrow \uparrow V$,
 f_{\min} relacionada con
 $V_{\min} = V_{\text{umbral}}$

$P_{\text{din}} = \alpha \cdot C \cdot V^2 \cdot f$	
α	factor de conmutación $\leq 1/2$
C	Σ capacidades conmutadas durante T_{ex} (transistores y cables del core)
V	Tensión de alimentación
f	Frecuencia de reloj
$P_{\text{est}} = I_{\text{fugas}} \cdot V$	
I_{fugas}	Aumenta con la escala de integración, la temperatura, y ...

$$E(2 \text{ core}, f/2, V/2) = P_{1\text{core}} \cdot T_{ex}$$

$$P_{2\text{core}} = P_{\text{din}} + P_{\text{est}}$$

$$P_{\text{din}} = \alpha \cdot 2 \cdot C \cdot (V/2)^2 \cdot f$$

$$P_{\text{din}} = \alpha \cdot C \cdot V^2 \cdot f/4$$

$$P_{\text{est}} = 2 \cdot I_{\text{fugas}} \cdot V/2$$

$$P_{1\text{core}} = \alpha \cdot C \cdot V^2 \cdot f + I_{\text{fugas}} \cdot V$$

vs.

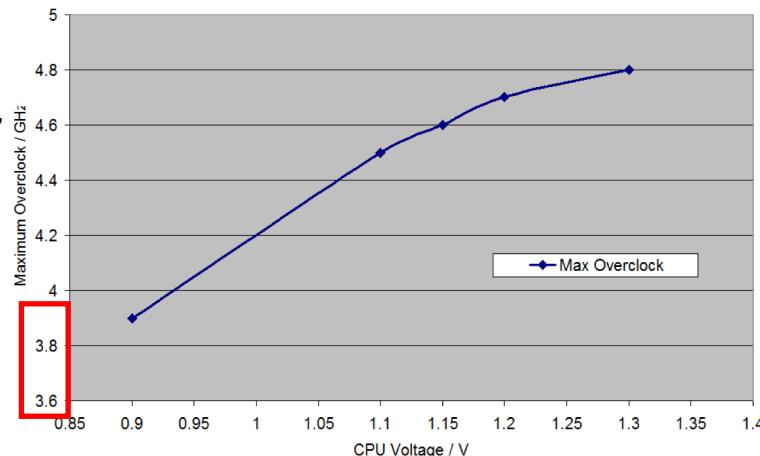
$$P_{2\text{cores}} = \alpha \cdot C \cdot V^2 \cdot f/4 + I_{\text{fugas}} \cdot V$$

- Usar DVFS (Dynamic Voltage-Frequency Scaling) permite disminuir el consumo sin perder prestaciones en aplicaciones paralelizables
- Factor 4 como máximo:
si relación potenci-tensión cúbica y potencia estática pequeña

MÁQUINAS MIMD: OBJETIVOS: CONSUMO (II)

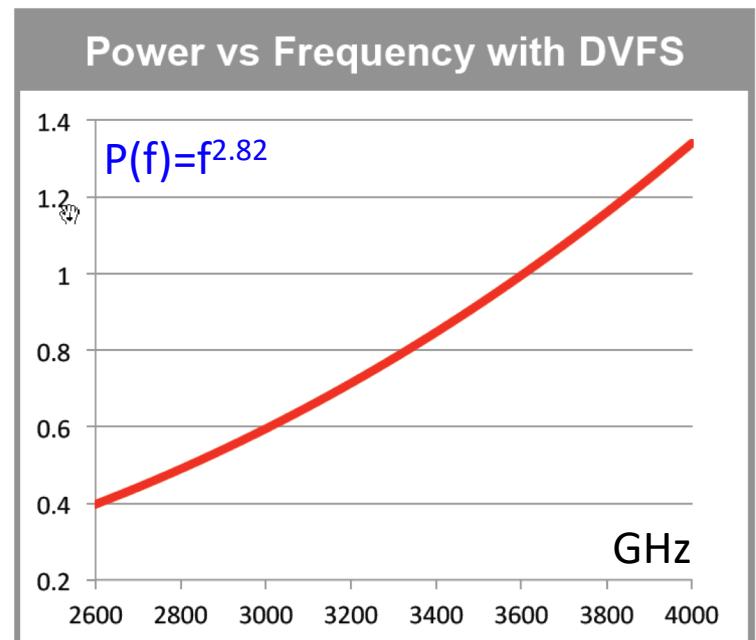
- Ejemplo: relación entre frecuencia y tensión con *overclocking*¹

Intel i7-3770K
(Ivy Bridge, Q2 2012)
4 cores 3.5GHz max 3.9 GHz,
22nm tri-gate



- Ejemplo: relación entre potencia y frecuencia, con escalado lineal de la tensión de alimentación

Oracle SPARC T5 (2013):
13 S3 cores
3.6 GHZ
28nm



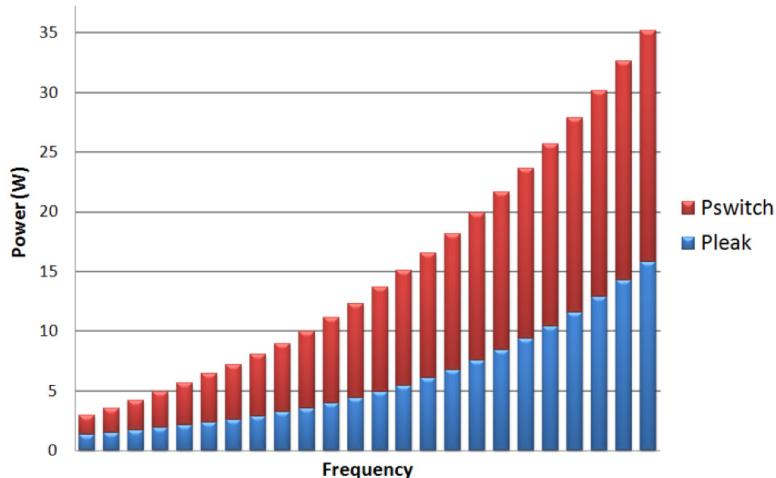
1. Undervolting and Overclocking on Ivy Bridge, Ian Cutress, April 23, 2012
<http://www.anandtech.com/show/5763/undervolting-and-overclocking-on-ivy-bridge>

MÁQUINAS MIMD: OBJETIVOS: CONSUMO (III)

- Relación entre potencia dinámica y potencia estática

High-end AMD cores,
possibly 45nm Opteron¹

P_{leak} may reach up to 40%
of P_{switch} and that
percentage does not
depend much on
frequency



A not so optimistic view
of the future

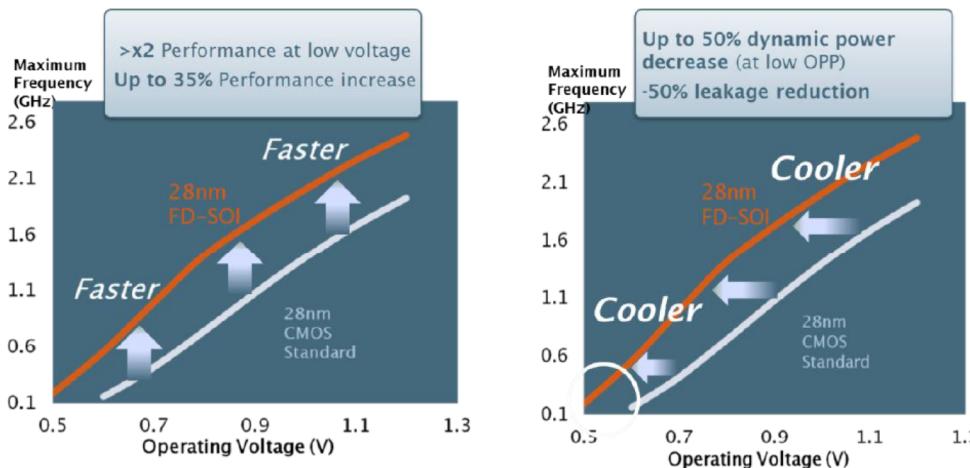


Figure 1: Leakage power becomes a growing problem as demands for more performance and functionality drive chipmakers to nanometer-scale process nodes (Source: IBS).

1. Practical Power Gating and Dynamic Voltage/Frequency Scaling Issues ,
August 17, 2011, Stephen Kesonocky. AMD

MÁQUINAS MIMD: OBJETIVOS: CONSUMO (IV)

- Más ejemplos de DVFS
NovaThor L8580¹, STM-Ericsson integrates on a single 28nm FD-SOI die;
 - 2-core, Equead 2.5GHz **processor** based on ARM Cortex-A9
 - Imagination PowerVR™ SGX544 **GPU** @ 600MHz
 - Advanced multimode **LTE modem**



- It's billed ad "the worl's fastest and lowest-power integrated LTE Smartphone platform", and is sampling in Q1 2013. FD-SOI=Fully Depleted Silicon-On-Insulator (a MOSFET type)

- Plataformas AMD Opteron para servidores², con posibilidad DVFA ajustable desde sistema operativo (2.6.33)

	System Idle power	CPU die codename, year, featuresize, #core	TDP ^a	Frequency (GHz)	Voltage (V)
Compucon K1-1000D-1U	74W	Sledgehammer, 2003, 130nm, 1 core	89W	0.8-2.0	0.9-1.55
Dell PowerEdge SC1435-1U	145W	Santa Rosa, 2006, 90nm, 2 cores	95W	1.0-2.4	0.9-1.35
HP ProLiant DL365 G5-1U	133W	Shanghai, 2009, 45 nm, 4 cores	75W	0.8-2.7	1.0-1.35

a. Thermal Design Power

1. M. Cornero and A. Anyuru, "Multiprocessing in Mobile Platforms: the Marketing and the Reality", 2012?, [ST-Ericsson white paper](#)
2. E. Le Sueur and G Heiser, "Dynamic Voltage and Frequency Scaling: The Laws of Diminishing Returns", Proc. of the 2010 int. conf. on Power aware computing and systems (HotPower'10).

MÁQUINAS MIMD: OBJETIVOS

C. PROTECCIÓN DE LA INVERSIÓN

O5. Facilitar modos “críticos” de operación (24x7x365)

- Safety critical -> salud
 - Aviónica, automóvil, instrumentación médica, energía nuclear, ...
- Mission critical -> negocios
 - Ministerios, bancos, bolsa, expedientes clínicos, ...

Ejemplo de misión crítica: base de datos de hacienda

RAS = Reliability, Availability ,Serviceability

$$Availability = \frac{uptime}{uptime + downtime}$$



O6. Conseguir escalabilidad en infraestructura y prestaciones

- Infraestructura: reuso del diseño y posibilidad de crecimiento
 - Dentro del chip, memoria compartida
 - Agrupaciones compactas de chips, memoria compartida
 - En un sustrato de alta densidad de interconexión
 - En una placa de circuito impreso
 - Agrupaciones de placas, paso de mensajes
 - Servidores estándar horizontales para montar en rack
 - Microservers
 - Servidores tipo Blade



MÁQUINAS MIMD: OBJETIVOS: PROTECCIÓN DE LA INVERSIÓN

- Dentro del chip: 2, 4, ..., 32, 64 procesadores llamados cores si consideramos también las cache privadas ejemplo chips AMD para servidores

AMD Opteron™ 6200 Series Processors						
Model	Cores	Core Speed	AMD Turbo CORE Max Frequency	L3 Cache	TDP	Socket Type
6284 SE	16	2.7GHz	3.4GHz	16MB	140W	G34
6282 SE	16	2.6GHz	3.3GHz	16MB	140W	G34
6278	16	2.4GHz	3.3GHz	16MB	115W	G34
6276	16	2.3GHz	3.2GHz	16MB	115W	G34
6274	16	2.2GHz	3.1GHz	16MB	115W	G34
6272	16	2.1GHz	3.0GHz	16MB	115W	G34
6262 HE	16	1.6GHz	2.9GHz	16MB	85W	G34
6238	12	2.6GHz	3.2GHz	16MB	115W	G34
6234	12	2.4GHz	3.0GHz	16MB	115W	G34
6220	8	3.0GHz	3.6GHz	16MB	115W	G34
6212	8	2.6GHz	3.2GHz	16MB	115W	G34
6204	4	3.3GHz	N/A	16MB	115W	G34

2011-12 Interlagos, 32nm
Land Grid Array (LGA), 1974 pins



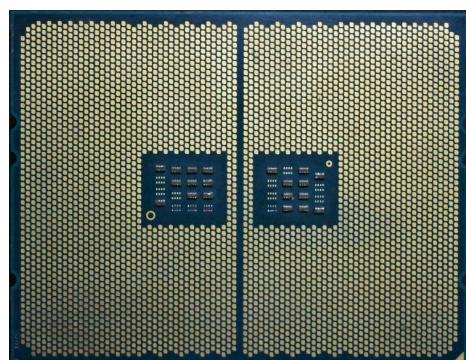
	Cores	Base Speed	TDP*	List Price	Est Perf†	Perf/\$	Perf/W
Epyc 7601	32	2.2GHz	180W	\$4,200	63	15	35
Plat 8176	28	2.1GHz	172W	\$8,719	61	7	36
Epyc 7451	24	2.3GHz	180W	\$2,400	50	21	28
Gold 6148	20	2.4GHz	157W	\$3,072	49	16	32
Epyc 7351	16	2.4GHz	155W	\$1,100	35	31	22
Gold 6130	16	2.1GHz	132W	\$1,894	35	18	26
Epyc 7251	8	2.1GHz	120W	\$475	15	32	17
Silver 4110	8	2.1GHz	92W	\$501	17	35	19

2017

Table 1. Epyc versus Xeon Scalable (selected models). Compared with Intel products that have similar performance, AMD's Epyc generally offers lower prices but greater power. *Intel parts include 7W for the south bridge; †core count times base speed times IPC (0.9 for Epyc, 1.03 for Xeon Scalable). (Source: vendors, except †The Linley Group)

AMD EPYC 7001 & 7002 Processors (2P) 2020

	Cores Threads	Frequency (GHz)		L3*	TDP	Price
		Base	Max			
EPYC 7742	64 / 128	2.25	3.40	256 MB	225 W	\$6950
EPYC 7702	64 / 128	2.00	3.35	256 MB	200 W	\$6450
EPYC 7642	48 / 96	2.30	3.20	256 MB	225 W	\$4775
EPYC 7552	48 / 96	2.20	3.30	192 MB	200 W	\$4025
EPYC 7542	32 / 64	2.90	3.40	128 MB	225 W	\$3400
EPYC 7502	32 / 64	2.50	3.35	128 MB	200 W	\$2600
EPYC 7452	32 / 64	2.35	3.35	128 MB	155 W	\$2025
EPYC 7402	24 / 48	2.80	3.35	128 MB	155 W	\$1783
EPYC 7352	24 / 48	2.30	3.20	128 MB	180 W	\$1350
EPYC 7302	16 / 32	3.00	3.30	128 MB	155 W	\$978
EPYC 7282	16 / 32	2.80	3.20	64 MB	120 W	\$650
EPYC 7272	12 / 24	2.90	3.20	64 MB	155 W	\$625
EPYC 7262	8 / 16	3.20	3.40	128 MB	120 W	\$575
EPYC 7252	8 / 16	3.10	3.20	64 MB	120 W	\$475



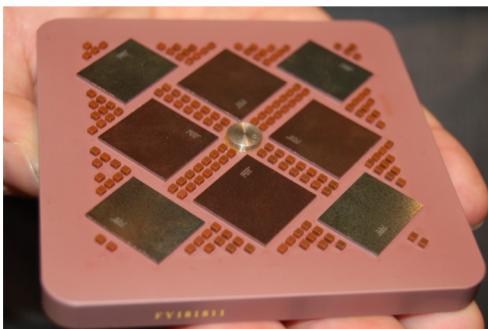
Socket SP3 (LGA) for EPYC
4096-contacts

MÁQUINAS MIMD: OBJETIVOS: PROTECCIÓN DE LA INVERSIÓN

Intel Second Gen Xeon Scalable (Cascade Lake)					AMD Second Gen EPYC ("Rome")					
		Cores	Freq	TDP (W)	Price	AMD	Cores	Freq	TDP	Price
Xeon Platinum 8200										
8280	M	28	2.7/4.0	205	\$13012	7742	64	2.25/3.40	225	\$6950
8280		28	2.7/4.0	205	\$10009					
8276	M	28	2.2/4.0	165	\$11722	7742	64	2.25/3.40	225	\$6950
8270		26	2.7/4.0	205	\$7405					
8268		24	2.9/3.9	205	\$6302					
8260	M	24	2.4/3.9	165	\$7705	7702	64	2.00/3.35	225	\$6450
8260		24	2.4/3.9	165	\$4702	7552	48	2.20/3.50	200	\$4025
8253		16	2.2/3.0	165	\$3115	7502	32	2.50/3.35	200	\$2600
Xeon Gold 6200										
6252		24	2.1/3.7	150	\$3665					
6248		20	2.5/3.9	150	\$3072					
6242		16	2.8/3.9	150	\$2529	7452	32	2.35/3.35	155	\$2025
6238		22	2.1/3.7	140	\$2612	7402	24	2.80/3.35	155	\$1783
6226		12	2.8/3.7	125	\$1776					
Xeon Silver 4200										
4216		16	2.1/3.2	100	\$1002	7282	16	2.80/3.20	120	\$625
4214		2x12	2.2/3.2	2x85	2x\$694	7402P	24	2.80/3.35	180	\$1250

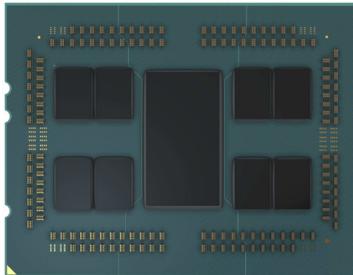
MÁQUINAS MIMD: OBJETIVOS: PROTECCIÓN DE LA INVERSIÓN

- Agrupaciones compactas de chips
 - En un sustrato de alta densidad de interconexión:
Multi Chip Module (MCM)



IBM POWER5 MCM with four and four 36 MB external L3 cache dies on a ceramic MCM.
http://en.wikipedia.org/wiki/Multi-chip_module

AMD Epyc 7002, Rome, 2Socket, up-to 128 cores (2020)



- En un placa de circuito impreso (PCB=Printed Circuit Board)

	Sockets	PCIe 3.0†	QPI	L3 Cache	Memory Channels	Max Memory	DIMMs	CPUs/Threads
E5-4600	4P	Up to 40/10	Up to 2 ports	Up to 20MB	4	1.5TB*	48	8/16
E5-2600	2P	Up to 24/6	1 port	Up to 20MB	4	768GB*	24	8/16
E5-2400	2P	Up to 16/3	None	Up to 8MB	3	384GB*	12	8/16
E3-1200 v2	1P	Up to 16/3	None	Up to 8MB	2	32GB	4	4/8

Table 1. New Intel Xeon processor families. The company added three new Xeon lines, including the four-socket E5-4600, the dual-socket E5-2400, and the E3-1200 v2 based on the Ivy Bridge microarchitecture. *Maximum memory for multiple-socket configurations; †lanes/controllers per chip. (Source: Intel)

