

Arquitetura de Computadores III

Parte 5

Multithreading, Multi/Many-core,
Redes-em-Chip, Máquinas paralelas

Multithreading e Multi-Core

Conceitos e Exemplos

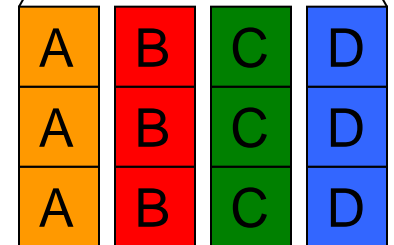
Suporte a múltiplas threads

Threads: instrução ou bloco



Vertical

Horizontal

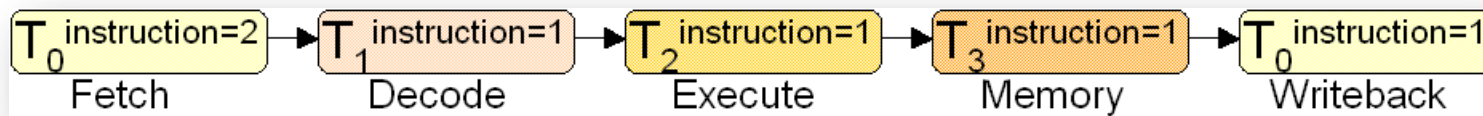


IMT / BMT

SMT

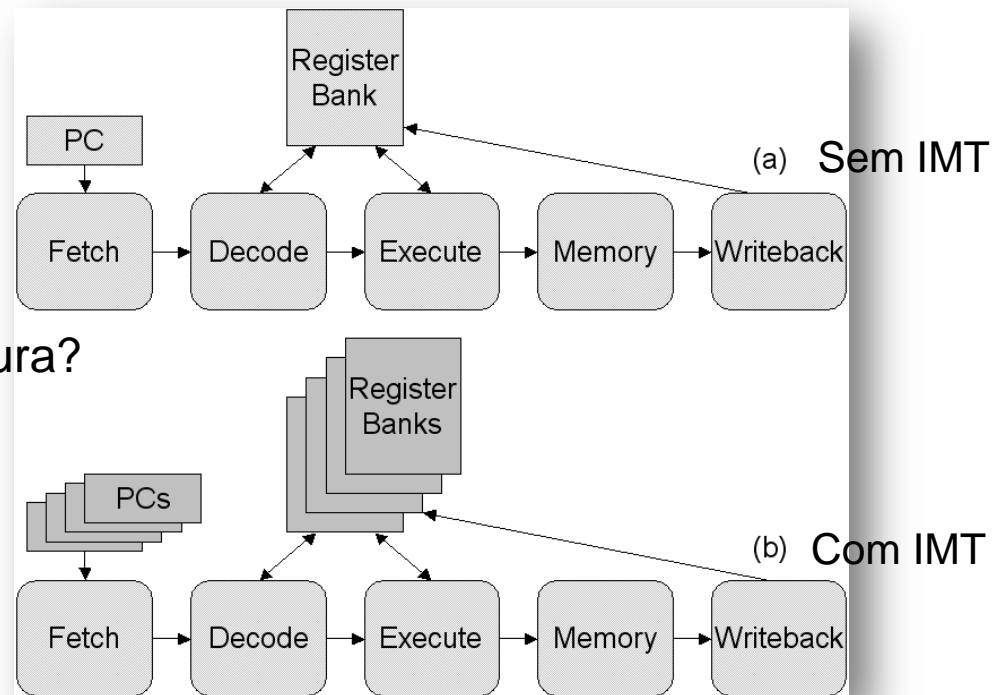
Suporte a múltiplas threads

Quantas instruções são executadas simultaneamente?