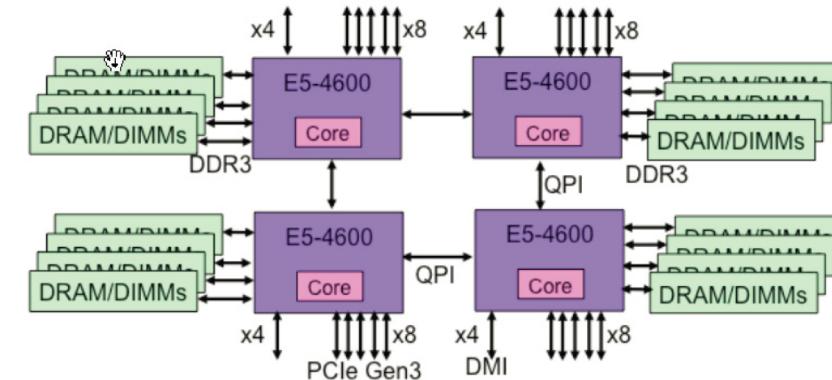
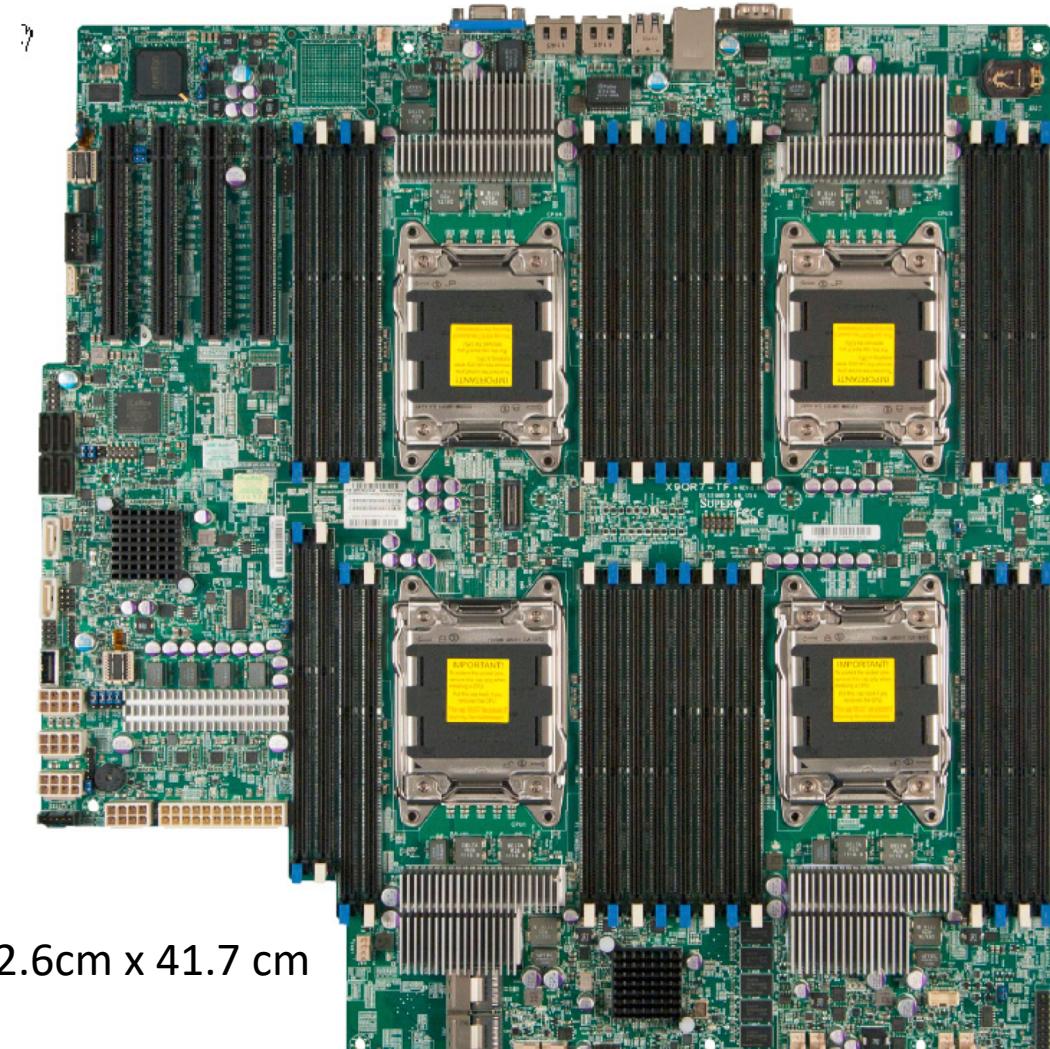


Figure 1. Intel Xeon E5-4600 four-socket configuration. The E5-4600 processors form a ring using the two QPI ports per device. Unlike the dual-socket Xeons, the E5-4600 family can pass traffic destined for another processor

J. Bolaria, "Xeon E5 xtends to big servers", Microprocessors report, June 4, 2012.

MÁQUINAS MIMD: OBJETIVOS: PROTECCIÓN DE LA INVERSIÓN



- Supermicro Motherboard X9QR7-TF+
 - Quad socket R (LGA 2011) supports Intel® Xeon® processor E5-4600
 - Intel® C602 chipset
 - Up to **1TB** DDR3 1600MHz ECC Registered DIMM; 32x DIMM sockets
 - Expansion slots: 2x PCIe3.0x16 and 2x PCIe3.0 x8
 - Intel® X540 Dual port 10GBase-T
 - 2x SATA3 ports + 4x SATA2 ports + 8x SAS2 ports via LSI 2208
 - Integrated IPMI 2.0 + KVM over LAN
 - 8x USB 2.0 ports (4 rear, 3 front + 1 Type A)
 - CUANTO VALE MONTADO EN TORRE ? --> ver p.e.
 - Titanus X650 (<https://www.titancomputers.com/Titan-X650-Quad-CPUs-Intel-Xeon-E7-4800-8800-p/x650.htm>)

MÁQUINAS MIMD: OBJETIVOS: PROTECCIÓN DE LA INVERSIÓN

- Agrupaciones de placas:
 - No es frecuente compartir memoria
 - La comunicación/sincronización se basa en paso de mensajes a través de una red más o menos elaborada
- Módulos estándar horizontales para montar en **rack**
 - 1U, 2U, 3U, 4U, ... (U=1.75 inches 44.5 mm)
 - 1 armario/rack \approx 48 Us

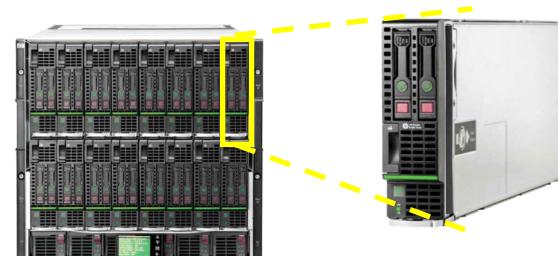


- **Microservers:** pequeños módulos *enrakkables* verticales con un chasis sencillo



Dell PowerEdge C5125 12-sled
3U chasis (2012)

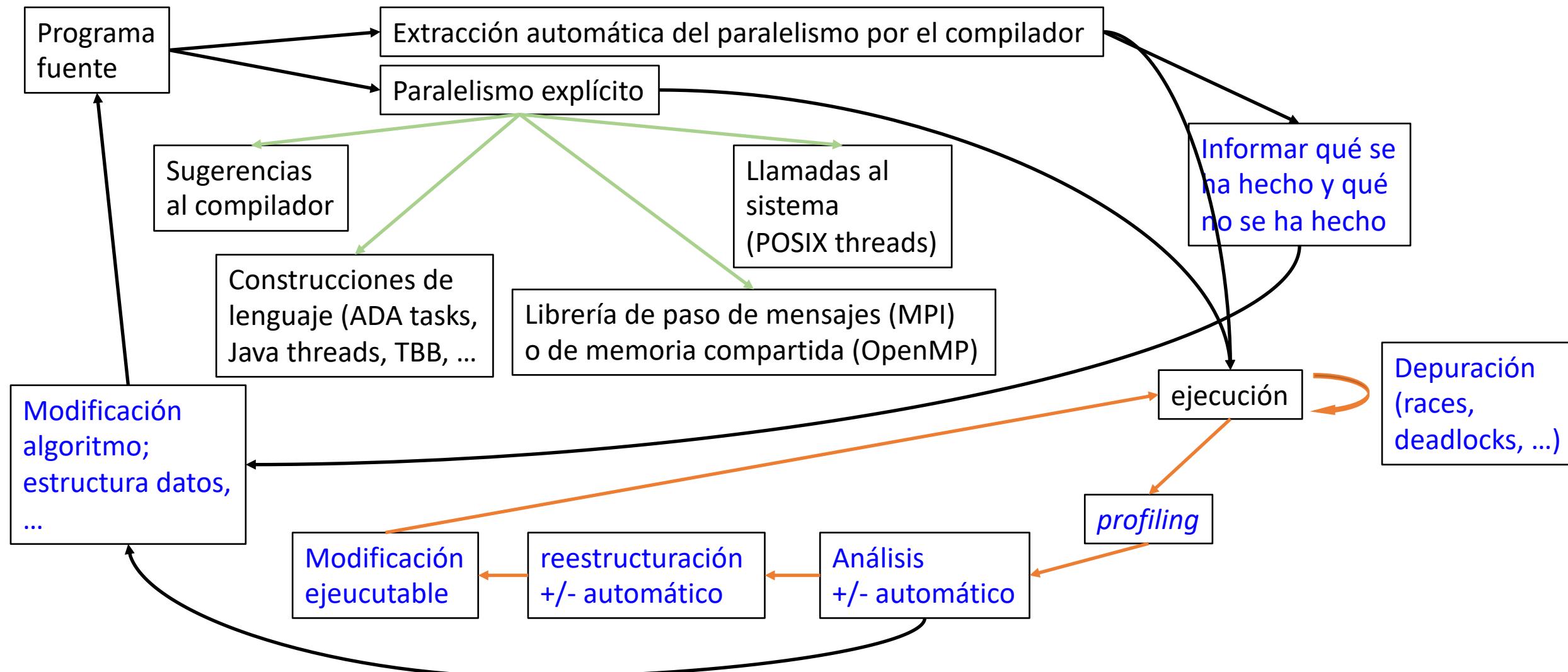
- **Blades:** módulos verticales para montar en un chasis que suministra conexiones, ventilación, alimentación, etc.



HP Proliant BL460c Gen8
blade server
10U chasis

- Centros de datos
 - Supercomputación (la red de interconexión es CLAVE para el rendimiento)
 - Negocios: empresa, cloud, mixto

MÁQUINAS MIMD: OBJETIVOS: PRODUCTIVIDAD DE DESARROLLO



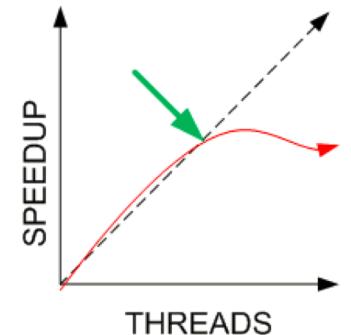
OBJETIVO 01: DISMINUIR LATENCIA

- Multiprocessors, two big families:
 - According to the **unit of work**, and the **communication-synchronization** method
 - **Threads** communicate-synchronize by **writing-reading a shared memory**
 - **Processes** communicate-synchronize by **passing messages**
- Buscamos algoritmos paralelos capaces de manejar N procesadores que colaboran en la resolución de un problema único

$$\text{Speedup } (N) = \frac{T_{seq} (1)}{T_{par} (N)}$$

- Conseguir un *speedup* lineal con el número de procesadores implica:

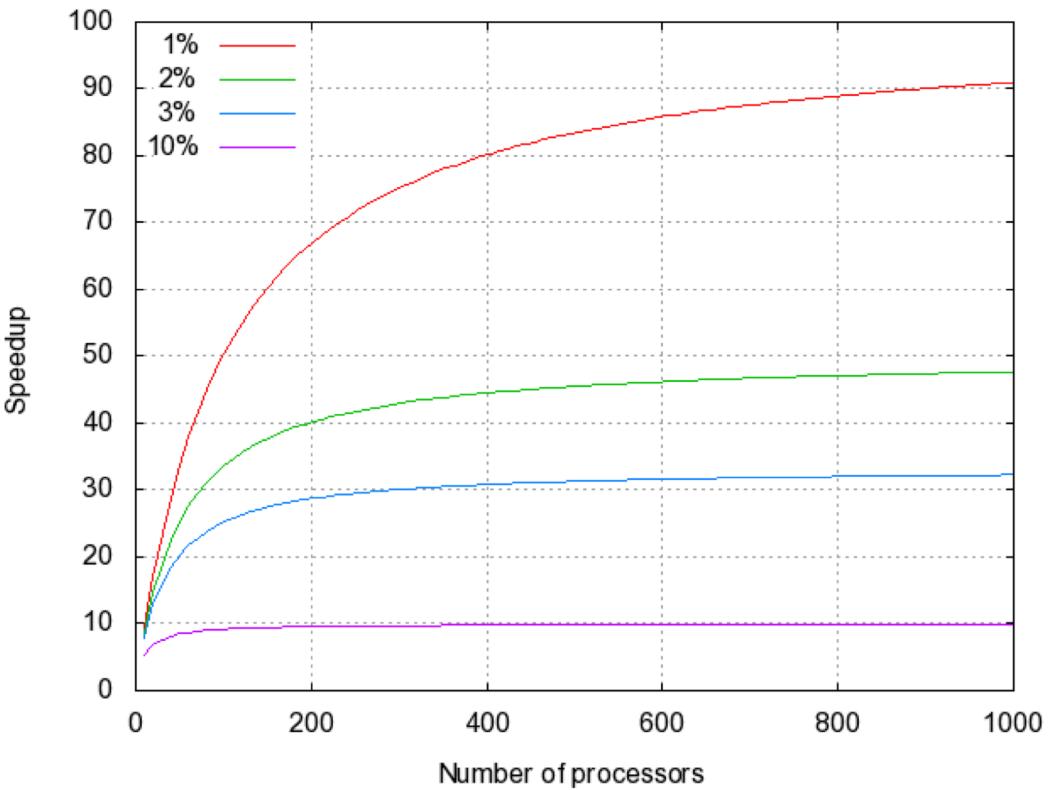
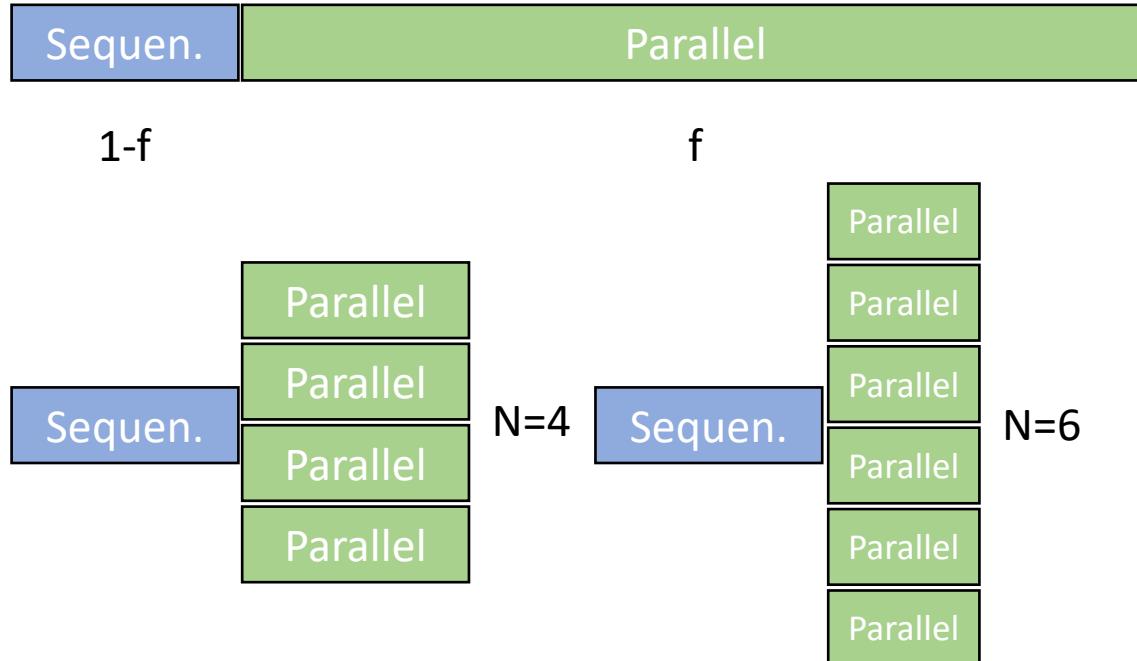
- Todos los procesadores ocupados **(1)**
- Los procesadores no realizan trabajo extra, i.e. trabajo que no se hace en el algoritmo secuencial **(2)**



- Posibles frenos:

- Al repartir el trabajo = planificar = *scheduling*
 - estática, en compilación -> mal balance **(¬1)**
 - Dinámica, en ejecución -> sobrecarga **(¬2)**
- Por el retardo de comunicación **(¬2)**
- Reestructuración de datos (*re-shaping*) **(¬2)**
 - E.g. RGBY RGBY RGBY ... -> RRR ... GGG... BBB... YYY...

EJEMPLO 1: LEY DE AMDHAL



$$\text{Speedup} (N) = \frac{T_{seq}(1)}{T_{par}(N)} =$$

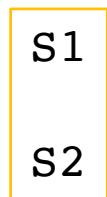
EJEMPLO 2

- Algoritmo **productor-consumidor**
 - productor y consumidor tardan lo mismo: R unidades de tiempo

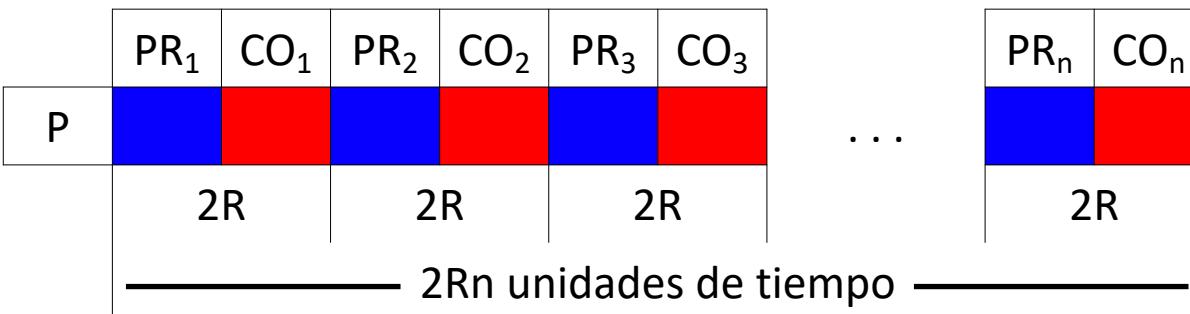
$i=1, n$

S1: $K = PR(i)$

S2: $A[i] = CO(K)$



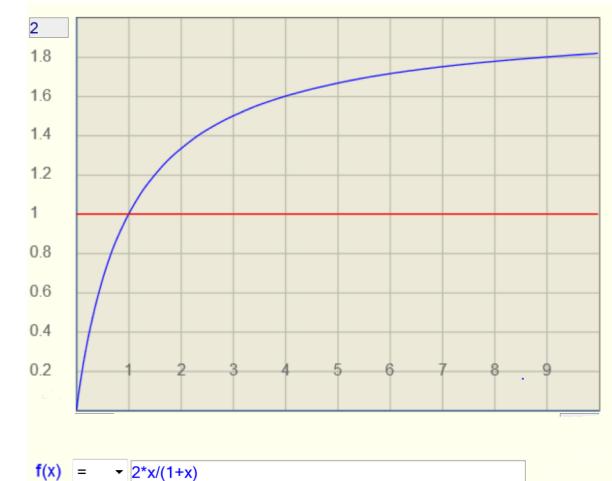
- Ejecución secuencial en **un** procesador P:



- Ejecución en dos procesadores P1 y P2, con algún mecanismo sencillos de comunicación-sincronización:
 - Los procesadores no pueden calcular y comunicar a la vez
 - Comunicar el valor K entre P1 y P2 cuesta C unidades de tiempo, tiempo que ocupa a los dos procesadores

$$Sup(2, n) = \frac{Ts(n)}{Tp(2, n)} =$$

$$\lim_{n \rightarrow \infty} Sup =$$



- Si $R/C < 1$ no vale la pena!

EJEMPLO 3

- Algoritmo genérico M funciones, “todos con todos”, n iteraciones

```
for (i=0, i<n, i++) {  
    T1[mod2(i+1)] = F1( T1[mod2(i)], T2[mod2(i)], ... TM[mod2(i)] )  
    ...  
    TM[mod2(i+1)] = FM( T1[mod2(i)], T2[mod2(i)], ... TM[mod2(i)] )  
}
```

- En cada iteración se calculan **M** funciones $F_1, \dots, F_m, \dots, F_M$ y sus resultados se dejan en la variables $T_m[0], T_m[1], T_1[\text{mod2}(i+1)], \dots, T_m[\text{mod2}(i+1)], \dots, T_M[\text{mod2}(i+1)]$
- Cada función usa todos los resultados de la iteración anterior
- Cada función tarda lo mismo: **R unidades de tiempo**

- Para ejecutar este algoritmo en N procesadores ($2 \leq N \leq M$):

- Convertimos las funciones F_1, F_2, \dots, F_M en procesos (o threads)
- Asignamos procesos a procesadores (planificación estática)
- Cada procesador calcula función, difunde resultado y recibe el resto de valores

- Ejemplo con tres funciones y dos procesadores (PR1 PR2)

3 funciones con 2 procesadores	Comunicación todos con todos	3 funciones con 2 procesadores	Comunicación todos con todos	3 funciones con 2 procesadores
Iter 1 $F_1 F_2$ F_3	$\text{PR1} \leftrightarrow \text{PR1}$ $\text{PR1} \leftrightarrow \text{PR2}$ $\text{PR1} \leftrightarrow \text{PR2}$	Iter 2 $F_1 F_2$ F_3	$\text{PR1} \leftrightarrow \text{PR1}$ $\text{PR1} \leftrightarrow \text{PR2}$ $\text{PR1} \leftrightarrow \text{PR2}$	Iter n $F_1 F_2$ F_3
...				...

un solo canal de comunicaciones, compartido para todos los procesadores $\text{PR1} \leftrightarrow \text{PR2}$ $\text{PR2} \leftrightarrow \text{PR1}$

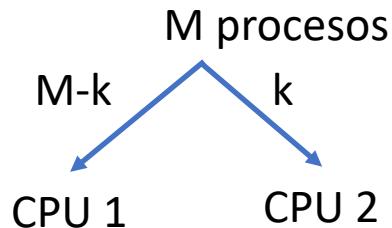
- Comunicar entre procesadores cuesta **C unidades de tiempo** o bien, **0 unidades de tiempo** entre procesos asignados al mismo procesador

PROBLEMA¹

Dados unos costes (R,C)
repartir M procesos en N procesadores
para obtener Speedup máximo

- **M** procesos (o threads)
- Cada proceso pasa por una fase de cálculo de **R** unidades de tiempo
- Tras la fase de cálculo, todos comunican con todos:
 - Comunicación no solapada con cálculo
 - Un solo canal de comunicación para todos los procesadores
 - **C** unidades de tiempo: comunicar dos procesos asignados a diferente procesadores
 - **0** unidades de tiempo comunicar procesos asignados al mismo procesador

2 PROCESADORES



$$T_{exe} = R \cdot \max(M - k, k) + C \cdot (M - k) \cdot k$$

k: factor de reparto
 $0 \leq k \leq M$

- Min (Tex)? -> problema de asignación (planificación estática)

$R/C =$ trabajo útil/sobrecarga
$M/2 =$ MaxPar/2

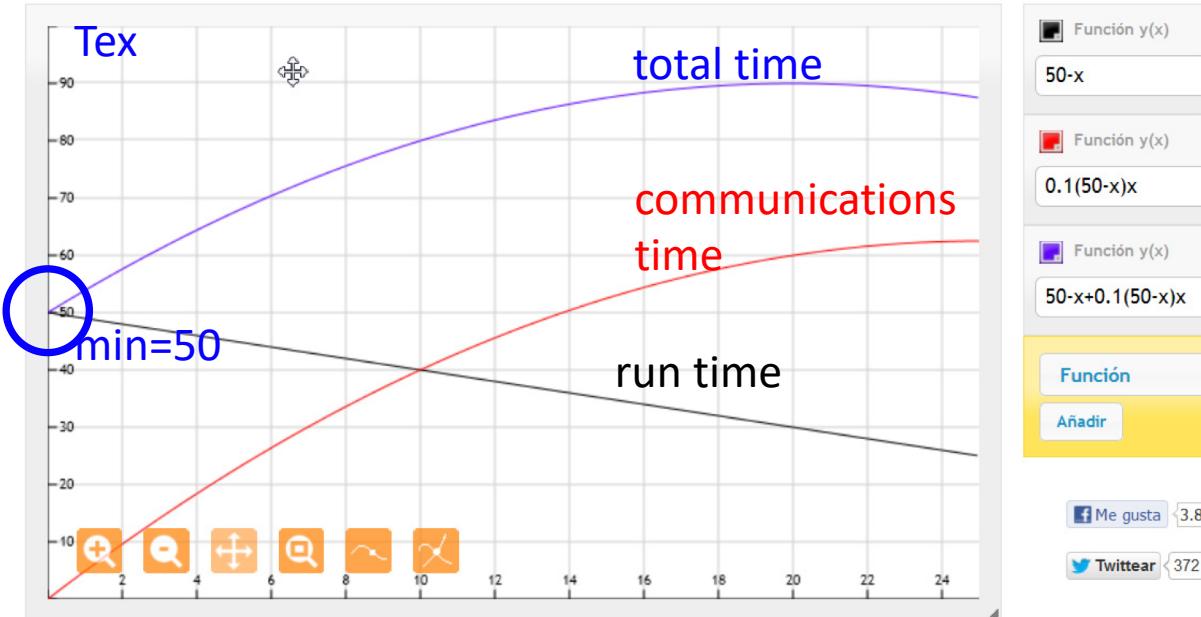
$k=0$ todos a CPU 1 si $R/C < M/2$

$k=M/2$ distribuir por igual si $R/C \geq M/2$

1. Capítulo 6 del libro de Harold S. Stone “High-Performance Computer Architecture”, Addison Wesley, 1987. Tomado de Indurkhya, Stone, and Xi-Cheng, “Optimal partitioning of randomly generated distributed programs”, IEEE Trans. on Software Engineering, Vol. 12, Issue 3, March 1986, pp. 483 - 495.

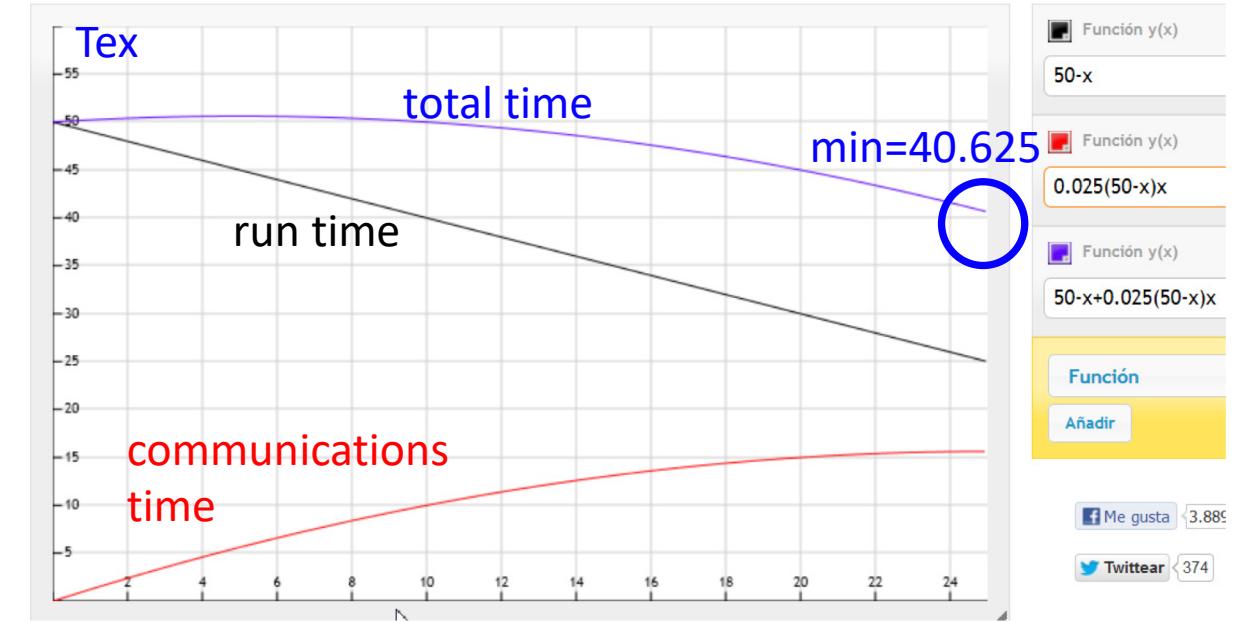
PROBLEMA 1

- $M=50, R=1, C=0.1$



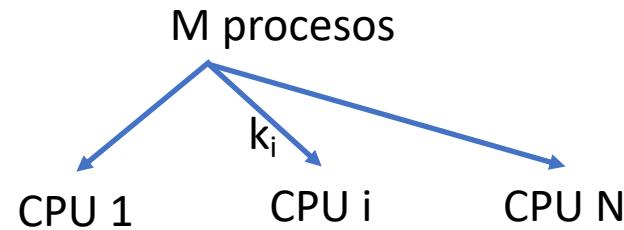
- $R/C=10 \ M/2=25 \rightarrow R/C < M/2$
 - Parámetro de partición óptimo $K=0$; todos los procesos a un procesador, apagar el otro

- $M=50, R=1, C=0.025$



- $R/C=40 \ M/2 = 25 \rightarrow R/C \geq M/2$
 - Parámetro de partición óptimo $K=M/2$: Ejecutar 25 || 25

N PROCESADORES



k_i procesos al procesador i

$$\sum_{i=1}^N k_i = M$$

$$T_{ex} = R \cdot \max(k_i) + \frac{C}{2} \cdot \sum_{i=1}^N k_i \cdot (M - k_i)$$

- De nuevo los óptimos aparecen en las asignaciones extremas

$$R/C = M/2$$

Menor: M procesos a 1 procesadore Sup = 1

Mayor: si M es múltiplo de N ent

M/N procesos a cada procesador
sino

$\lceil M/N \rceil$ procesos a N-2 o N-1 procesadores

e.j.: 19 procesos en 6 procesadores
reparto? Sup?

$$Sup_{k_i=M/N} = \frac{N}{1 + \frac{M \cdot (N - 1)}{2} \div R/C}$$

Si $M \cdot N \div R/C \rightarrow 0$ ent Sup $\rightarrow N$

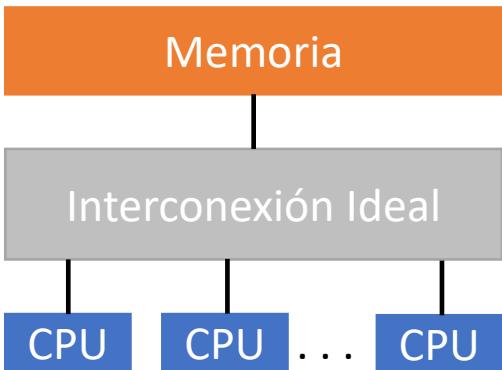
Si $N \rightarrow \infty$ ent Sup $\rightarrow \frac{R/C}{M}$ independientemente de N !!!

ORGANIZACIÓN DE MULTIPROCESADORES DE MEMORIA COMPARTIDA

- 1º MULTIS UMA: UNIFORM MEMORY ACCESS
- 2º MULTIS NUMA: NON-UNIFORM MEMORY ACCESS
- 3º MULTIS EN CHIP, UMA Y NUMA
- 4º MULTIS EN PLACA SIN CHIPS DE INTERCONEXIÓN (GLUELESS)

1º MULTIS UMA: UNIFORM MEMORY ACCESS

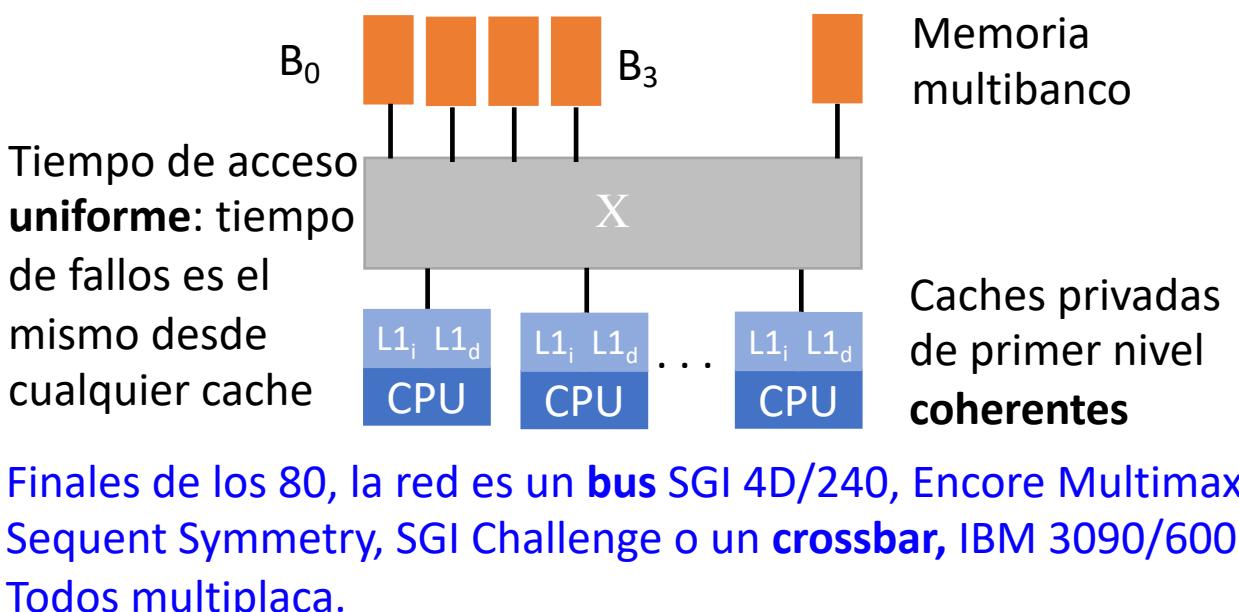
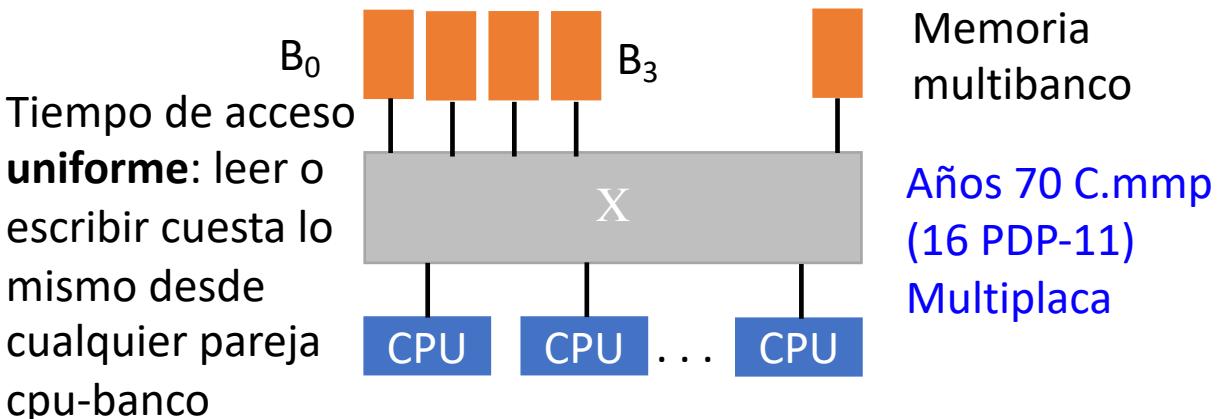
- Modelo de programación:
 - Espacio único de direcciones
 - Comunicación mediante escritura-lectura de memoria



- Evolución histórica del hardware:

1º MULTIS UMA: UNIFORM MEMORY ACCESS

- La memoria principal es multibanco, pero su latencia de servicio es **única**
- La red de interconexión **procesador/memoria** es conmutada (o **indirecta**): bus, crossbar, multietapa, etc.



2ªMULTIS NUMA: NON-UNIFORM MEMORY ACCESS

- Ideal de jerarquía: partimos de un multi de la generación anterior, lo replicamos e interconectamos las réplicas, que se llaman nodos
 - La memoria principal es multibanco, distribuida en los nodos, varios bancos por nodo, pero su latencia de acceso ya **no es única**
 - La red de interconexión nodo-nodo es **punto a punto** (o **directa**): anillo, malla, hipercubo, etc.
 - Sinónimos NUMA (Berkeley):
 - CC-UMA, CC-NUMA -> Cache Coherent ...
 - DSM multiprocessor -> Distributed Shared Memory (Stanford)

Finales de los 70: Cm*, sin cache, 1 cpu/nodo

En los 80: IBM RP3, CEDAR

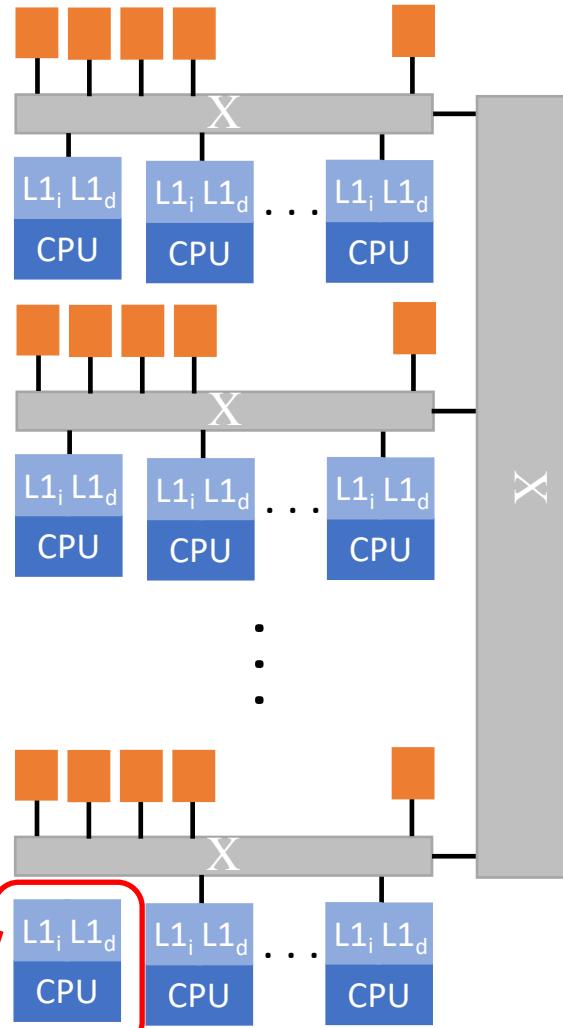
En los 90: Stanford DASH: bus+hipercubo

Convex SPP: crossbar + 4 anillos

Implementaciones multichip y multiplaca

Un **nodo** se parece mucho a un multi de la generación anterior

Tiempo de acceso **no uniforme**: fallo desde cache tarda diferente según el bloque resida en un banco *local o remoto*

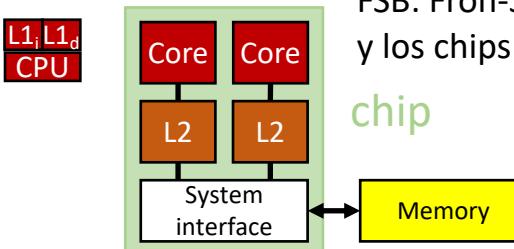


Caches privadas de primer nivel
- **coherentes** -

Red de interconexión global entre nodos multiprocesador

3º MULTIS EN CHIP, UMA Y NUMA

- La memoria principal se accede desde el chip desde 1 o más canales DDR
- La jerarquía de memoria se complica:
 - Múltiples niveles de cache
 - Último nivel de la jerarquía evoluciona de *privado* a *compartido*
 - La red de interconexión se incorpora al chip:
 - Red en chip → *network-on-chip, NoC*
 - Entre cores, entre caches → conmutadas y punto a punto

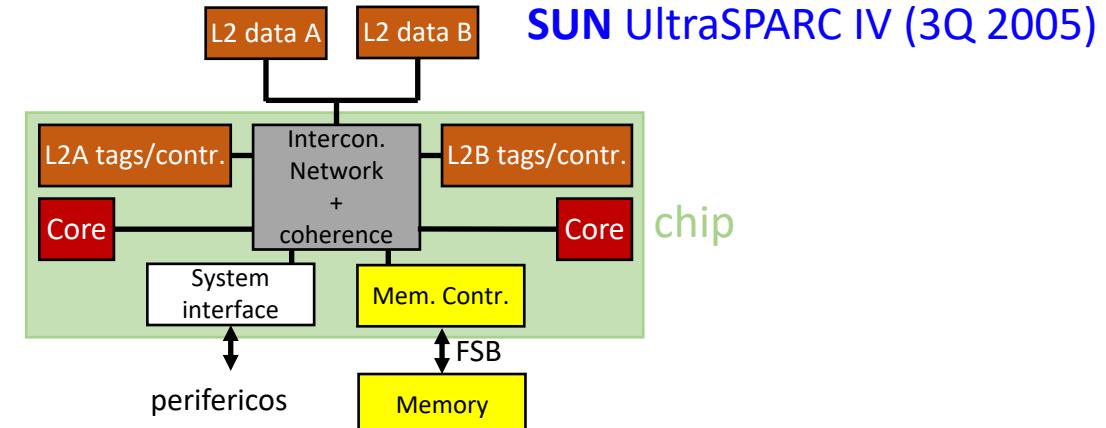


AMD Athlon 64x3 (2Q 2005)
L2 son "exclusivas" -> $L1 \cap L2 = \emptyset$

FSB: Fron-Side Bus, hacia el controlador y los chips de DDR

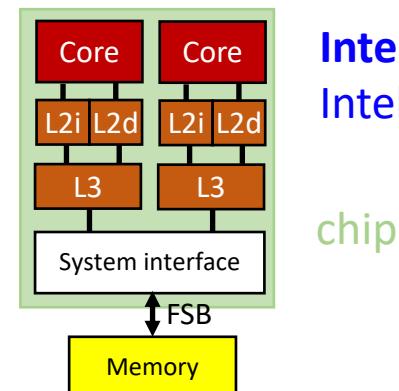
chip

Último Nivel de cache es **PRIVADO** y coherente



SUN UltraSPARC IV (3Q 2005)

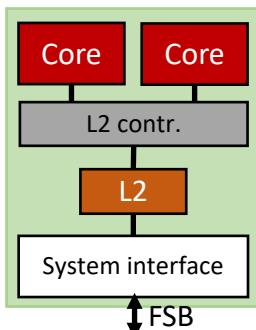
Intel Montecito, dual-core
Intel Itanium (3Q 2006)



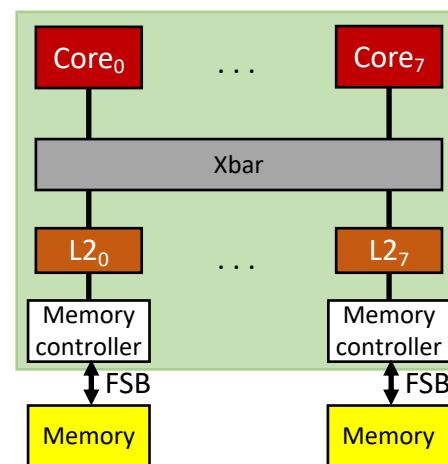
chip

ÚLTIMO NIVEL DE CACHE ES COMPARTIDO

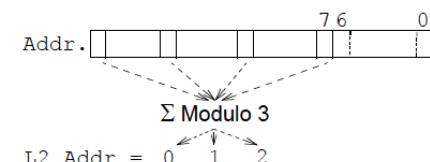
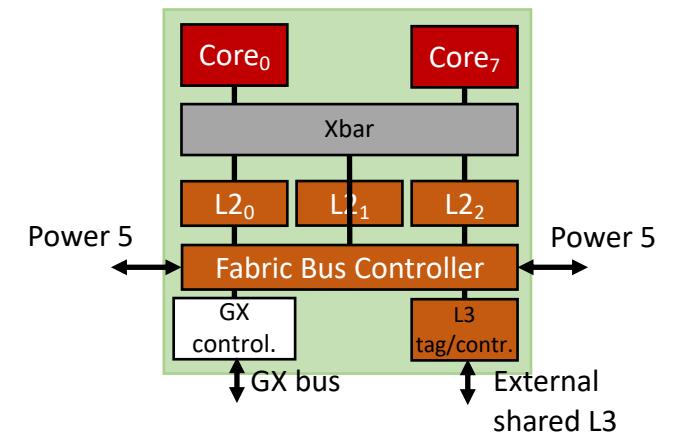
Intel Merom (3Q 2006)