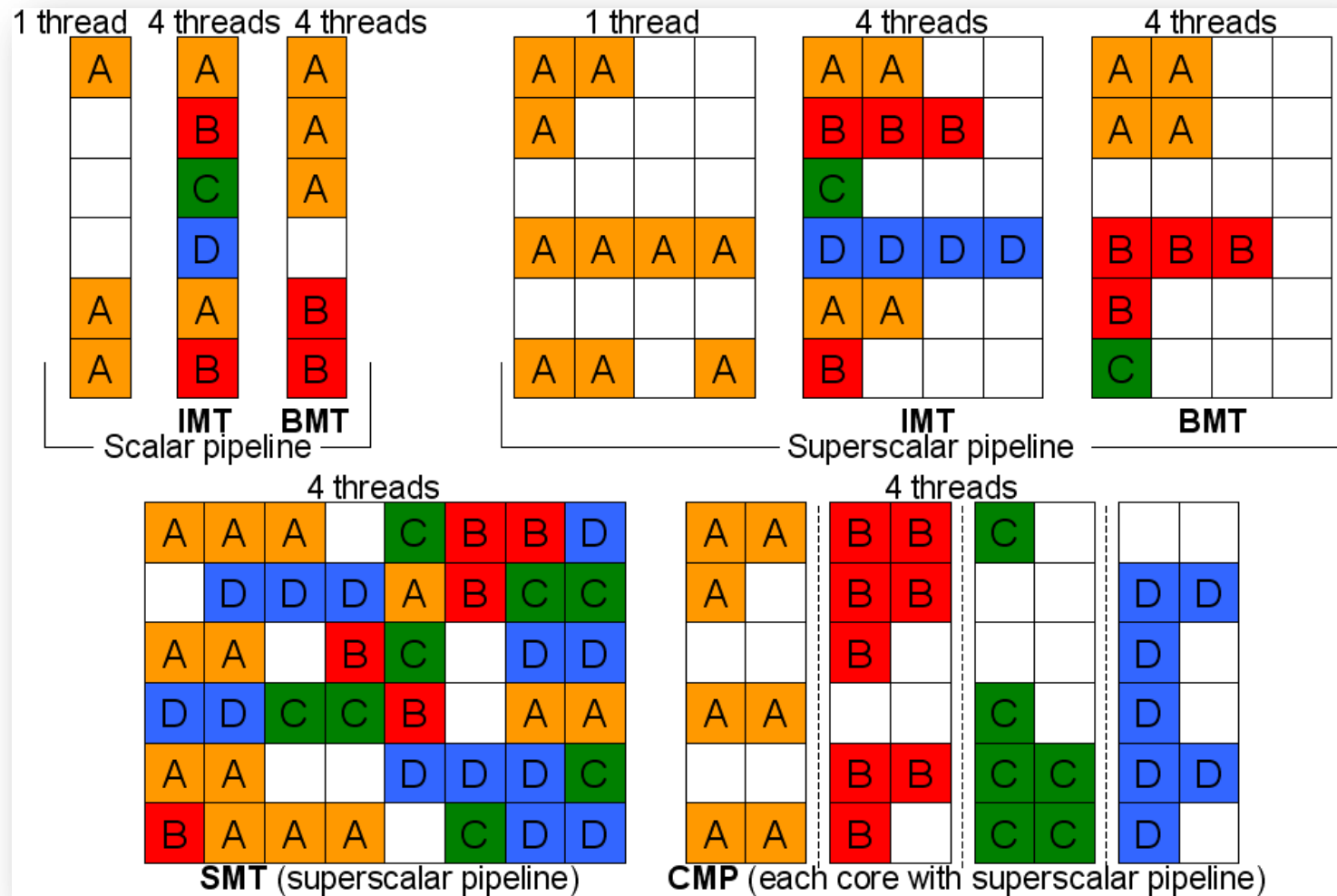


Freitas, 2009

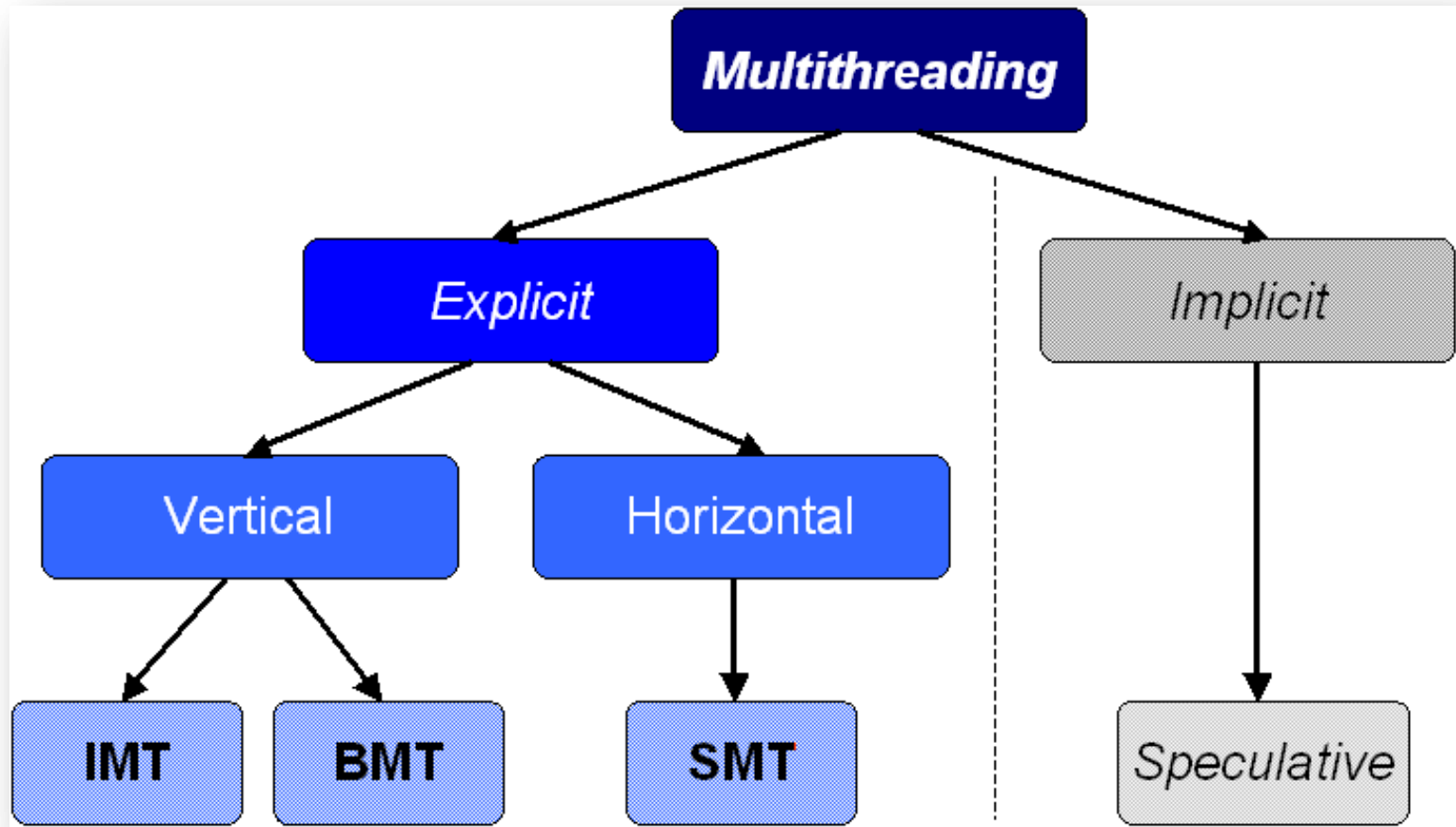
Qual o impacto na arquitetura?