SUN Niagara T1 (2005)

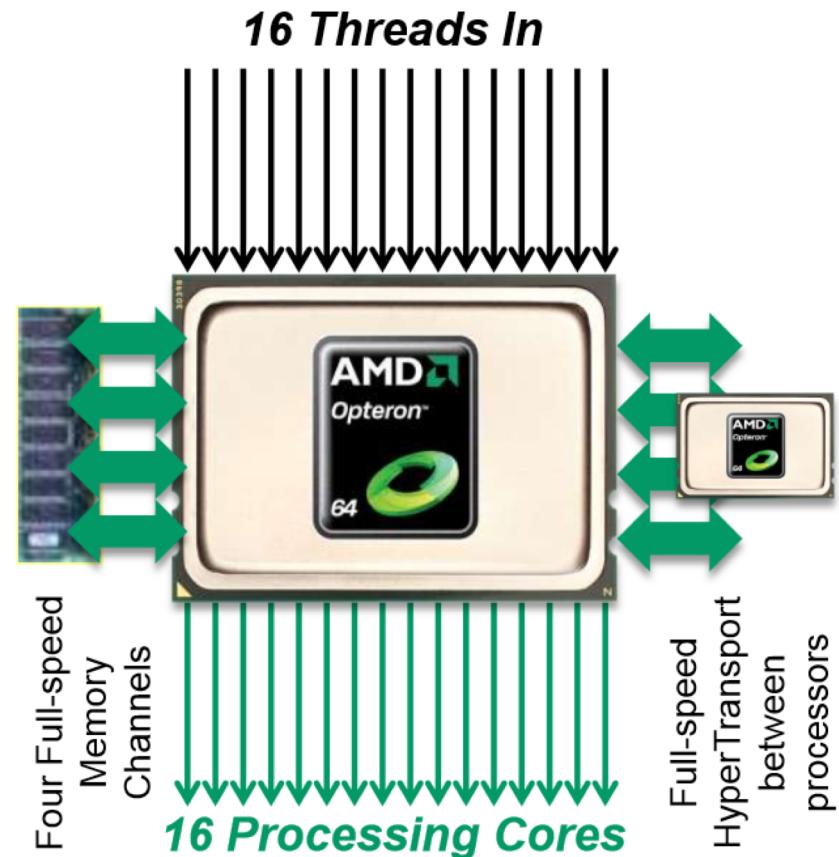
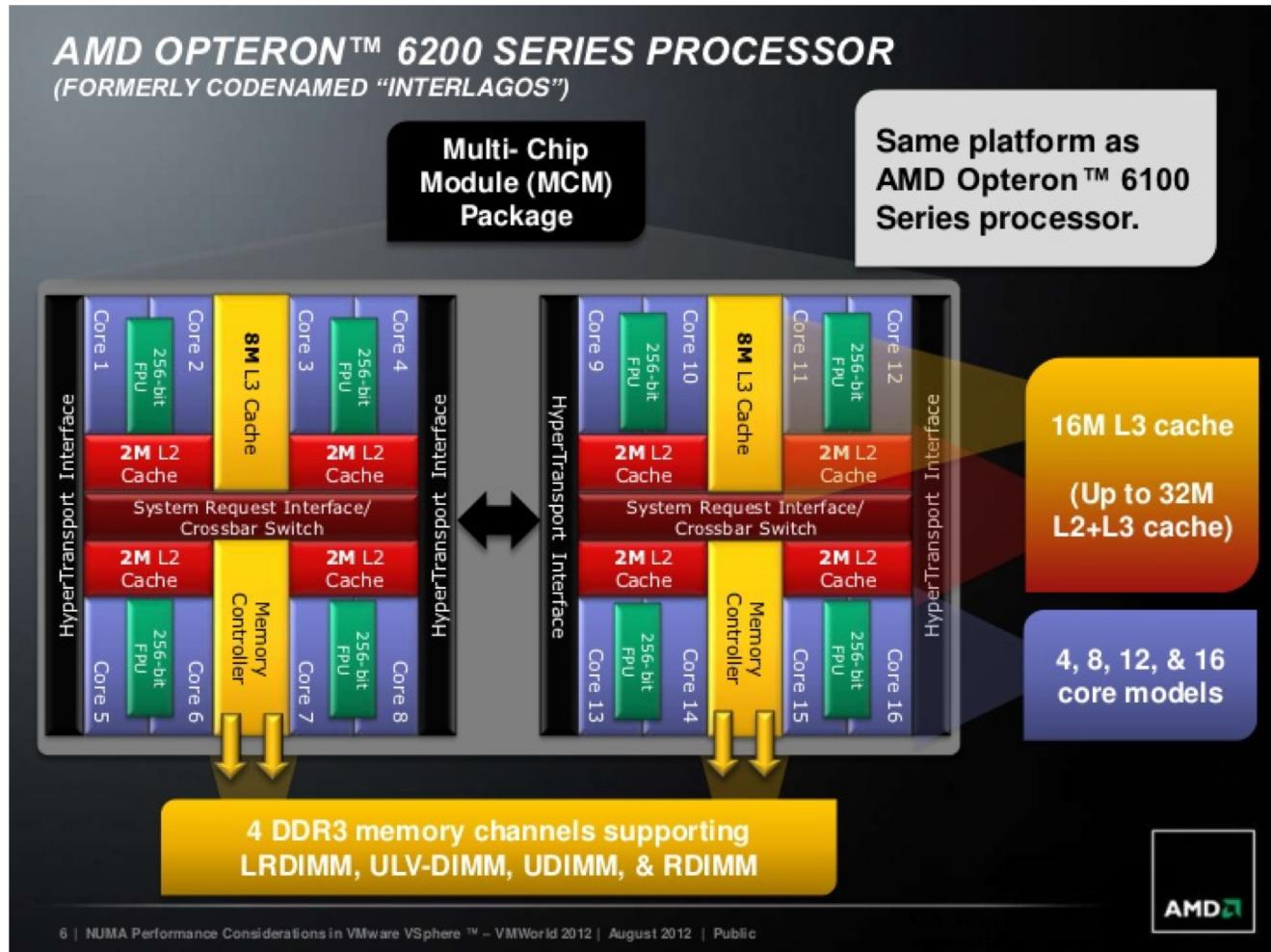


IBM Power 5 (2004)



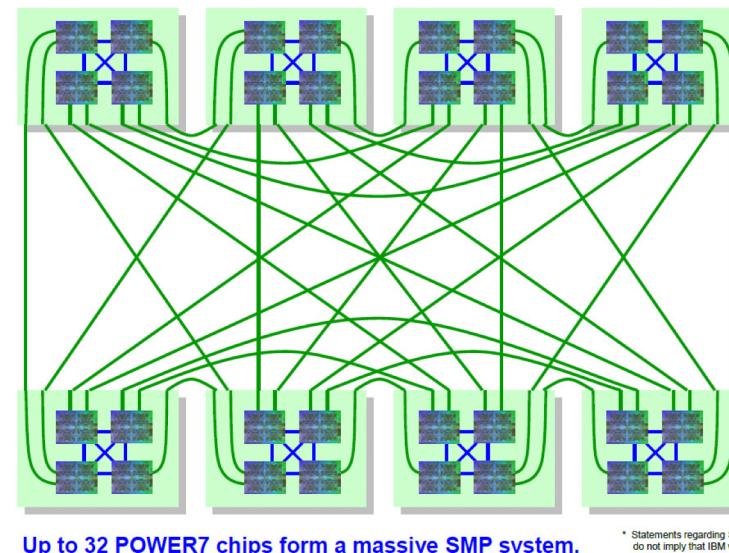
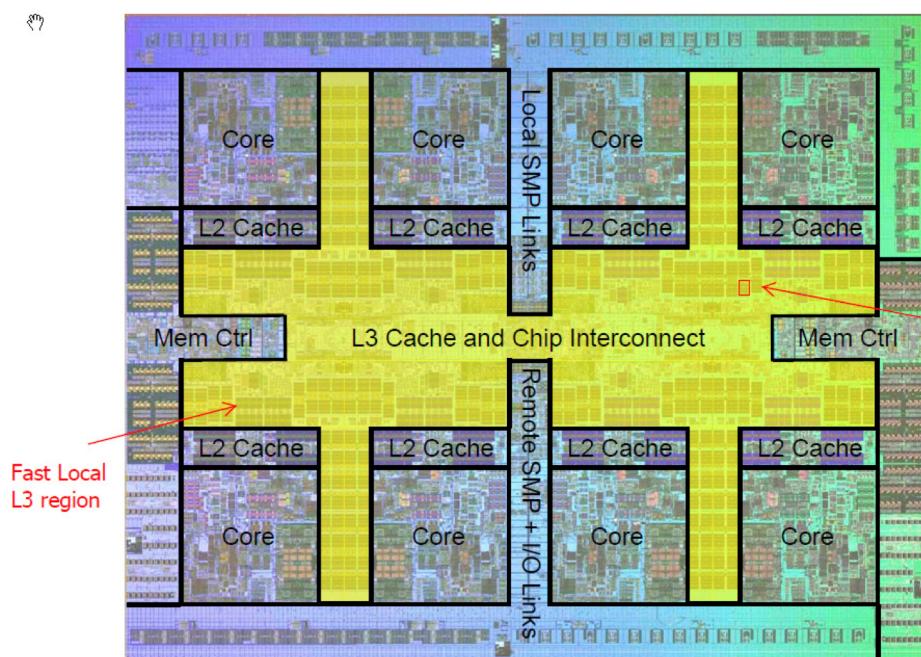
4º MULTIS EN PLACA SIN CHIPS DE INTERCONEXIÓN (GLUELESS)

AMD Opteron 6200 Series Processor, Interlagos (32 nm, Nov 2011)



4º MULTIS EN PLACA SIN CHIPS DE INTERCONEXIÓN (GLUELESS)

IBM Power 7, (45 nm, Feb. 2010)



Water-chilled Multiprocessor
With 4 POWER 7 Chips

The Implementation of POWER7: A Highly Parallel and Scalable Multi-Core High-End Server Processor. D. Wendel et al. ISSCC 2010

Eight quad-threaded cores are integrated together with two memory controllers and high-speed system links (both local and remote symmetric multiprocessing) on a 567mm² die, employing 1.2B transistors in 45nm CMOS SOI technology. High on-chip performance and therefore bandwidth is achieved using 11 layers of low-k copper wiring. The technology features deep trench [DT] capacitors that are used to build the 32MB embedded DRAM L3. Grafo completo de grado 7: 28 arcos $\rightarrow N(N-1)/2$

5º MANY-CORE WITH STACKED DRAM (Nov 2016)

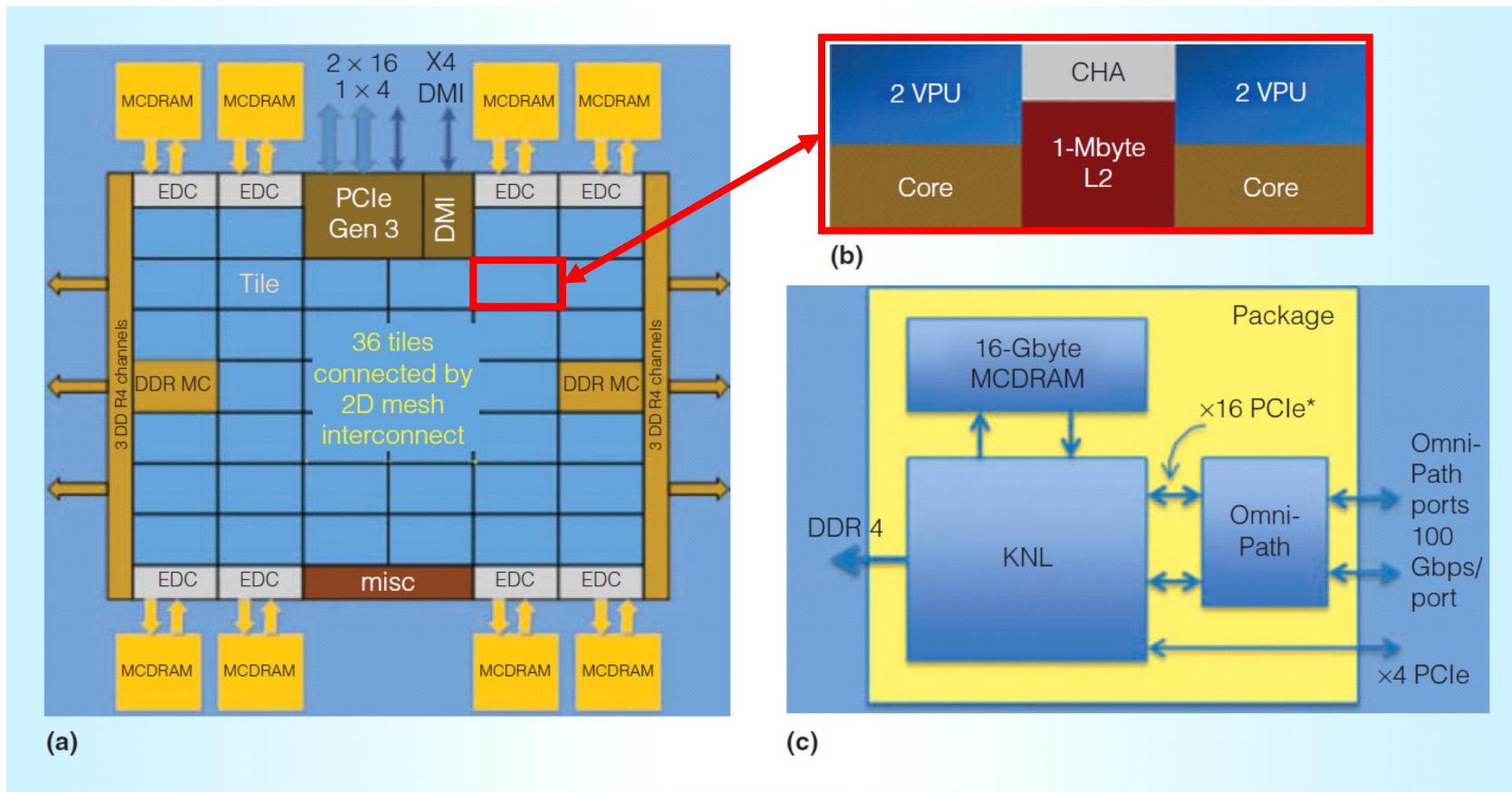


Figure 1. Knights Landing (KNL) block diagram: (a) the CPU, (b) an example tile, and (c) KNL with Omni-Path Fabric integrated on the CPU package. (CHA: caching/home agent; DDRMC: DDR memory controller; DMI: Direct Media Interface; EDC: MCDRAM controllers; MCDRAM: multichannel DRAM; PCIe: PCI Express; VPU: vector processing unit.)

Intel® Xeon Phi™ Processor KNL (Knight's Landing)

Specifications

Essentials	
Processor Number	7290F
Status	Launched
Launch Date	Q4'16
Lithography	14 nm
Recommended Customer Price	\$6401.00
Performance	
# of Cores	72
Processor Base Frequency	1.50 GHz
Max Turbo Frequency	1.70 GHz
Cache	36 MB L2
TDP	260 W
VID Voltage Range	0.550-1.125V

REDES DE INTERCONEXIÓN PARA MULTIPROCESADORES

- CONTEXTO
- CARACTERÍSTICAS
 - Coste y Energía
 - Topología
 - Modo de Funcionamiento
 - Disponibilidad
 - Prestaciones
 - Diferencias con Redes de Comunicaciones
- CONFLICTOS EN EL SUBSISTEMA RED CONMUTADA-MEMORIA
- Bus
 - Coste, Fases, Arbitraje, Implementación, Topología, Tolerancia a Fallos, Valor Añadido, Modo de Funcionamiento, Rendimiento, Otros usos
 - Conmutación de Circuitos
 - Conmutación de Paquetes
 - Recapitulación y mas términos
 - Implementación del protocolo BR/BG
- Enterprise 6000/6500 (1996, SUN Microsystems)
- Ejemplos de Buses COMERCIALES

CONTEXTO

MULTIPROCESADORES DE MEMORIA COMPARTIDA

- Conexión chip-memoria principal: canales DDR, RAMBUS, ...
- Conexión procesadores-memorias cache privadas- memoria cache compartida
 - Red en el multi-core chip (NoC, Network on Chip)
 - Red en el módulo multichip (MCM)
 - Red en el circuito impreso
- Conexión entre tarjetas de circuito impreso (p.e. SGI UV 300¹).
- Red de sistema
 - En el rack (entre placas de circuito impreso)

1. En un rack, hasta 32 sockets Intel Xeon E7-x800 v4 (hasta 24 cores por socket, hasta 1536 cores en total), hasta 40 TB de memoria compartida, red de sistema NUMALINK (todos con todos). Abril 2017.

MULTIPROCESADORES DE MEMORIA DISTRIBUIDA

- Conexión entre nodos independientes (Sistema Operativo, E/S, ...)
 - Un nodo puede ser un
 - Multi-core chip
 - Pocos multi-cores chip formando un multi de memoria compartida
- Uno o más nodos en una tarjeta
- Conexión entre tarjetas de circuito impreso
 - Redes de sistema
 - En el rack (entre placas de circuito impreso)
 - Entre racks

CARACTERÍSTICAS

COSTE

- Longitud total del cableado $L = \sum_{\forall \text{ camino } i} l_i \cdot w_i$

l: longitud del cable

w: anchura en bits del cable

- Número de comutadores o encaminadores
- Lógica de control y arbitraje para establecer conexiones

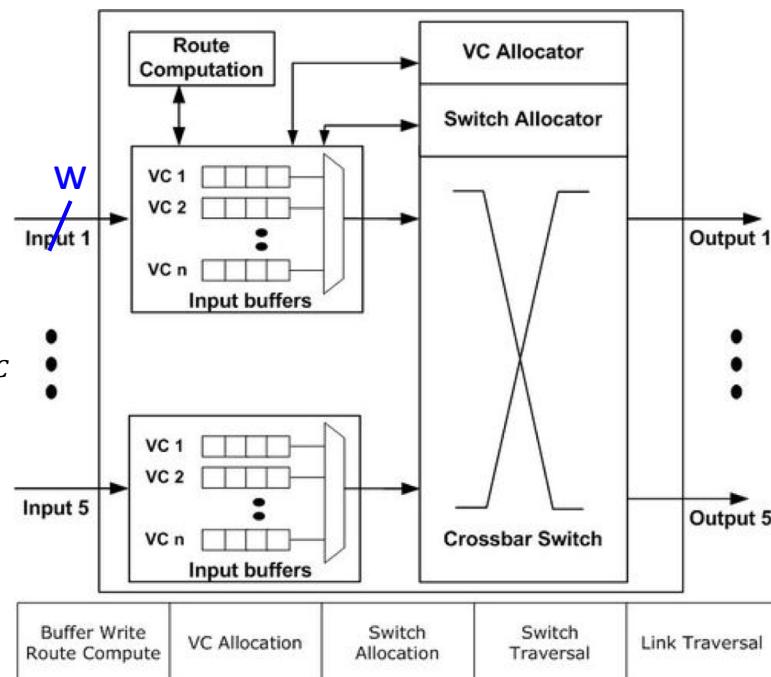
Parte importante del esfuerzo de diseño de un multiprocesador

En una red en chip, coste es igual a la superficie ocupada

1. **Phit:** *physical digit*. Se mide en bits o bytes y es el ancho de enlace **w**. Suele coincidir con el **flit**, que es la *unidad de control de flujo* entre encaminadores conectados (*flow-control unit*).