Suporte a múltiplas threads



Suporte a múltiplas threads

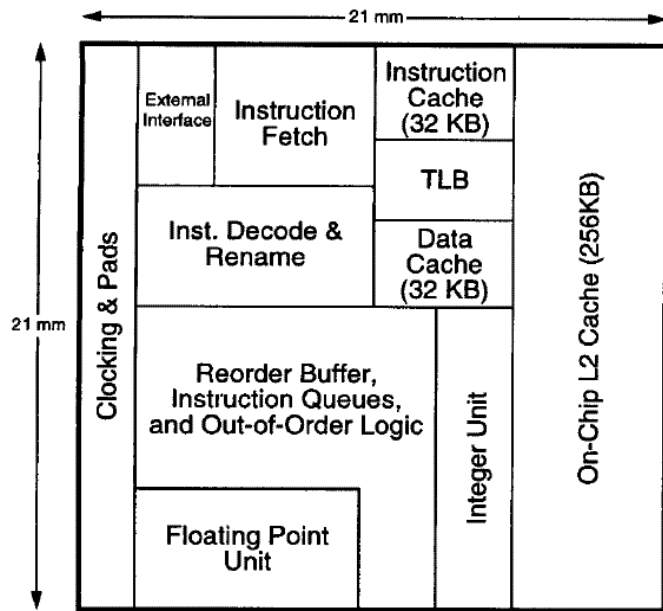


Suporte a múltiplas threads

- Benefícios:
 - CPI \Rightarrow IPC (Superescalar).
 - Vazão de instruções (Superescalar) \Rightarrow Vazão de *threads* (SMT).
 - Ilusão de mais de um núcleo de processamento.
 - Não existe o esvaziamento de *pipeline* comum no BMT.
 - Não há atraso na execução de *threads*, comum no IMT/BMT.
- Desafios / Problemas:
 - Tamanho da arquitetura.
 - Banco de registradores muito grande para guardar vários contextos.
 - Divisão de recursos e equilíbrio de desempenho.
 - Conflitos de *cache* sem degradação de desempenho.

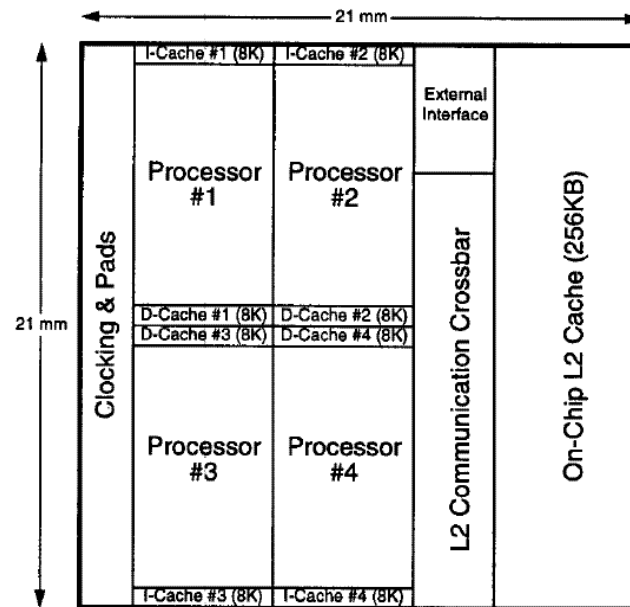
Processadores multi-core

- (a) Superescalaridade de seis vias de execução.
- (b) Chip multi-core. Cada core superescalar com duas vias de execução.



(a)

10% a 30%
ILP

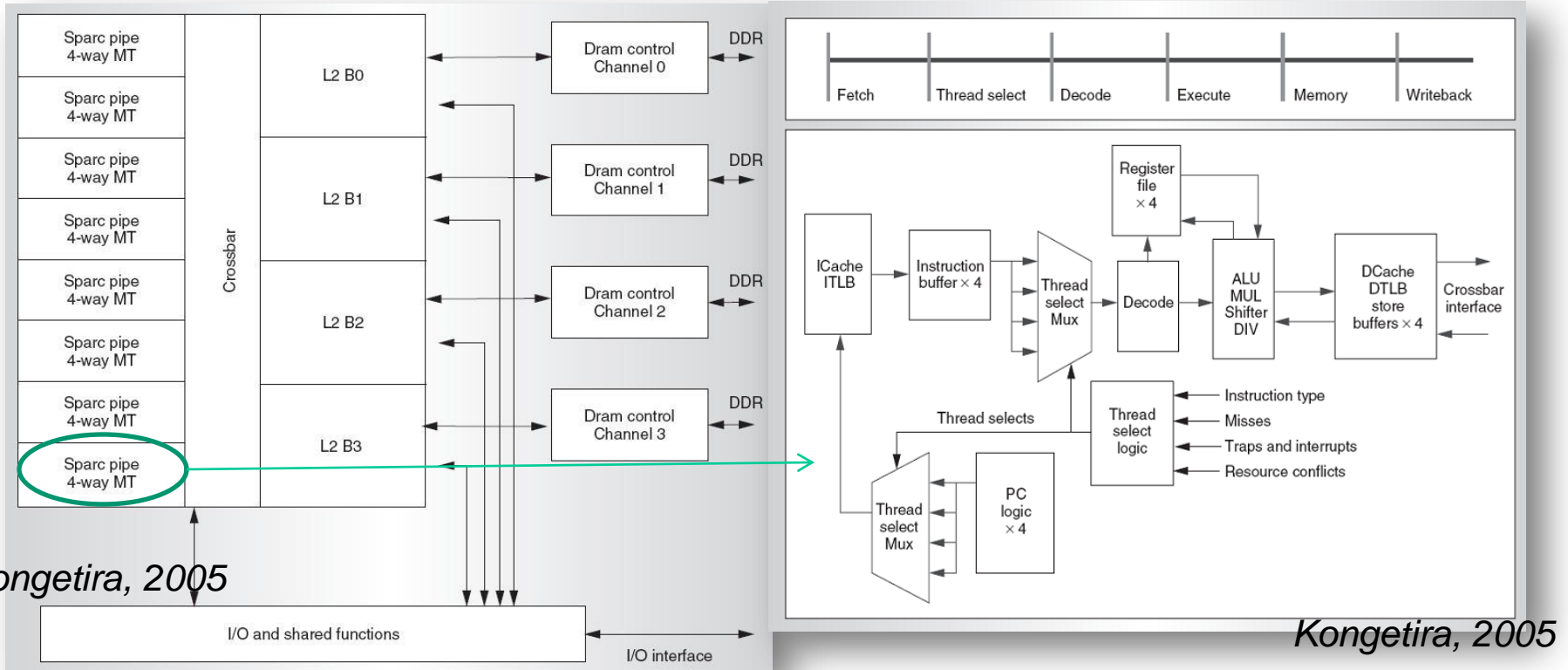


(b)

50% a 100%
TLP

Olukotun, 1996

Processadores multi-core



- Suporte a 4 *threads* IMT por núcleo (32 *threads* ativas, 8 *threads* simultâneas).
- *Crossbar switch* de 134,4 GB/s.
- 4 canais DDR 23GB/s.
- Potência < 80W.

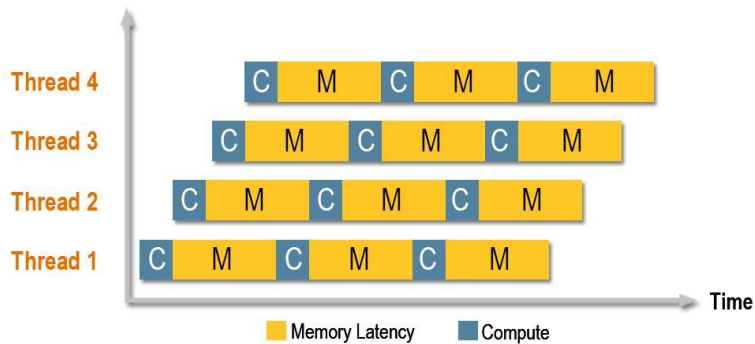
Processadores multi-core

Quantas threads simultâneas?

?

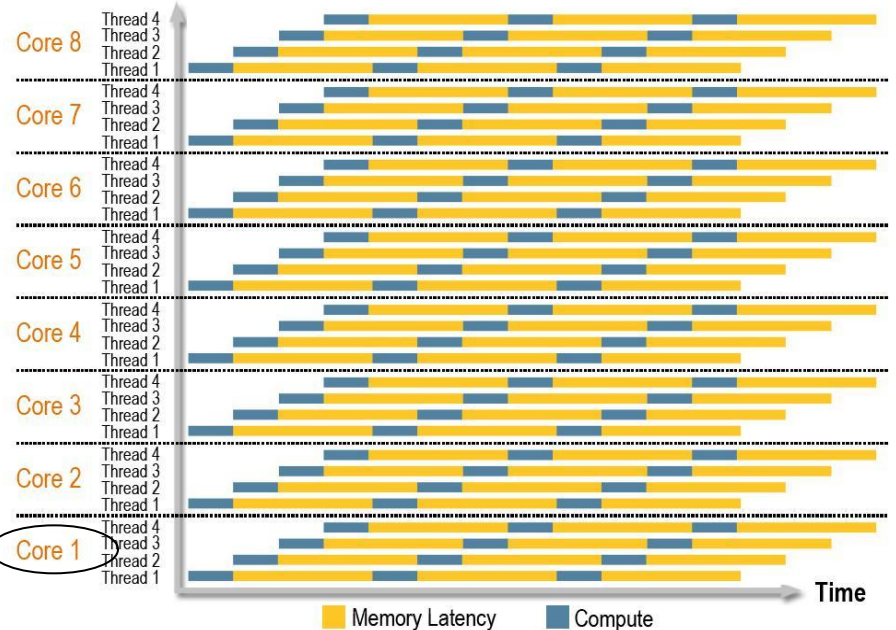
?

Chip Multithreading (CMT)