ENERGÍA

- Ejemplo: red en chip
 - Enlaces + encaminadores (*routers*) a frecuencia/tensión igual o menor que la de los cores
 - Energía necesaria para que un *phit*¹ atravesie un encaminador:
 - Energía dinámica para
 - Escribir y leer en el *input buffer*
 - arbitrar y atravesar el *crossbar*
 - Energía estática
 - Potencia cables
 - Dinámica:
 - $\alpha \cdot C_{link} \cdot V_{NoC}^2 \cdot f_{NoC}$
 - Estática ≈ 0
 - Parte respetable del total gastado por un chip multicore



TOPOLOGÍA

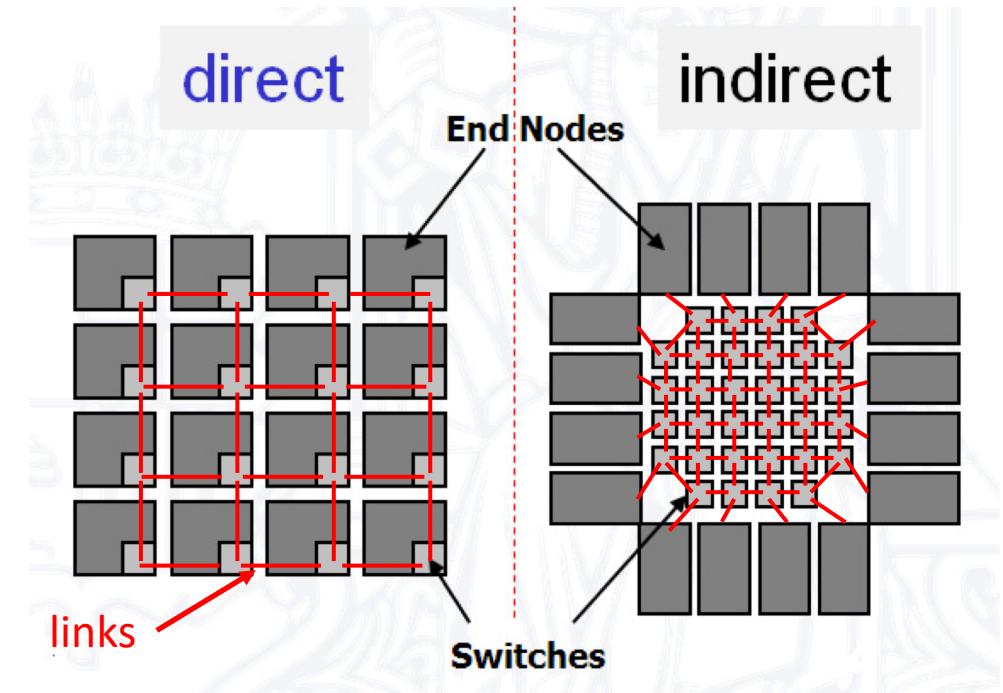
- Lista de parejas que pueden comunicar simultáneamente

Dinámica → red indirecta

- la lista es variable: se puede cambiar en cada “ciclo de red”
 - Familia de los buses: bus compartido ... *crossbar*
 - Familia de las redes multietapa
 - Se usa en multis de memoria compartida

Estática → red directa

- la lista es fija
 - Malla, toro, hipercubo, fat tree, dragonfly
 - Se usa en multis de memoria compartida y distribuida



Valor añadido

- **1 fuente → n destinos**
 - n=parte: *multicast*, difusión parcial
 - n=todos: *broadcast*, difusión
- **n fuentes → 1 destino** (y vuelta, 1 destino → n fuentes)
 - combinación de peticiones (ej. Fetch&add combining)

MODO DE FUNCIONAMIENTO

- La utilización de los caminos físicos (tramos de cables) para conseguir el enlace fuente-destino durante la comunicación
- **Transacción** = secuencia de **mensajes**, típicamente, un mensaje de **petición** y otro de **respuesta**. Un mensaje es un paquete único formado por una secuencia de *flits*¹.

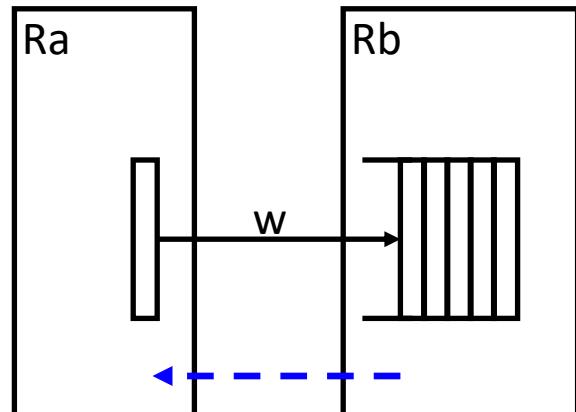
Comutación de circuitos

- El enlace fuente-destino existe durante toda la transacción, una vez establecido sirve para comunicar destino ↔ fuente
- Control de flujo sencillo, entre extremos
- Ancho de banda *poco aprovechado*: un circuito puede impedir la formación de otros ...
- Latencias pequeñas para transacciones con mucho mensajes

Comutación de paquetes

- El enlace fuente-destino se va formando y deshaciendo junto con el movimiento del mensaje
- Control mas complejo
 - Ej. Tras fallo de cache, una petición de bloque de cache va ocupando y liberando tramos de la red (+ control de flujo entre routers) hasta llegar al banco de memoria destino
- Ancho de banda *mejor aprovechado*
- Latencia mas grande: unos pocos ciclos por router atravesado

1. *Flit: flow-control unit.* Control de flujo.
Suponemos dos routers **R_a** y **R_b** conectados mediante enlace directo según la figura.
R_a puede colocar un flit si el buffer en **R_b** tiene espacio suficiente.
El “control de flujo” se encarga de controlar la inyección de *flits* para no desbordar el buffer.

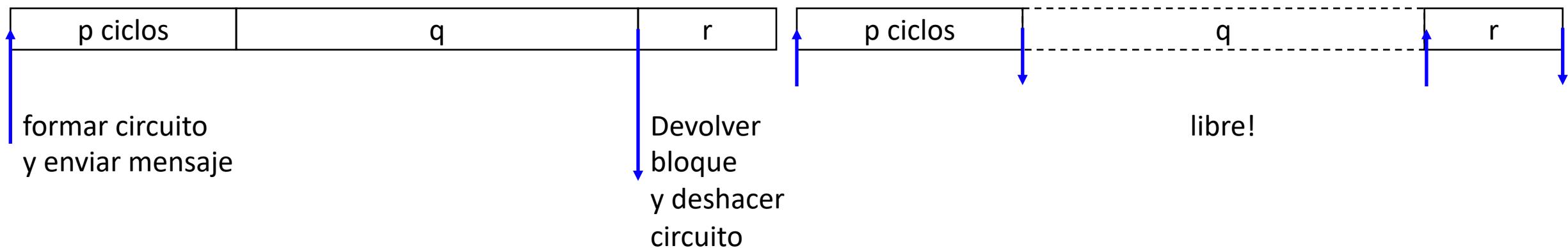


EJEMPLO

Load r4, @x' → miss (desde cache C_i a memoria M_j)

CC: $C_i > M_j$ (rB, @x);
lectura bloque x en M_j ;
(ack, x)

CP: $C_i > M_j$ (rB, @x);
lectura bloque x en M_j ;
 $M_j > C_i$ (ack, x)



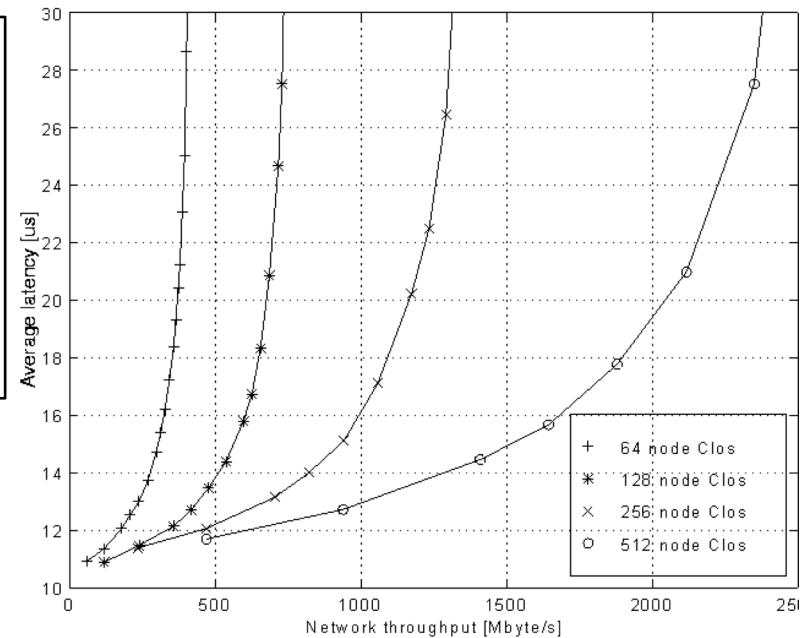
El enlace físico P_i-M_j se mantiene durante $p+q+r$ ciclos

DISPONIBILIDAD, PRESTACIONES, DIFERENCIA CON REDES DE COMUNICACIONES

- Disponibilidad
 - Enlaces redundantes o de repuesto
 - Caminos múltiples (*path diversity*)
 - Mensajes protegidos con códigos detectores y correctores de error
- Prestaciones
 - Ancho de banda:
 - Crudo, pico, medio, ...
 - $6 \times 10^9 \text{ Bytes/segundo} = 6 \text{ GB/s} = 5.59 \text{ GiB/s}$
 - $1 \times 2^{30} \text{ bits/segundo} = 1 \text{ GiB/s} \approx 1,074 \text{ Gb/s}$
 - Latencia:
 - Depende de la carga de la red: mínima, media, ...
 - Se mide en unidades de tiempo

Latency vs. Throughput for Clos networks under random traffic with 64 Byte packets

Haas, Thornley, Zhu, Dobinson, Heeley, and Martin. "Results from the Macramé 1024 Node IEEE 1355 Switching Network."

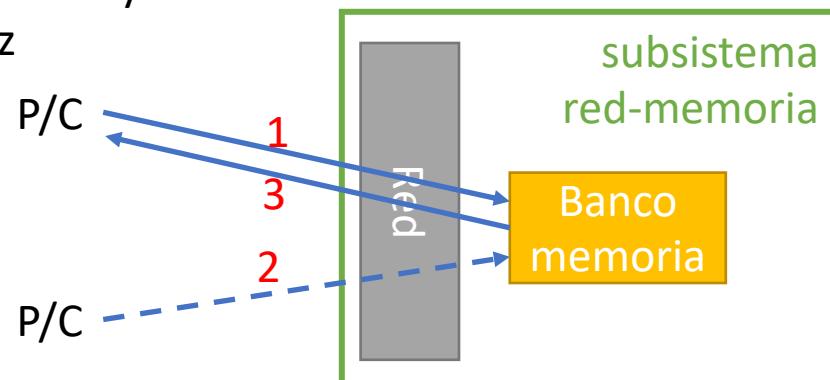


- Diferencia con redes de comunicaciones
 - Las distancias físicas son más pequeñas, el menor número de componentes implicados es menor, las interferencias electromagnéticas están mas acotadas ... por todo ello es normal asumir una tasa de errores muy pequeña.
En consecuencia no suelen existir mecanismos de retransmisión, como mucho se añaden bits de paridad o de ECC

CONFLICTOS EN EL SUBSISTEMA RED-MEMORIA

- **CONFLICTOS**: serializan la actividad paralela por el uso de un recurso compartido único
→ se produce **CONTENCIÓN** (de contienda, pelea)
 - Dos o mas peticiones al mismo banco de memoria (1 puerto)
 - Su probabilidad depende de:
 - nº de bancos
 - distribución de los datos y el código (programador, compilador, SO)

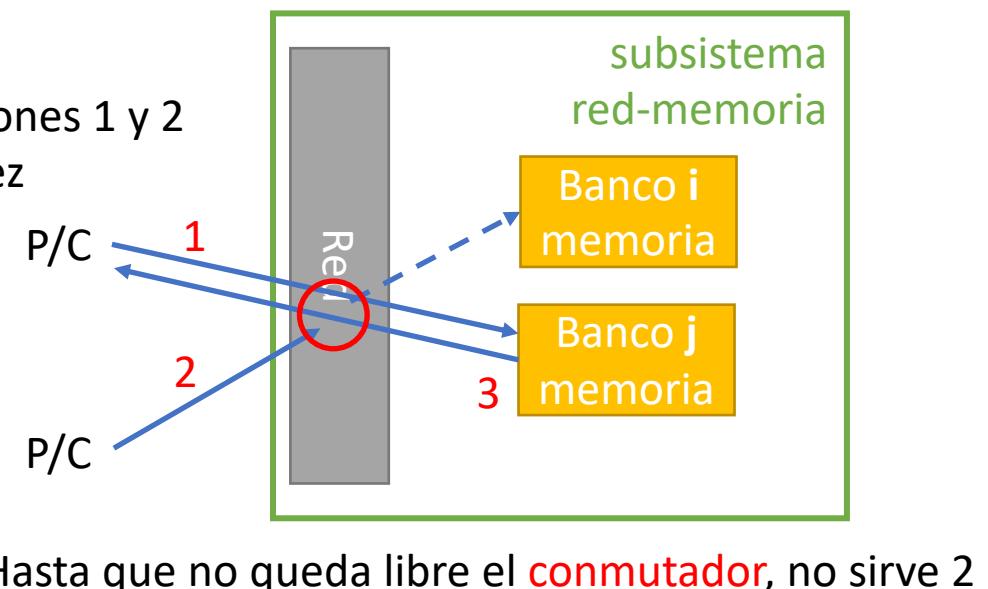
Peticiones 1 y 2
a la vez



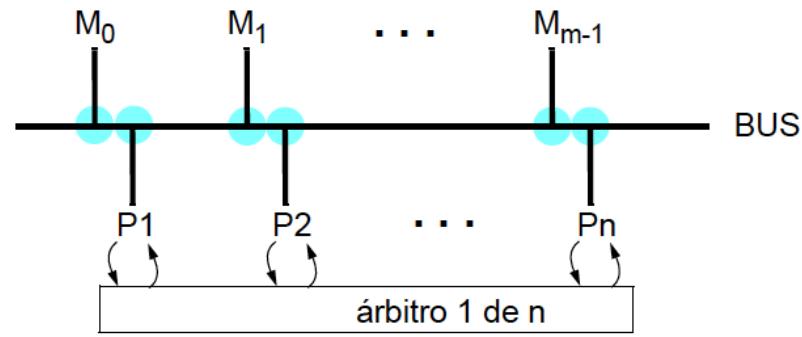
Hasta que no queda libre el banco no sirve 2

- Dos o mas peticiones usan el mismo recurso de red (caminos o *routers*) → DEGRADACIÓN
 - La red “degrada” el rendimiento del sistema
 - Probabilidad de conflicto depende del tipo de red

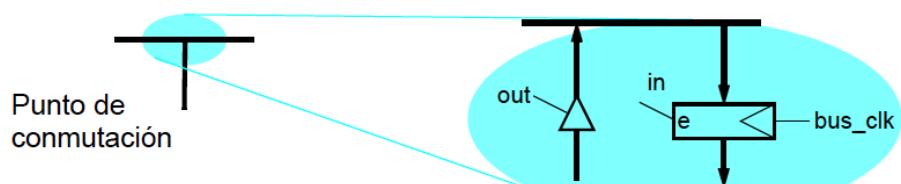
Peticiones 1 y 2
a la vez



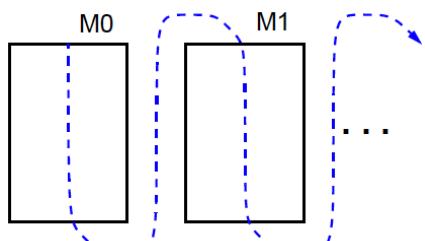
BUS (SHARED BUS, SINGLE BUS)¹



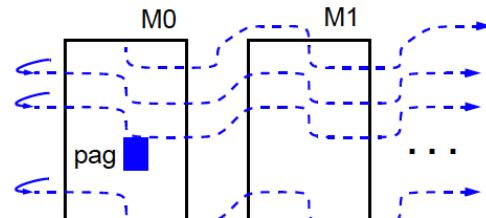
Conexión para envío (out) y recepción (in) en un bus síncrono



Entrelazado de memoria con los bits de mas peso de la @física



Entrelazado de memoria por páginas
de SO: $n^{\text{o}}\text{banco} = n^{\text{o}} \text{pagina mod } m$



- COSTE

- $M+n$ puntos de conmutación
- Árbitro "1 de n "
- Ancho de los caminos (datos, @, control) + alientación y masa,
- frecuencia

1. Principles and practices of interconnection networks.
Dally & Towles. Elsevier- Morgan-Kaufmann 2004.

BUS (SHARED BUS, SINGLE BUS) (II)

FASES DE UN MENSAJE

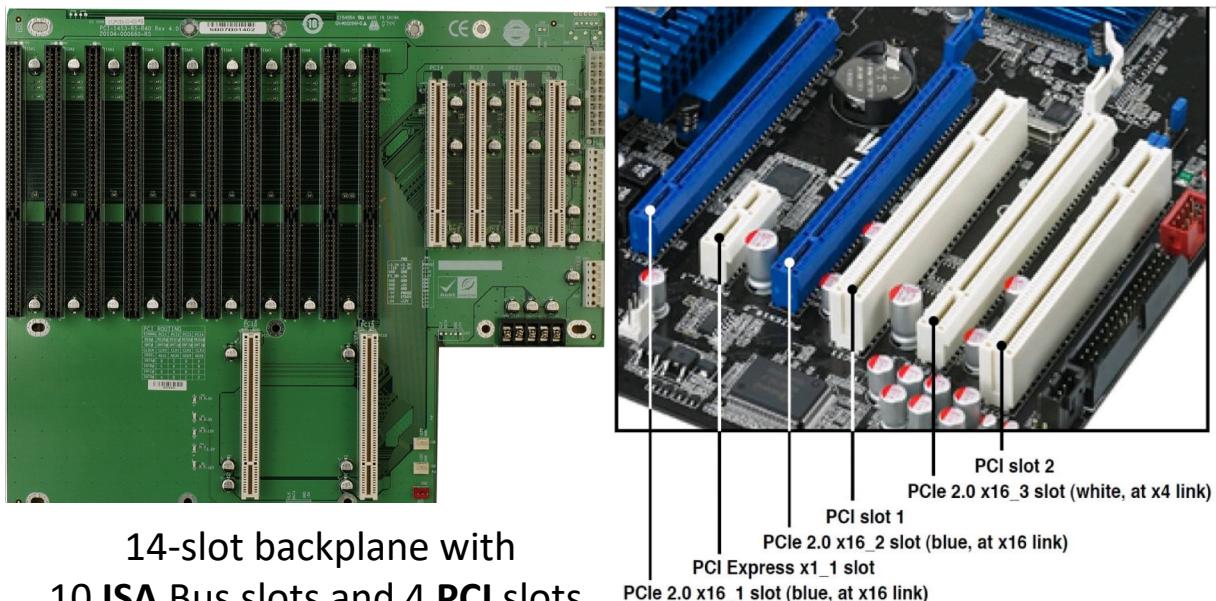
- *Arbitrar* para obtener la propiedad del bus (MASTER)
- *Direccionar* el componente destino (SLAVE)
- *Transferir*
 - En un bus síncrono cada fase ocupa unos ciclos de reloj de bus

ARBITRAJE

- El de la figura anterior es centralizado, pero puede ser distribuido
- Prioridad fija, rotativa, aleatoria, Least-Recently-Granted

IMPLEMENTACIÓN DE LOS CABLES ELÉCTRICOS

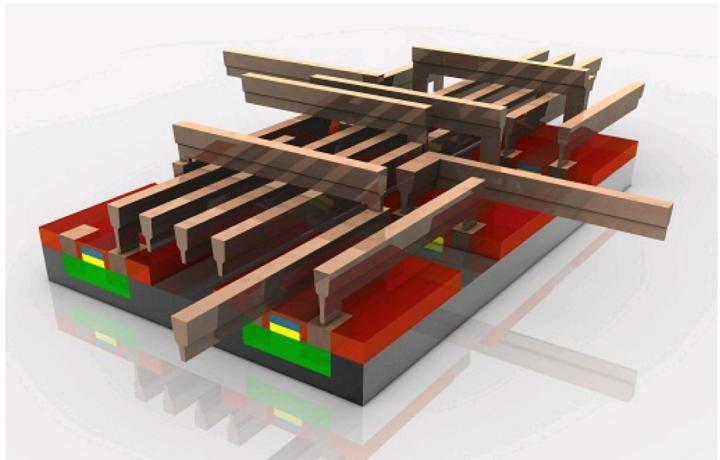
- Pistas de cobre en una placa de circuito impreso *backplane*, *middleplain*, ...



PCI is a bus used to connect peripherals to different types of computer systems (**I/Obus**), from embedded to large server systems. It is synchronous and has multiplexed address/data lines. Maximum configuration: 64-bit datapath and 133 MHz.

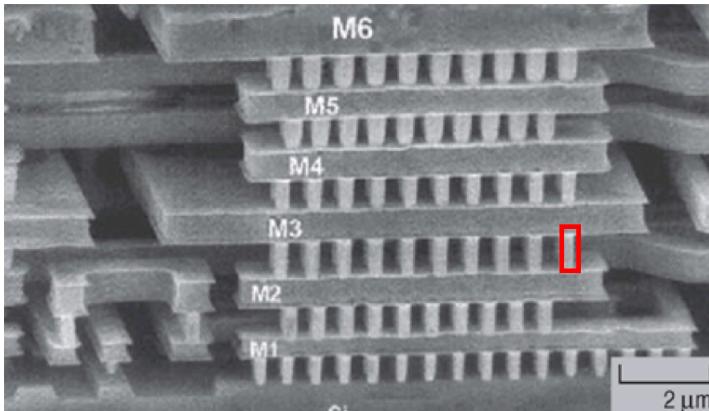
BUS (SHARED BUS, SINGLE BUS) (III)

- Niveles de metal (aluminio o cobre) en el interior del circuito integrado



- From Sand to Silicon: The Making of a Chip

- A cross-section of a six-level metal on-chip interconnect (electron microscope image)
Electrical connection between the metal layers is provided by vertical interconnects referred to as **vias**



- The Making of a Chip with 22nm/3D Transistors

BUS (SHARED BUS, SINGLE BUS) (IV)

TOPOLOGÍA DINÁMICA

- Permite enviar un mensaje entre *cualquier* pareja de componentes
- Pero dos mensajes no pueden enviarse en paralelo, deben serializarse: máxima *degradación*

TOLERANCIA A FALLOS

- No

VALOR AÑADIDO

- Difusión gratis: coste mensaje punto a punto = coste mensaje a destino múltiples
- Serialización: solo un componente puede transmitir
- Los protocolos de coherencia cache (ya veremos) aprovechan ambas cualidades
 - Algunas instrucciones de escritura difunden su dirección
 - Todos los mensajes se ven en el mismo orden por todos los componentes

MODO DE FUNCIONAMIENTO

- Comutación de circuitos
 - Solo puede funcionar un banco de memoria
 - Sencillo
 - Hay que establecer un camino inicio-fin (señalizar)
 - Bueno para “pocos” mensajes “largos”
- Comutación de paquetes
 - Pueden trabajar varios bancos de memoria a la vez
 - Complejo
 - El camino se hace y deshace a tramos
 - Bueno para “muchos” mensajes “cortos”

BUS (SHARED BUS, SINGLE BUS) (V)

MODO DE FUNCIONAMIENTO (CONT.)

- Protocolo:
determina las fases de un mensaje,
su duración en ciclos,
las condiciones excepcionales, ...
para todas las **transacciones** posibles: lectura,
escritura, etc.
- **T_lectura:** leer el bloque que empieza en la
dirección **@x**:
 - Arbitrar,
 - Mensaje ida con **@x**,
 - Mensaje vuelta con bloque **x** = 1 o mas phits¹ para **x**
- **T_escritura:** escribir el bloque **x** a partir de la
dirección **@x**:
 - Arbitrar,
 - Mensaje único con **(@x,x)** = 1 o mas phyts para **x**

RENDIMIENTO

- Se mide en *ancho de banda* = $\frac{\text{bytes}}{\text{tiempo}}$ o $\frac{\text{bits}}{\text{tiempo}}$
p.e. 10 GB/s = 10^9 Bytes/s
- Se distinguen varios anchos de banda:
 - Curdo, pico o máximo, sostenible, ...
- Limitaciones:
 - Más frecuencia ↔ menor tamaño físico,
menos procesadores y
memorias en el bus

1. **Phyt:** *physical transfer unit*. Cuando nos interesamos por los niveles mas bajos de una red con conmutación de paquetes, es útil hablar de phyts y de flits. El término *phyt* se refiere a los bits que se transfieren, en un ciclo del reloj, entre los dos extremos de un camino; normalmente es el ancho de dicho camino. El término *flit (flow-control unit)*, en cambio, se usa para la unidad de información que maneja el control de flujo entre los dos extremos de un enlace, o sea, es una colección de uno o mas phyts consecutivos. En un bus, una transacción de lectura empieza con un mensaje a la memoria comunicándole la dirección (un phyt de unos 30-40 bits) y continúa con un mensaje de devolución del bloque, que puede tener por ejemplo cuatro phyts de 16B, si estamos en un bus con una anchura del camino de datos de 16B. El flit de este mensaje tendría 64B.

BUS (SHARED BUS, SINGLE BUS) (VI)

OTROS USOS

- Dentro de un procesador
 - Conexiones en la ruta de datos, p.e. entre ALU's y bancos de registros
- En un encaminador o conmutador de comunicaciones
 - Para comunicar tarjetas de red
- En el subsistema de E/S
 - Para comunicar el procesador o la memoria con dispositivos de E/S: [ISA](#), [PCI](#), [PCIe](#), [SCSI](#), ...
- En un sistema en chip (SOC, system-on-chip)
 - Para conectar diferentes procesadores (RISC propósito general, VLIW, DSP), aceleradores (gráficos, criptografía, video, ...) y periféricos (ethernet, JTAG, Wireless ...)

NOTAS HISTÓRICAS

A partir de 1985 introducción comercial de los multis de memoria compartida basados en bus. Ofrecen un número respetable de micros de 32 bits, junto con un sistema operativo UNIX.

Marca	Modelo	Año	Procesador	F_{cpu} (MHz)	N_{max}	Caches
Encore ^a	Multimax	1985	NS-32032	30	20	no
Alliant ^a	FX/8	1985	varios ^a	5,8	8	Varias
Sequent ^{ab}	Balance 2100	1986	National Semiconductor NS-32032	10	30	Write-through
	Symmetry S-81	1981	Intel-80386-80387	16	30	Copy-back

- a. **8 Computational elements**, included a set of Witeck 1064/1065 FPU's and several custom designed support chips to implement a Vector Processor based on the Motorola 68000 architecture.
Interative processor. Motorola 68008's (and, subsequently, Motorola 680101's and 68020's)
- b. Marcas extinguidas hace ya tiempo

BUS (SHARED BUS, SINGLE BUS) (VII)

ALLIANT FX/8

The computational complex of the FX/8 contains 8 CEs. When executing concurrency instructions, the CEs communicate via a concurrency control bus.

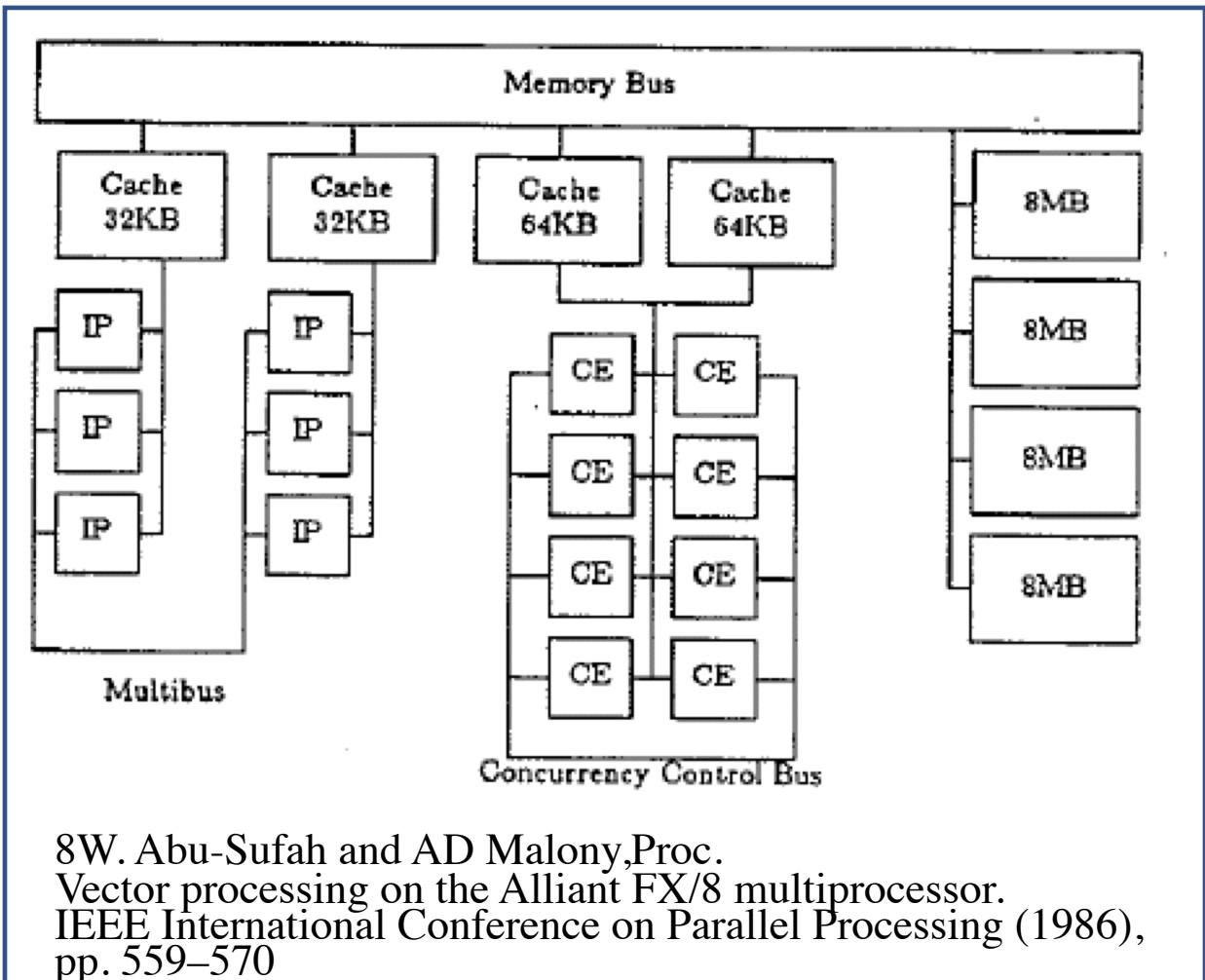
Each CE has a computational clock period of 170 nsec with a peak execution rate of 11.8 MFLOPS and 5.9 MFLOPS for single and double precision computation, respectively.

With the 8 CEs working concurrently, the FX/8 advertised peak performance is 47.2 MFLOPS for double precision computations.

The CEs are connected by a crossbar switch to a direct-mapped, write-back, shared cache of 16K double precision words(c). The cache is implemented in 4 quadrants with a peak interleaved bandwidth to the CEs of 47.125 MW /sec.

It is connected to a 4 MW (32 MB) shared memory via a bus with a peak bandwidth of 23.5 MW /sec.

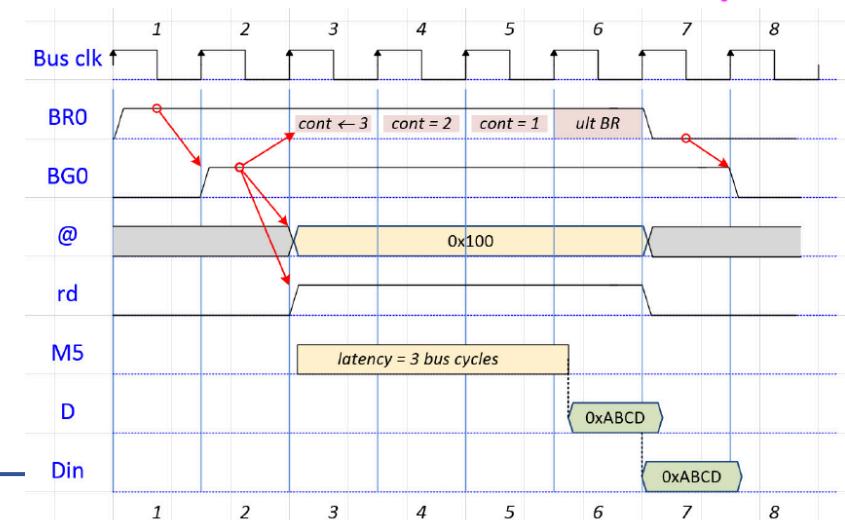
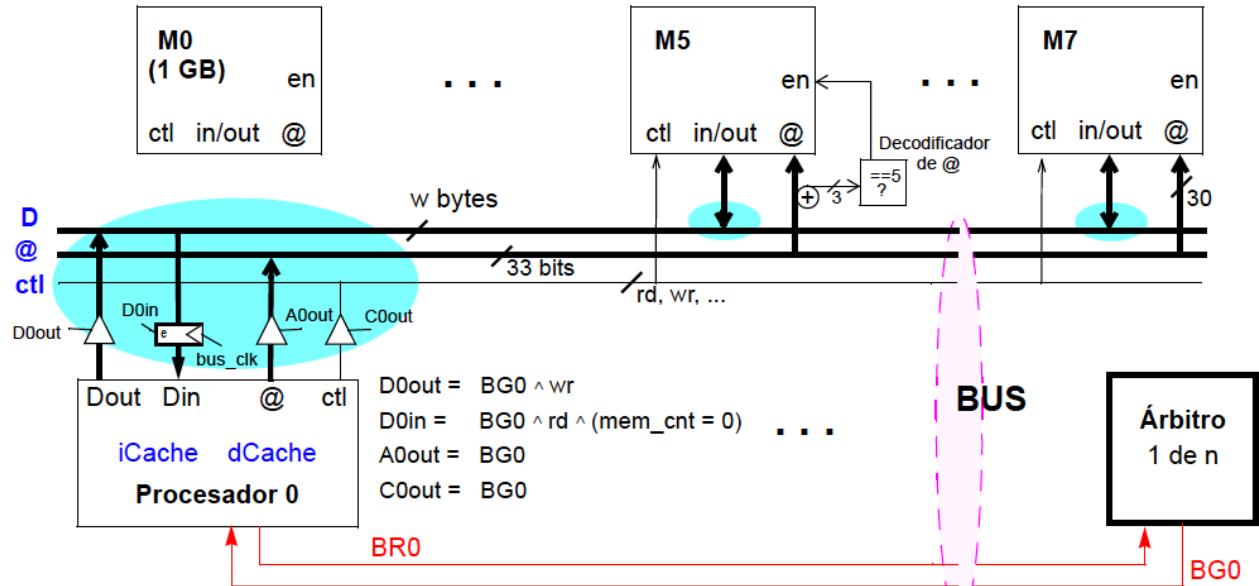
The system also contains 6 interactive processors (IPs) connected to their own caches as shown in Figure 1. The IPs primarily perform operating system related functions and I/O operations.



8W. Abu-Sufah and AD Malony, Proc.
Vector processing on the Alliant FX/8 multiprocessor.
IEEE International Conference on Parallel Processing (1986),
pp. 559–570

BUS EN CONMUTACIÓN DE CIRCUITOS

- El árbitro concede a la vez todos los caminos del bus, direcciones y datos. Aquí suponemos un árbitro centralizado.
- Se forma un circuito eléctrico que permanece durante toda la transacción.
 - Las transacciones se hacen de forma *atómica*. Su orden depende del árbitro.
- Solo los procesadores pueden tener la propiedad del bus (*master*). Las memorias se limitan a seguir órdenes (*slave*)
- Protocolo sencillo: $\langle Br^n, BG, @ \rangle$, además, $\langle Br^{\text{último}} = \text{último ciclo transferencia} \rangle$



EJEMPLO I

CICLO 1, P1 QUIERE LEER BLOQUE EN M3(DIR. @A)

CICLO 6, P2 QUIERE LEER BLOQUE EN M5(DIR. @B)

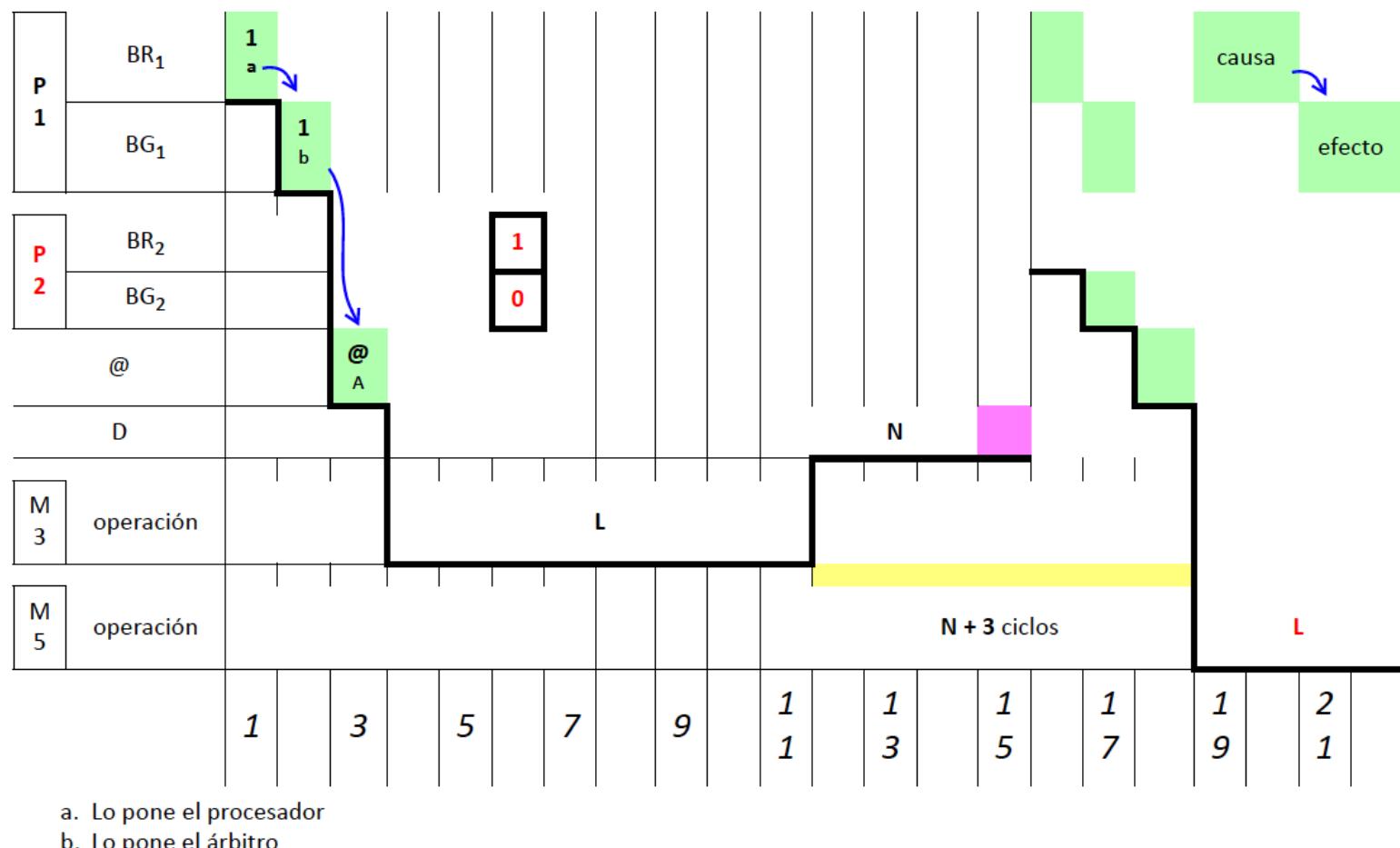
- L=8. Latencia de memoria: 8 ciclos de bus
- N=4. Transferencia: 4 physts = 4 (tamaño de bloque / ancho del camino de datos)

Prestaciones:

$$BW \text{ crudo} =$$

$$BW_{cc} = \quad = \quad \cdot BW_{crudo}$$

$$\min LAT_{cc} =$$



- a. Lo pone el procesador
b. Lo pone el árbitro

EJEMPLO II

P1 LEE BLOQUE EN M3(DIR. @A)

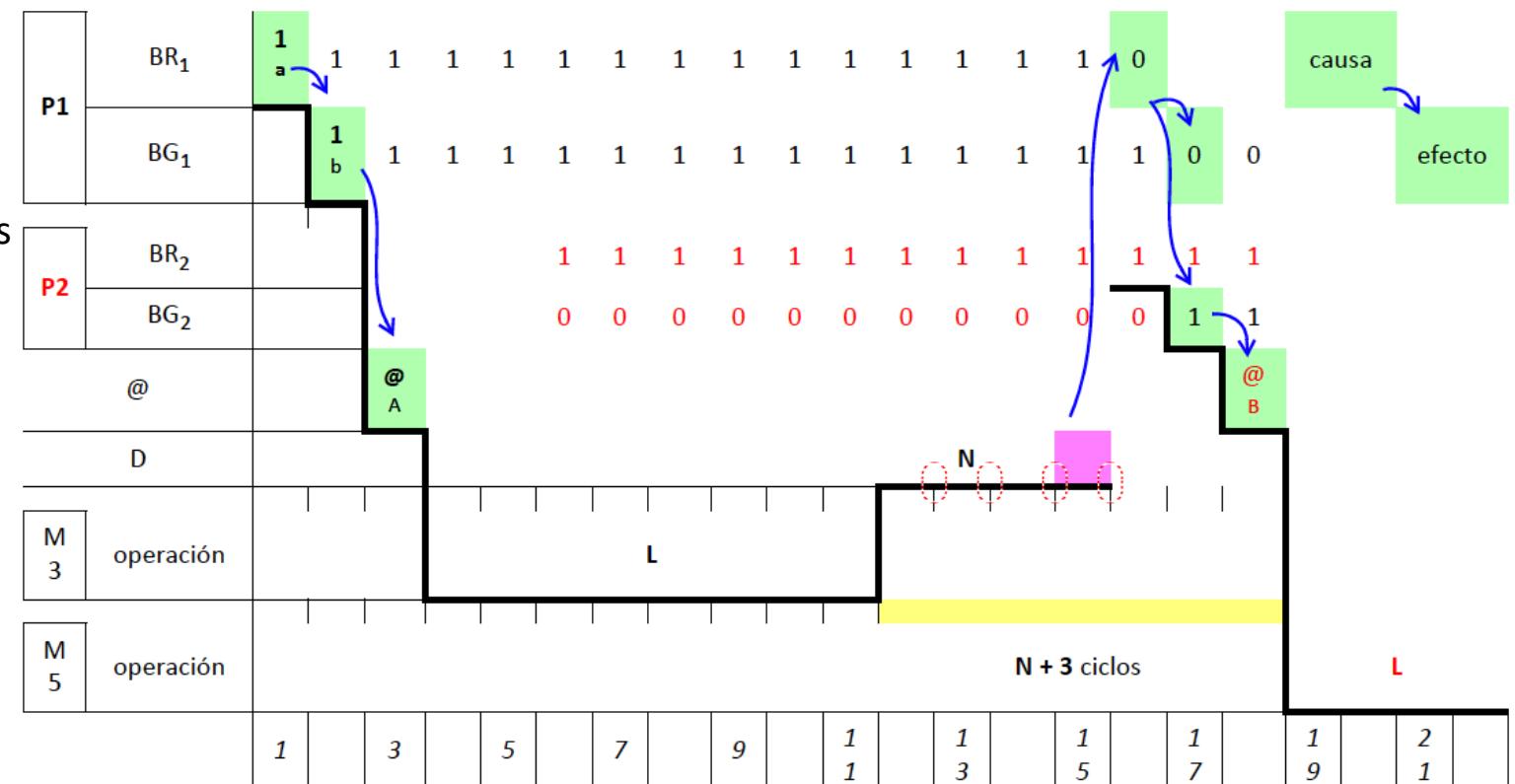
P2 LEE BLOQUE EN M5(DIR. @B)