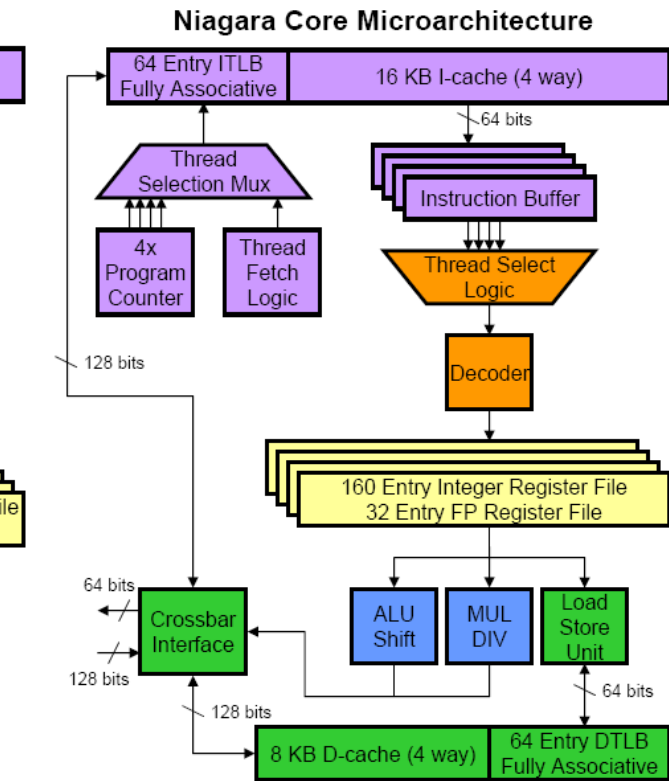
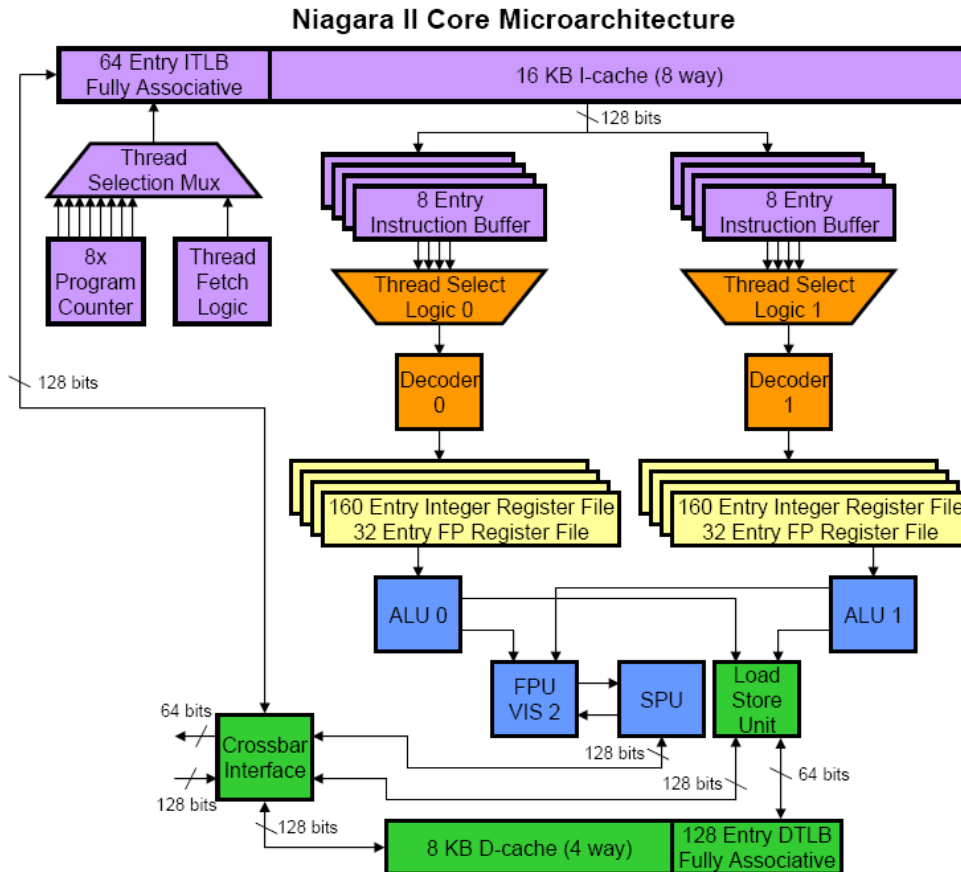
Yen, 2005

CMT – Multiple Multithreaded Cores



Yen, 2005

Processadores multi-core

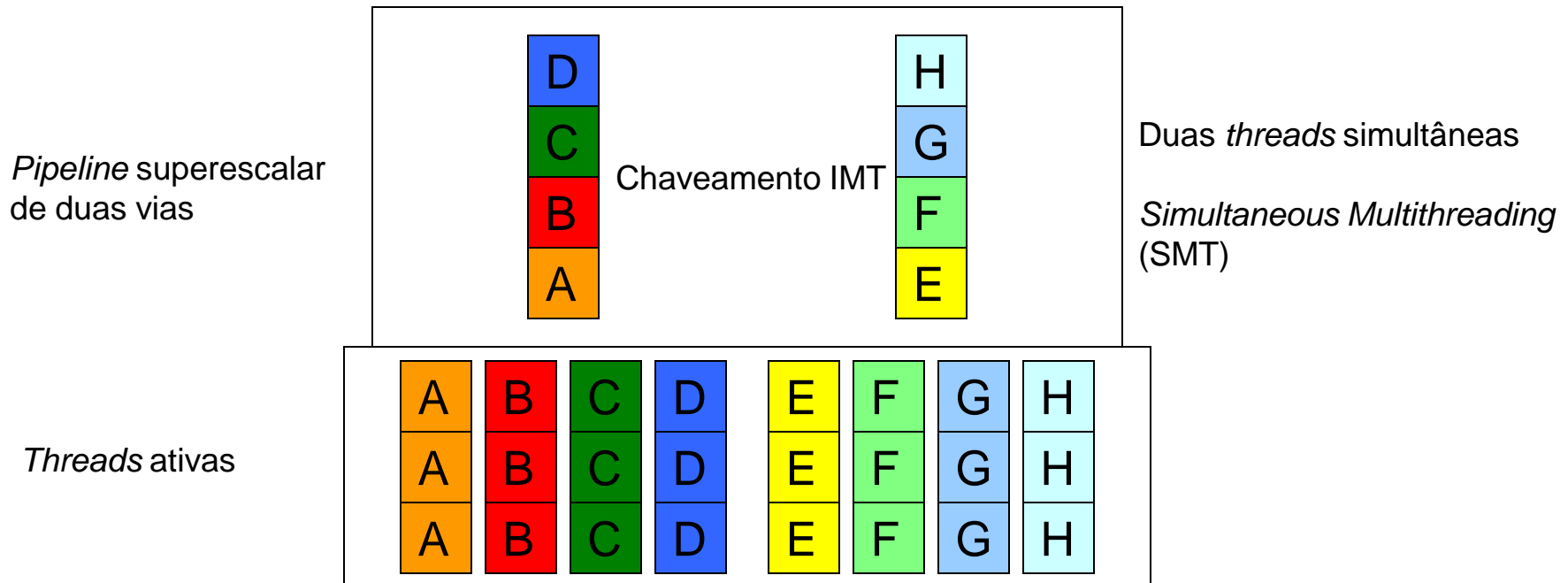


Kanter, 2006

- Suporte a 8 *threads* por núcleo (64 *threads* ativas, 16 *threads* simultâneas).
- Instruções executadas em ordem, previsão de desvio estático ou pela última decisão tomada.
- Crossbar switch de 268,8 GB/s.

Processadores multi-core

- SMT combinado com IMT em *chip multi-core*.
 - Superescalaridade associada a SMT com entrelaçamento de instruções.
 - Aumenta desempenho para cargas de trabalho de propósitos gerais.
 - Reduz tamanho e complexidade da arquitetura superescalar do processador.



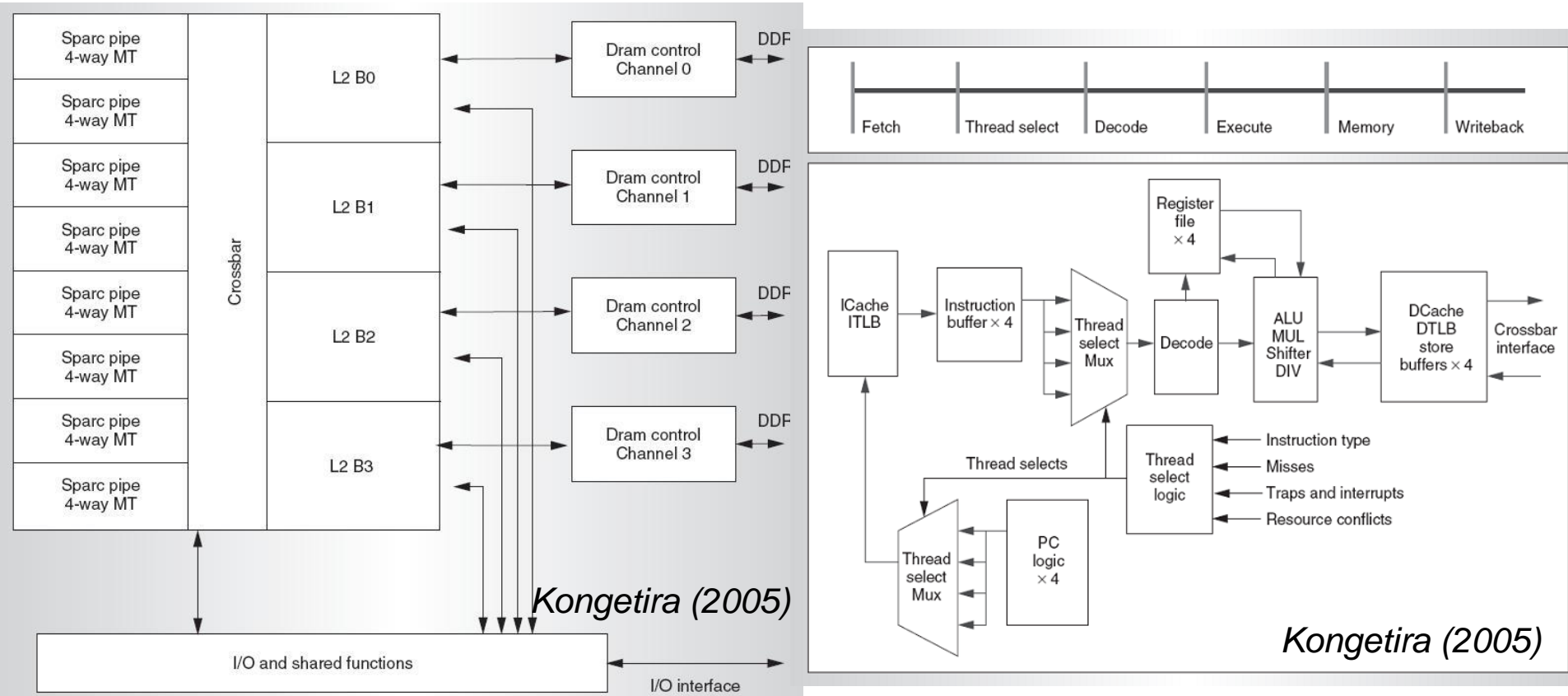
Qual a origem do processador multi-core?

- Em uma fábrica de processadores tão tão distante...
 - O diretor para o arquiteto: Precisamos de mais desempenho!
 - O arquiteto para o diretor: Não é possível aumentar o paralelismo de instruções!
 - O diretor para o engenheiro: Precisamos de mais desempenho!
 - O engenheiro para o diretor: Não é possível aumentar a frequência, o chip vai queimar!

Qual a origem do processador multi-core?

- Em uma fábrica de processadores tão tão distante...
 - Alguém diz: E a Lei de Moore!?
 - A Lei de Moore está relacionada à capacidade de integração.
 - Não está relacionada ao aumento de frequência.
 - Está relacionada a quantidade de transistores em um mesmo espaço.
 - Se diminuirmos o tamanho dos transistores?
 - 180 nm, 130 nm, 90 nm, 65 nm, 45 nm, 32 nm, 22 nm....
 - Vamos aumentar a quantidade de processadores dentro do chip de processador!
 - Vamos chamá-los de núcleos!
 - Portanto, não adianta apenas aumentar paralelismo de instruções, nem frequência de operação!
 - Precisamos aumentar a quantidade de núcleos!