- L=8. Latencia de memoria: 8 ciclos de bus
- N=4. Transferencia: 4phyts = 4 (tamaño de bloque / ancho del camino de datos)

Prestaciones:

$BW_{crudo} =$

$BW_{cc} = \dots = \dots \cdot BW_{crudo}$



- a. Lo pone el procesador
b. Lo pone el árbitro

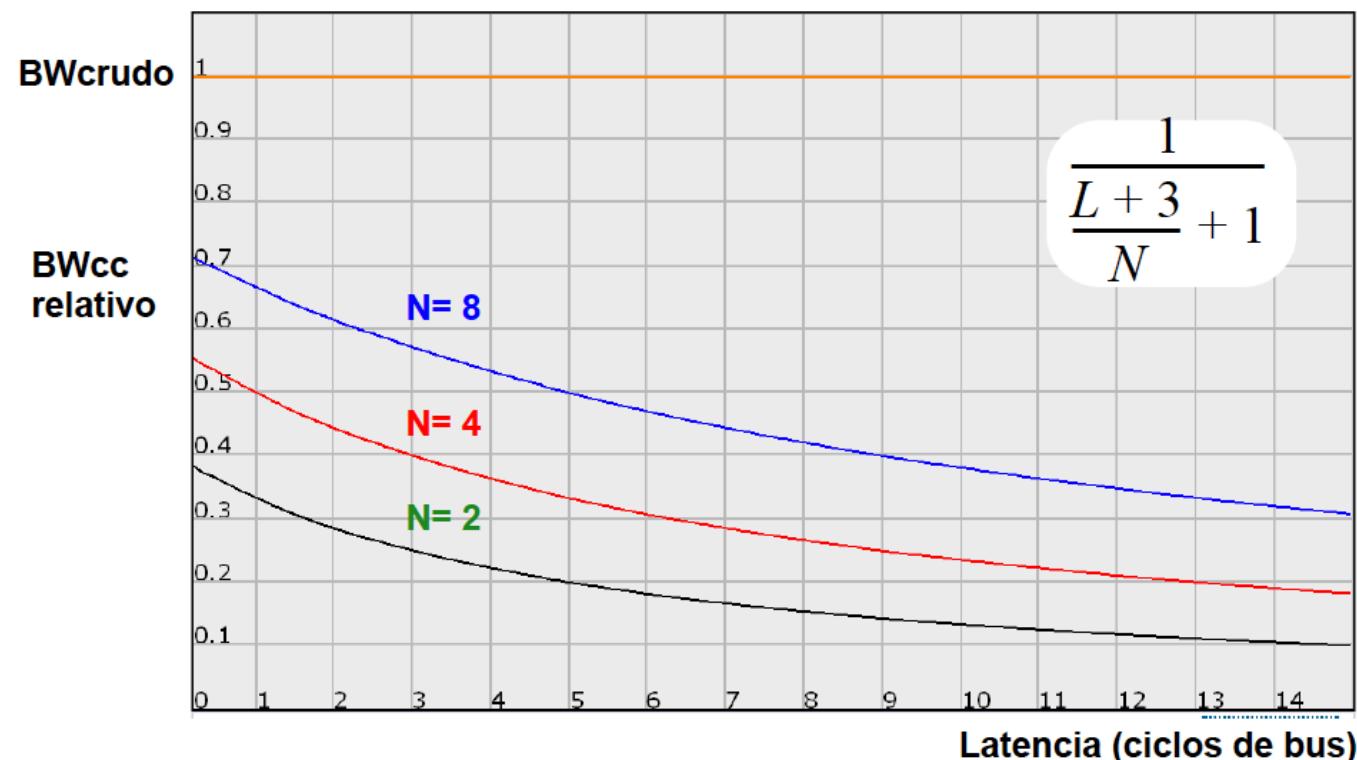
$minLAT_{cc} =$

EJEMPLO III. PRESTACIONES

$$BW_{crudo} = w \cdot f_{bus}$$

$$BW_{cc} = \frac{Bsize}{L + N + 3} \cdot f_{bus} = \frac{1}{\frac{L+3}{N} + 1} \cdot BW_{crudo}$$

$$\min LAT_{cc} = 3 + L + N$$



BUS EN CONMUTACIÓN DE PAQUETES

Split-transaction Bus o Bus de transacción partida:

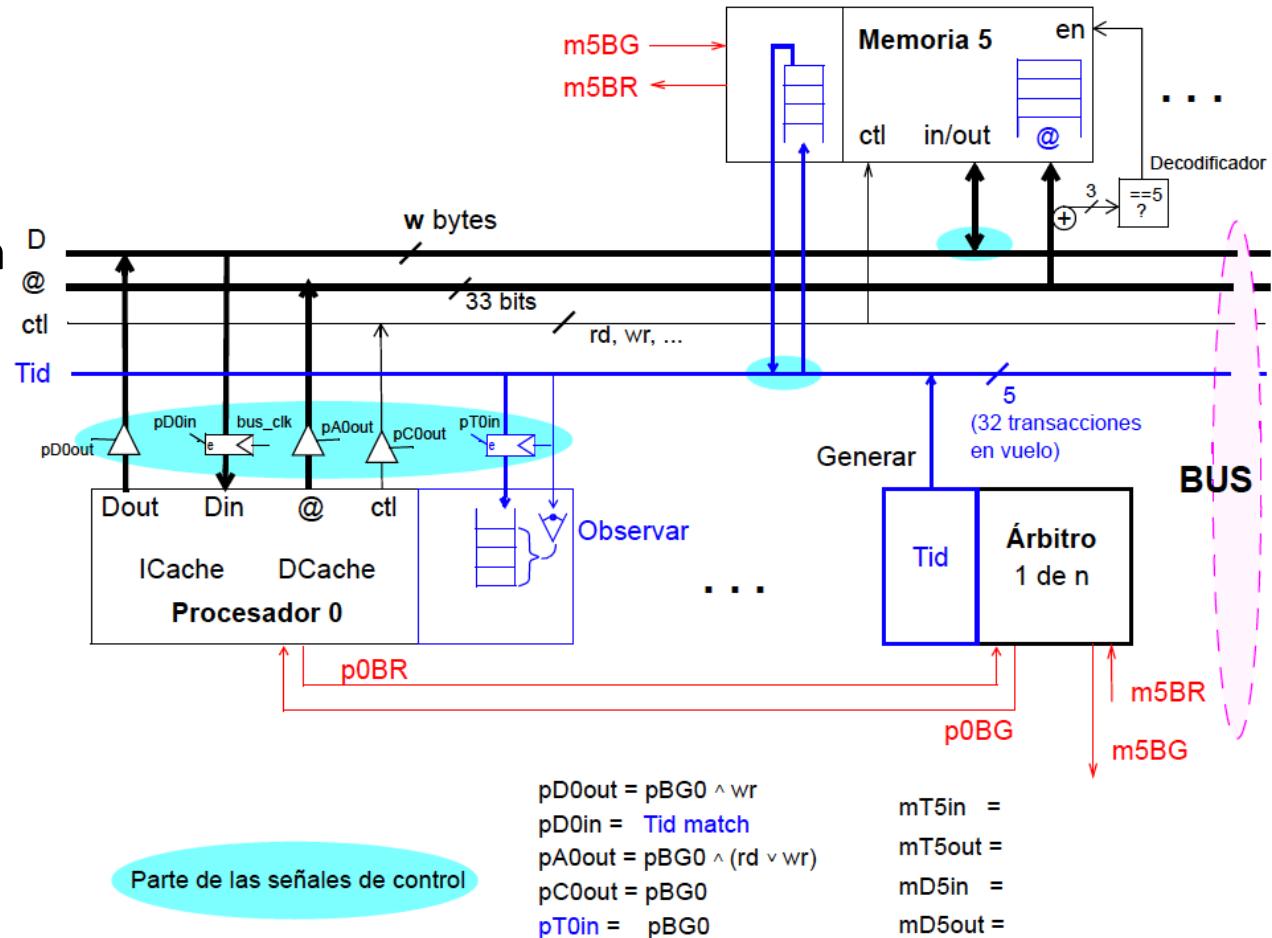
En la transacción de lectura de bloque se desacopla el envío de dirección y la recepción del bloque

La propiedad de las líneas de bus se cede después de cada mensaje. En buses no multiplexados, se arbitran por separado, y en tiempos diferentes, los caminos de dirección y de datos.

- Se guardan peticiones en los bancos de Mp (*buffering*)
- Un procesador capaz de producir varios fallos de cache antes de parar la ejecución, puede recibir los bloques de memoria en "desorden", o sea, en un orden distinto al de petición

Puede resolverse este problema con un identificador de transacción (Tid) de unos pocos bits gestionado por el árbitro.

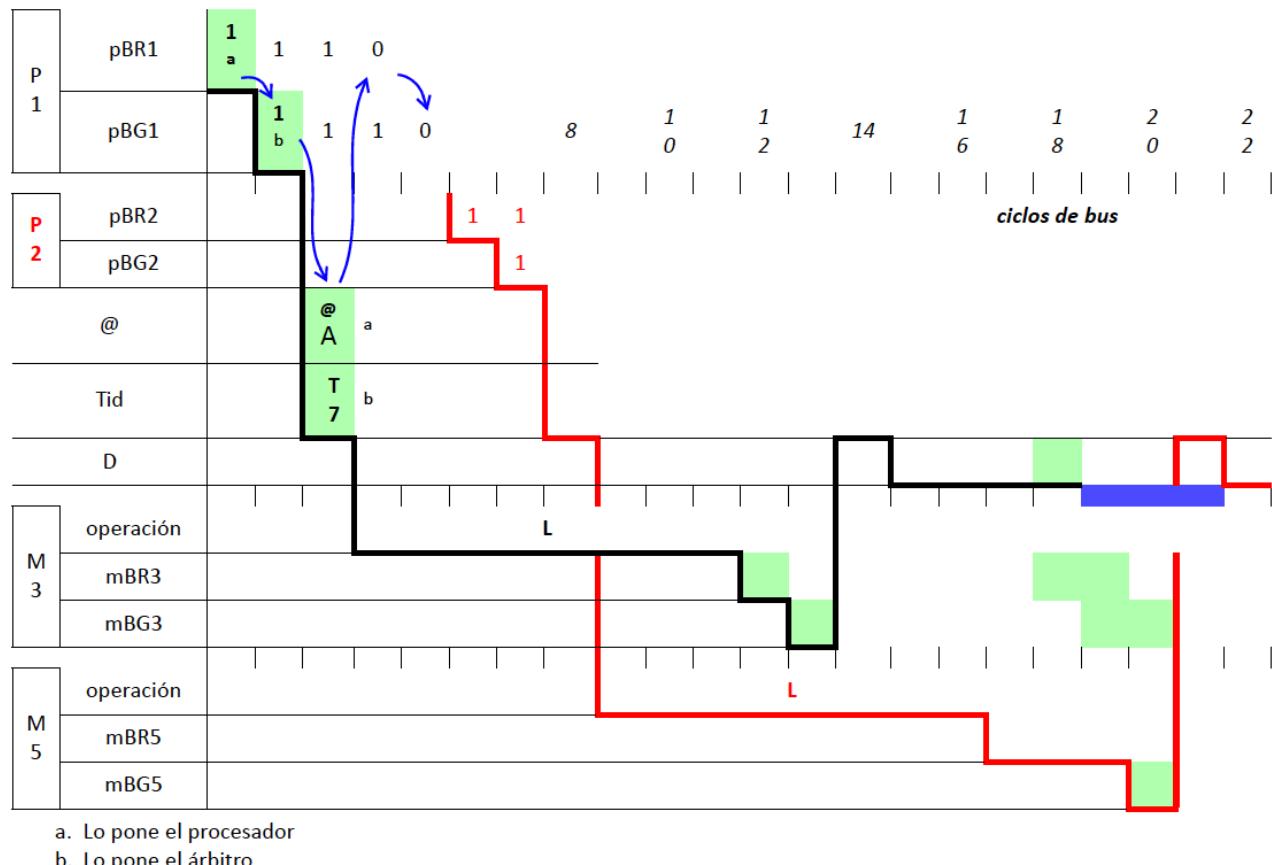
Hay pues un nº máximo de transacciones pendientes.



TRANSACCIÓN DE LECTURA:

- Mensaje ida con dirección de memoria + mensaje vuelta con datos:
 - Mensaje de ida: procesador pide bus (p0BR); el árbitro concede bus (p0BG) y produce un identificador de transacción nuevo (Tidx); el procesador guarda Tidx, coloca @ y se retira *hasta que aparezca de nuevo la misma Tidx* (**Observar**)
 - Mensaje de vuelta: memoria pide bus (m5Br): el árbitro concede (m5BG); la memoria coloca en el bus el identificador (Tidx) y los datos en una secuencia de physts; el procesador correspondiente, observa coincidencia con Tidx (**Tid match**) y retiene los physts de datos; el árbitro recicla Tidx para un uso futuro.
- Protocolo sencillo: <BRⁿ, BG, @-Tid>, además, <Br^{último} = último ciclo transferencia>

Ej. P1 LEE BLOQUE EN M3 (@A) Y P2 QUIERE LEER BLOQUE EN M5 (@B)



EJEMPLO I

P1 LEE BLOQUE EN M3(DIR. @A)

P2 LEE BLOQUE EN M5(DIR. @B)

PRESTACIONES:

$BW_{crudo} =$

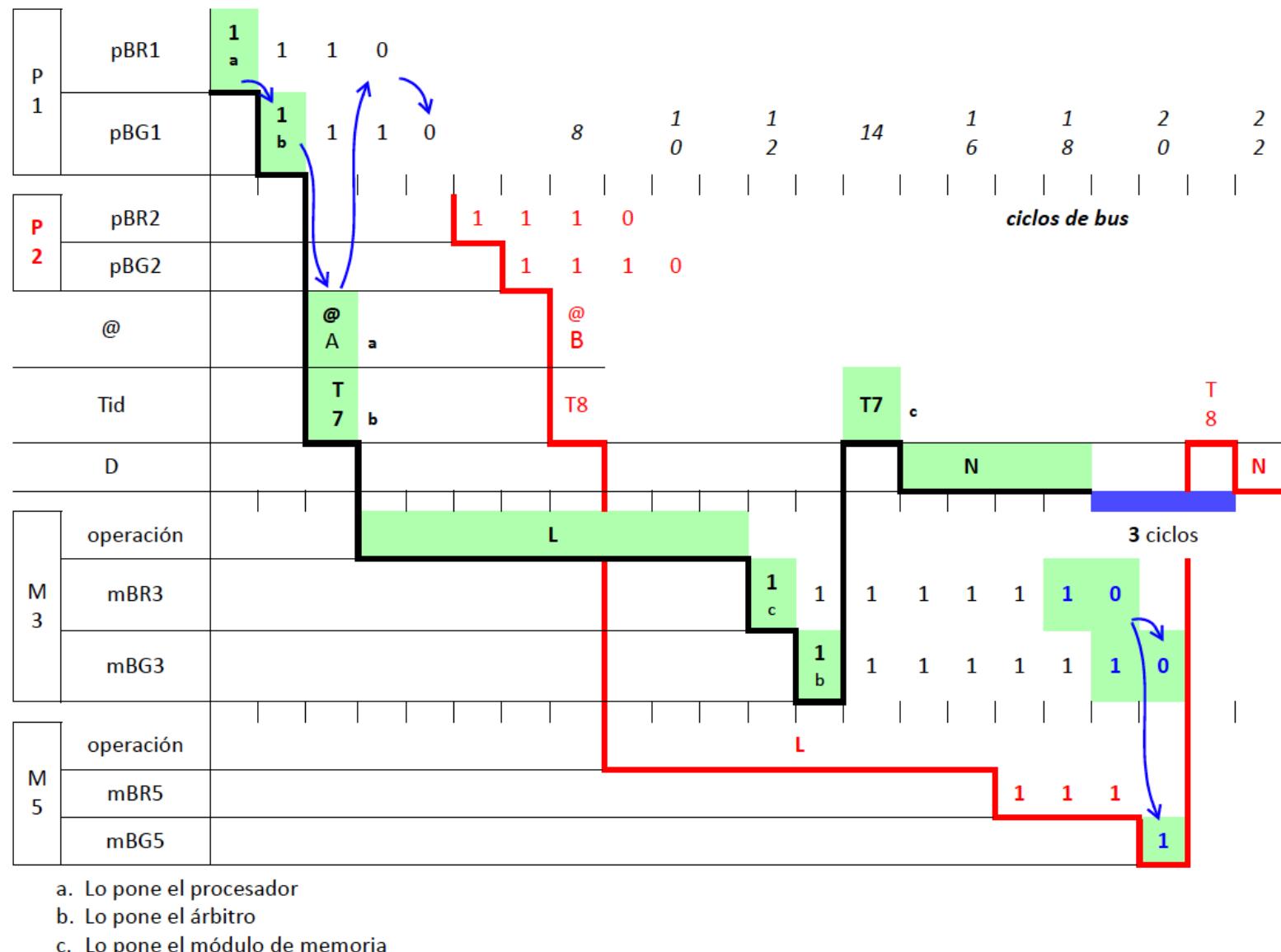
$$BW_{cp} = \frac{Bsize}{N + 3} \cdot f_{bus} = \frac{1}{1 + 3/N} \cdot BW_{crudo}$$

$\min LAT_{cc} = 3 + L + 3 + N$

EJERCICIO

Bsize=128B, fbus=1GHz, w=16B, L(tiempo)=30ns

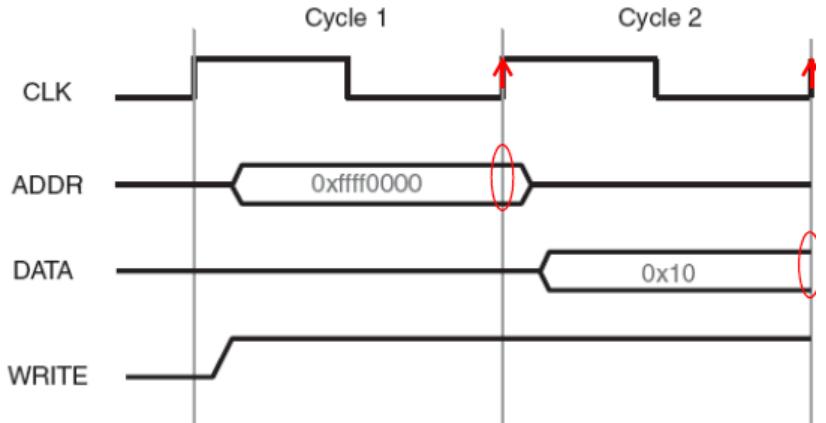
- Calcular BWcc y minLATcc
- Calcular BWcp y minLATcp



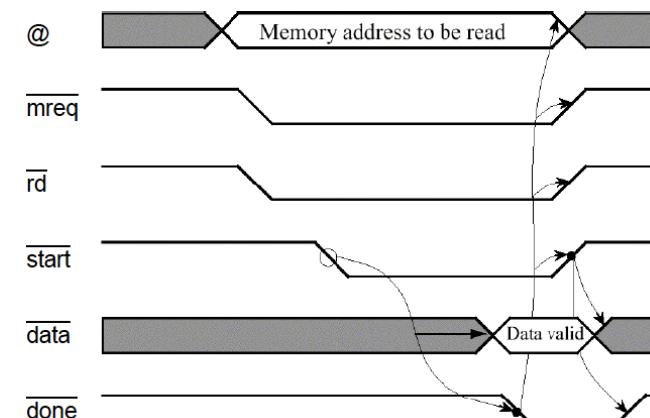
- a. Lo pone el procesador
b. Lo pone el árbitro
c. Lo pone el módulo de memoria

RECAPITULACIÓN Y MAS TÉRMINOS

- Caminos separados o multiplexados
 - ↓ separados (no multiplexados): direcciones y datos por separado
 - ↓ multiplexados: direcciones y datos en los mismo cables, en momentos diferentes
 - ↑ menos coste y prestaciones
 - ↑ no es usual en buses procesador-memoria para multiprocesadores
- Bus síncrono o asíncrono
 - ↓ **síncrono**: los ejemplos vistos
 - ↑ hay un reloj único, visible a todos los componentes están conectados al bus;
 - ↑ es fácil organizar estados y transiciones
 - ↑ el tiempo mínimo de una fase de mensajes es un ciclo de bus



- Asíncrono: no hay reloj;
 - Es fácil incluir dispositivos que responden en menos de un ciclo de bus;
 - Hay que establecer protocolos de *handshake* entre productores y consumidores;
 - Diseño lógico más difícil



Ejemplo transacción lectura trás haber ganado la propiedad del bus (no se muestra) Se coloca la dirección, después la petición de memoria (mreq) y el tipo(rd). Finalmente se señala el inicio de operación (start). Una vez los datos ya son válidos en el camino de datos, se señala desde la memoria (done)

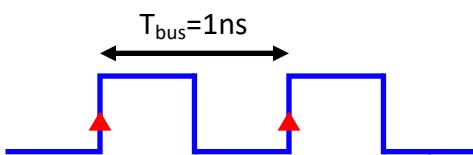
RECAPITULACIÓN Y MÁS TERMINOS

- Single-pumped, dual-pumped, quad-pumped

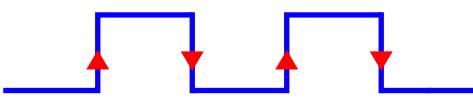
Partiendo de un bus existente, puede aumentarse su ancho de banda con poco esfuerzo de diseño, sin tocar la anchura del bus de datos (**w**) o la frecuencia de bus (f_{bus}).