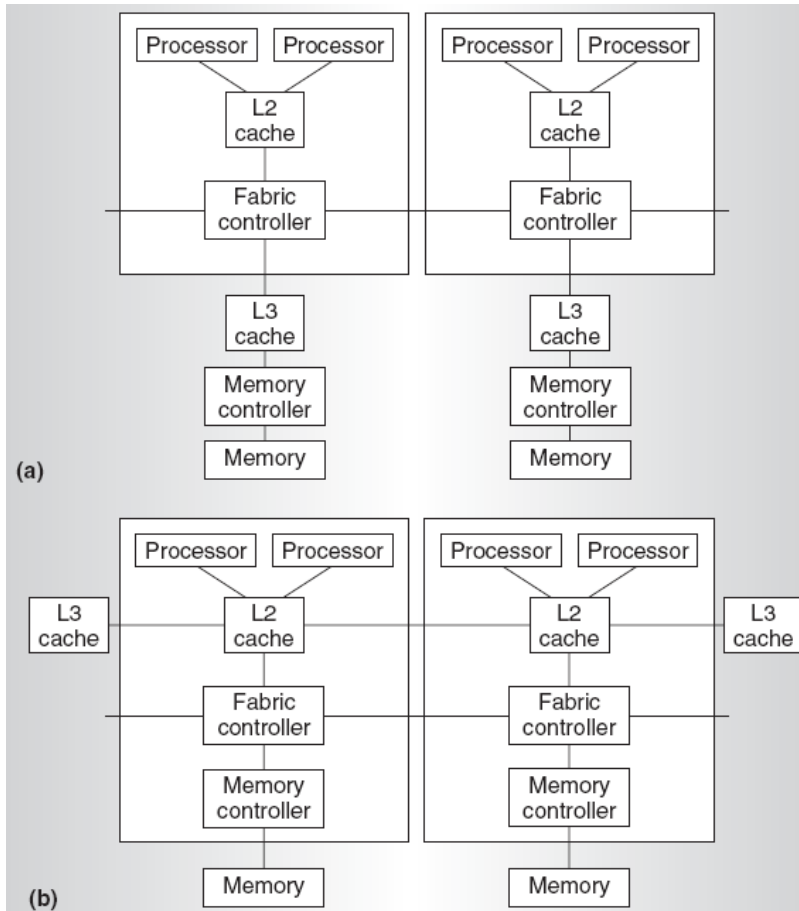
UltraSparc-T1 (Niagara 1)



- Suporte a 4 *threads* IMT por núcleo (32 *threads* ativas, 8 *threads* simultâneas).
- *Crossbar switch* de 134,4 GB/s.
- 4 canais DDR 23GB/s.
- Potência < 80W.

Power 4 e Power 5

Power 4

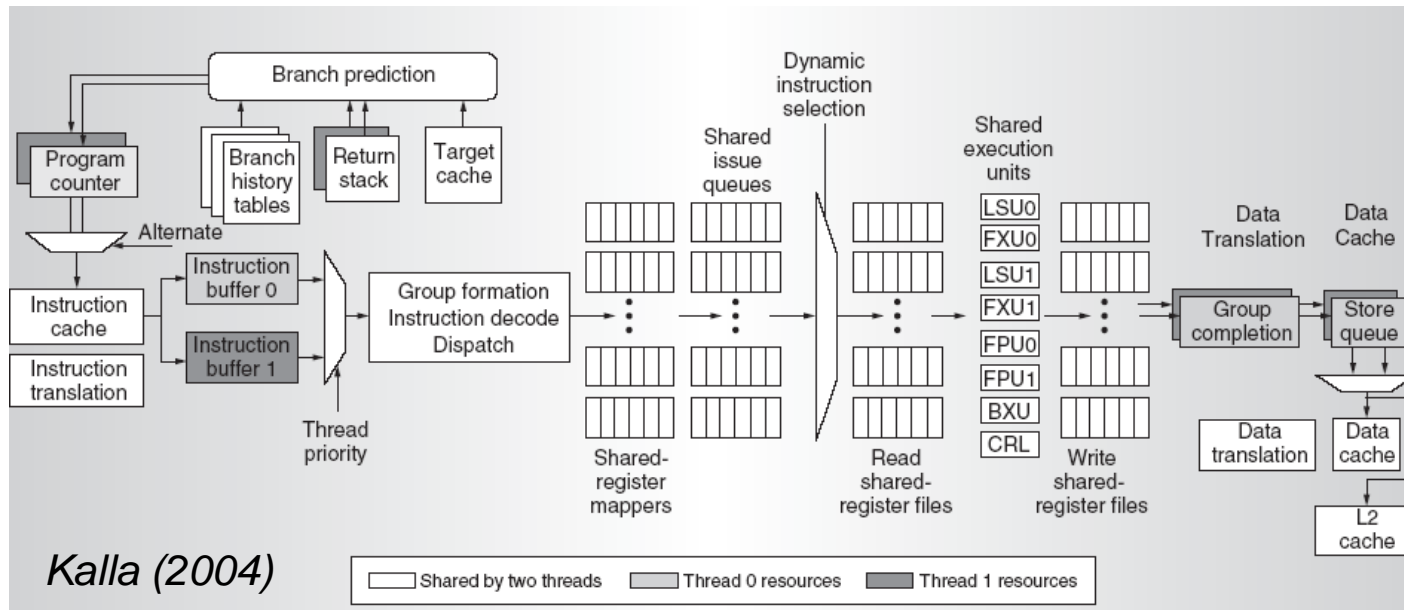


Power 5

- Os processadores Power 4 e Power5 possuem códigos binários e estrutura compatíveis.
- Diferença no **acesso a cache L3**.
- A grande diferença: o **Power 5** suporta a duas *threads* simultâneas (**SMT**) por núcleo.
 - O **Power5** possui dois núcleos físicos, mas **quatro núcleos lógicos de processamento**.

Kalla (2004)