¿Cómo?

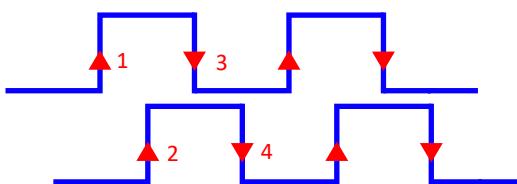
Aprovechando los modos de transferencia Double Data Rate de los chips de DRAM y posibilitando *varias* transferencia de datos por ciclo de bus



$F_{bus} = 1/T_{bus} = 1\text{GHz}$
Single-pumped: 1 transferencia/ciclo
 $BW_{crudo} = w \cdot f_{bus}$

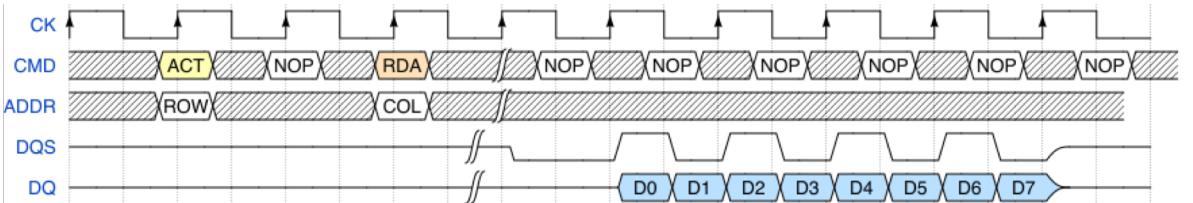


Double-pumped: 2 transferencia/ciclo
 $BW_{crudo} = 2 \cdot w \cdot f_{bus}$



Otra señal de rejoe, retrasada $\frac{1}{4}$ de ciclo ($\pi/2$)
Double-pumped: 4 transferencia/ciclo
 $BW_{crudo} = 4 \cdot w \cdot f_{bus}$

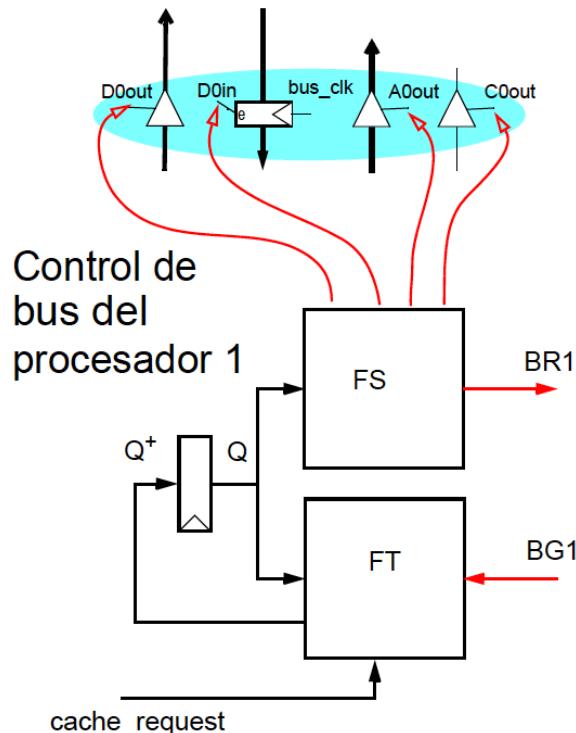
TIMING DIAGRAM OF A DDR READ OPERATION WITH BURST LENGTH OF 8 (BL8)



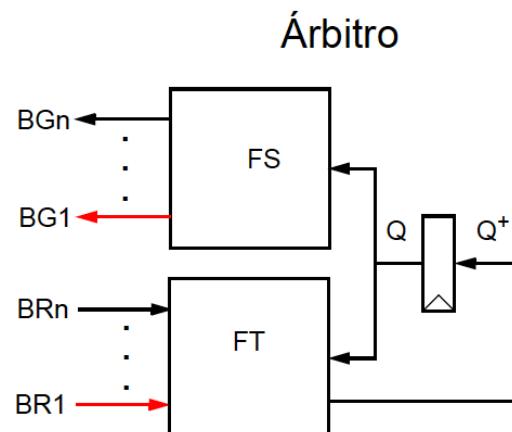
- The first step is an ACT command. The value on the address bus at this time indicates the row address.
- In the second step a TDA (Read with Auto-Precharge) is issued. The value on the address bus during at this time is the column address
- The RDA command tells the DRAM to automatically PRECHARGEs the bank after the read is complete.
- Taken from: <https://www.systemverilog.io/ddr4-basics>

PC4-25600 module with DDR4-3200 chips: JEDEC Standard DDR4 Module (284-pin DIMM)			
CLK	Bus speed	Bus width	BW
1.6GHz	3.2G trans/s	8 Bytes	25.6 GB/s

DETALLE DEL PROTOCOLO BR/BG



Modelo sencillo con autómatas de Moore



- El símbolo “-” indica repetición de la casilla anterior
 - Este protocolo pierde dos ciclos de bus en cada concesión (en amarillo)

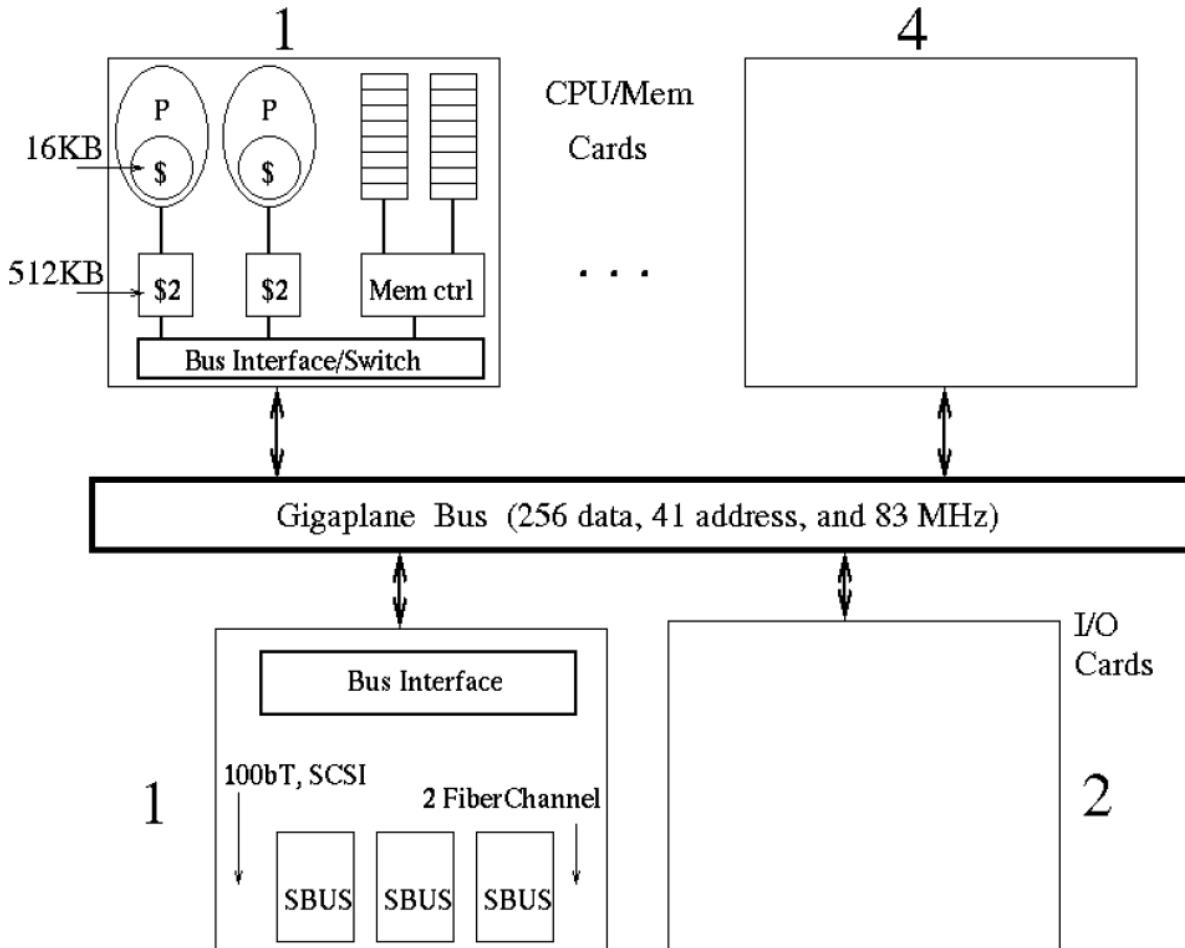
P a	BRa	0	1	-	-	-	-	-	-	-	-	-	-	-	1	0				
	BGa			1	-	-	-	-	-	-	-	-	-	-	-	1	0			
BUS					a	-	-	-	-	-	-	-	-	-	a		b	-	-	-
P b	BRb						0	1	-	-	-	-	-	-	1	-	-	-	-	1
	BGb														0	1	-	-	-	-

Protocolo sencillo:
<BRⁿ, BG, uso bus>
<Br^{último}=último ciclo uso buss>

- BR=0 → no quiero bus
- BR=1 → quiero bus
- BG=0 → espera
- BG=1 → toma

SUN MICROSYSTEMS ENTERPRISE 6000/6500 (1996)

The following diagram shows the logical organization of the Enterprise multiprocessor



The boxes labeled with "\$" and "\$2" stand for primary and secondary cache, respectively. This design uses a hierarchical structure, where each card is either a complete dual processor with memory or a complete I/O system.

- The full system configuration provides 16 bus slots that can be occupied by either processor or I/O boards, but must be at least one of each.
- Each processing board contains 2 CPU modules and 2 (512-bit wide) banks of memory of up to 1GB each, which are uniformly accessible to all boards.
- The I/O board provides connectors for multiple independent peripheral buses and appears like any another cache controller on the system bus.
- Each processor is an UltraSparc with 16KB level 1 cache and 512KB level2 cache.
- The split-transaction bus allows up to 112 transactions at a time.
- Max number of processors supported is 30 and a peak performance at 9 GFLOPs.
- Both memory and bandwidth scales up with the number of processors.
- The Gigaplane system bus provides a peak bandwidth of 2.67 GB/s.
- The system has a nonmultiplexed, split-transaction bus (Sun Gigaplane) with 256-bit data lines and 41-bit physical address. Gigaplane is clocked at 83.5 MHz.
- Gigaplane can support up to 112 outstanding transactions, including up to 7 from each board.
- The bus consists of a total of 388 signals: 256 data, 32 ECC (error correcting code), 43 address (with parity), 7 ID tag, 18 for arbitration, and a number of configuration signals.

SUN MICROSYSTEMS ENTERPRISE 6000/6500 (1996)

MEMORY SYSTEM

- Since the system is designed to support up to 30 processors, and since each processing board contains up to 2 GB, the overall system can support up to 30 GB of up to 16-way interleaved memory.
- In the above diagram it is shown that every processor is associated with first level and second level cache. Arrays of 16 KB or less fit entirely in the first-level.
- Second-level cache accesses have an access time of about 40 ns, and the transfer between the two levels is about 16 bytes. Overall the access time is about 300 ns, 130 ns out of this goes to the bus protocol which has a transfer rate of 83.5 MHz.

OPERATING SYSTEM SOFTWARE AND ENVIRONMENT

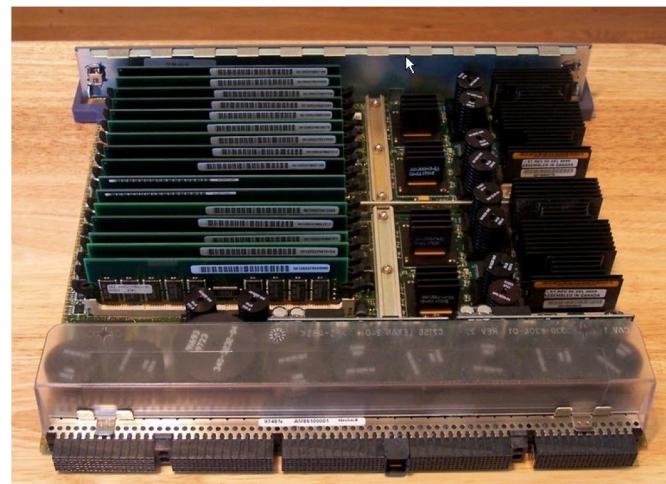
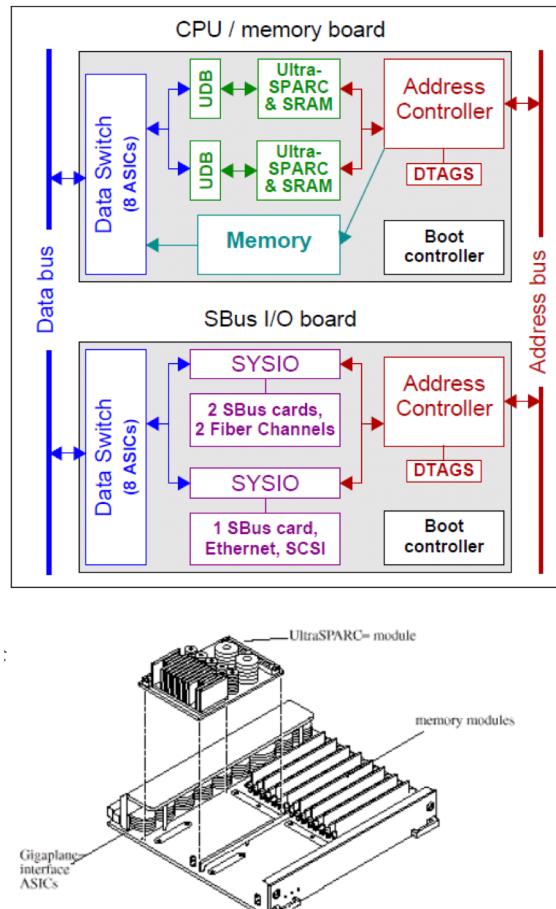
- Operating System: Solaris 2.5.1/2.6 operating environment
- Languages: The following languages are supported: C, C++, Pascal, Fortran, and Java.
- Windowing system: Is OpenWindows Version 3 optional.
- System monitoring: Hostview.
- System and network: is done through Solstice Site Manager and Solstice Domain Manager, also through Network management software for Solaris, Solstice DiskSuite, and SPARCstorage Volume Manager.

NETWORKABILITY | I/O SYSTEM | INTEGRABILITY | RELIABILITY | SCALABILITY

- The Enterprise I/O board uses the same bus interface as the processing board, but the internal bus is only half as wide and there is no memory path.
- Externally, the I/O boards only do cache-block-sized transactions, just like the processing boards.
- Internally, 2 independent 64-bit 25-MHz SBUSs are supported. One of these supports 2 dedicated FiberChannel modules providing redundant, high-bandwidth interconnect to large disk storage arrays. The other provides dedicated Ethernet and fast wide SCSI connections. In addition, 3 SBUS interface cards can be plugged into the 2 buses to support arbitrary peripherals, including a 622-MB/s ATM interface.
- The I/O bandwidth, the connectivity to peripherals, and the cost of the I/O system scales with the number of I/O cards.
- Again in the above figure each I/O card provides two independent 64-bit X 25-MHz SBUS I/O buses, so the I/O bandwidth scales with the number of I/O cards.
- Total of disk storage is in tens of terabytes.
- The system scales up to 30 processors of 167 MHz, 250 MHz, 336MHz.

SUN MICROSYSTEMS ENTERPRISE 6000/6500 (1996)

FOTOS SUN E60001



Placa biprocesador+memoria
La conexión al bus en la parte frontal



Palca de E/S
La conexión al bus en la parte trasera



1. Autor: Elf Kellogg. Álbum público "Sun Enterprise 6000" en Picasa

EJEMPLOS DE BUSES COMERCIALES

Bus <i>Sistema</i> max. nº cpus	año	modo split transaction	@ y datos	long cm	@, Tid bits	f MHz	w Bytes	BW crud GB/s	BW max GB/s	Coherencia (Bsize)
SGI Challenge Power Path-2 36 x MIPS R10K @200 MHz	93	8 trans. en vuelo	separados	30,5	40 ?	47,5	32	1,52	1,2 79%	si
DEC Alpha Server 8200, 8400 12 x Alpha 21164@300Mhz	95	en orden, 8 trans. en vuelo	separados	30,5	40 ?	75	32	2,4	1,6	si
Sun Gigaplane Ultra Enterprise E3000- E6000 30 x UltraSPARC-II	96	desorden, 112 trans. en vuelo	separados	40,6	41 7	83,5	32 + 4 ECC	2,67	2,67 100%	OSI-Dtags MOESI-cpu (64B)
HP Runway Bus 4 x PA7200 - PA8x00 a partir del PA8500: Runway+	96	desorden, 64 trans. en vuelo	multiplex.		40 ?	120	8	0,96	0,77 80%	MESI
la sincronización de datos se hace <i>dual-pumped</i>						125		2		
IBM CoreConnect PLB6 ^a	09	split trans, desorden, trans. en vuelo: 32rd+16wr command &data pipelining ^b	separados	chip	64 5 rd 4 wr	400	16 +16 ^c	6,4	6,4	MESI + 3 cache intervent. states ^d + I/O ^e
Hardware-enforced cache coherency. Coherent and non-coherent I/O devices. I/O masters attach to a master interface, while coherent processors attach to both a master and a snoop interface. Coherent slaves, such as memory controllers, connect to a single slave segment, and I/O slaves connect to a slave interface on one of seven other segments. Each non-coherent I/O master requires on the order of 450 I/O signals. Each coherent master uses about an additional 250 I/O signals, for snooping and intervention data.										

- a. Embedded Bus Architecture Report. Presented by the Bus Architecture TSC. Version 1.0, 11 April 2008. Power.org™. Este bus es el de gama mas alta para construir SoC con los procesadores IBM Power Architecture empotrados 4xx.
- b. Command pipelining occurs when a master makes a second request without having received a response for the first request. Usually, masters must receive an acknowledge for an initial request before making a second request. PLB6 masters can make the subsequent request before receiving an acknowledge for the first. The ability to support command pipelining is critical to maintaining a high level of performance in implementations with hardware-enforced cache coherency. Data pipelining is also supported to sustain high bandwidth despite having a moderate to high level of latency to memory.
- c. Separate Data Buses: PLB6 doubles the peak bandwidth at a given frequency by using separate buses for read and write data. These buses are used in conjunction with pipelining command to data phases to increase performance.
- d. Modified/Exclusive/Shared/Invalid, Tagged, Shared Last, Modified Unsolicited (MESI + T + SL + MU)
- e. I/O Coherency: Single processor implementations can also benefit from supporting cache coherency. I/O masters making read requests to coherent memory can 'hit' the cache, and the returning read data will be provided with low latency, since it is driven from the cache and not main memory.

TEMAS INTERESANTE PARA PROFUNDIZAR

- Formas de arbitra: filosofías, diseño y rendimiento
- Buses estándar para unir Intellectual Property Block (IP blocks) en SOC como el IBM CoreConnect de la tabla.

Otros ejemplos

- AMBA 3-AXI (ARM)
- Sonics Smart Interconnect (Sonics)
- Open Core Protocol (OCP)
- STBus (STMicroelectronics)

- Extensión del protocolo de acceso a memoria
 - lectura, escritura, sincronización – para conseguir coherencia entre las cache privadas de los procesadores → lo veremos

CLASIFICACIÓN DE COMPUTADORES PARALELOS

- Mike Flynn, 1966¹
- SISD: Single instruction operates on single data element
- SIMD: Single instruction operates on multiple data elements
 - ✓ Conventional processor

CLASIFICACIÓN DE COMPUTADORES PARALELOS