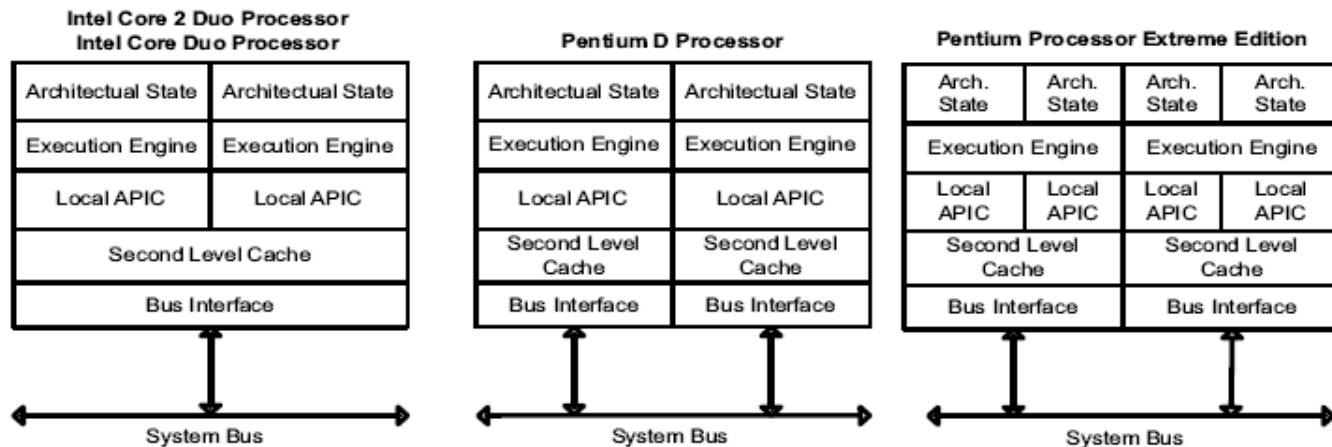
Power 5



- A informação mais interessante no projeto do **Power5** é justamente o fato do **desempenho não melhorar com o suporte SMT por núcleo**. Os principais motivos são:
 - O **número limitado de unidades de execução** que são compartilhados entre as duas *threads*.
 - O alto consumo da **largura de banda de memória** pelas duas *threads*.
- Esta é a principal razão para que o Power5 também suporte apenas uma *thread* por núcleo.

Intel *Dual Core*

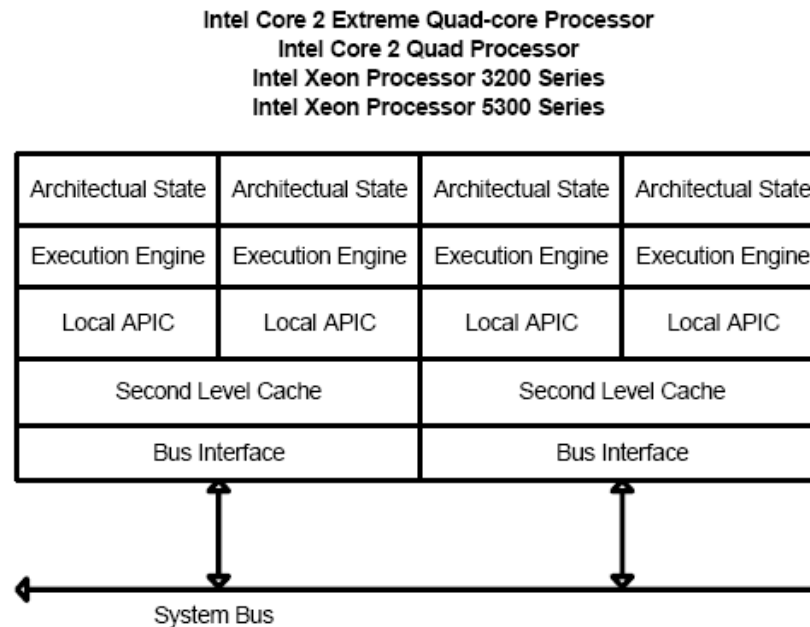
- Os processadores **Pentium D** e **Core Duo**, são compostos por dois núcleos, mas sem suporte a *hyperthreading*. Fisicamente e logicamente são dois núcleos internos.
- Diferenças básicas:
 - Suporte a múltiplas *threads*
 - *Cache* nível L2



Intel (2007b)

Intel *Quad Core*

- O projeto de um processador com 4 núcleos pela Intel é basicamente a *união de dois processadores dual core*.
- A *cache L2* compartilhada ganhou força.
- SMT não é utilizado.



Intel (2007b)

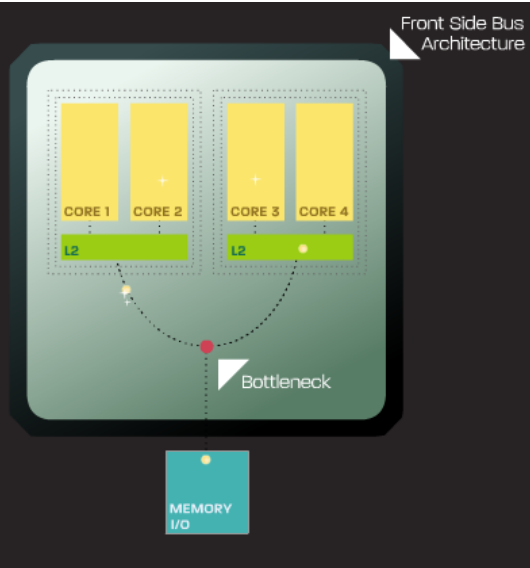
AMD *Quad Core*

Efficient By Design

Get What You Need

x86 processors with
Front Side Bus architecture:

- Share bandwidth between processors, memory, and I/O
- Can result in latency



Quad-Core

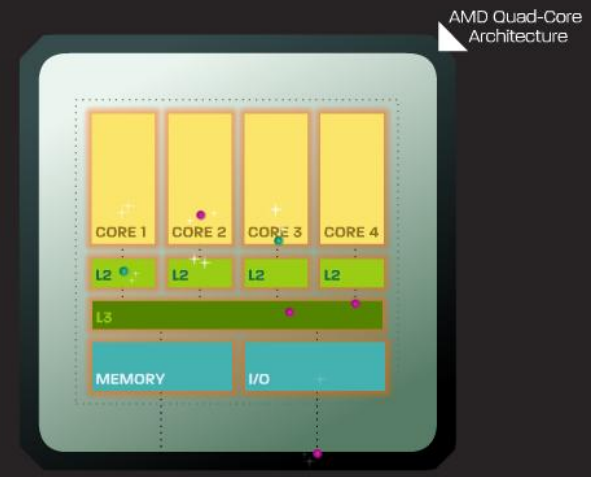
AMD (2007a)

Efficient By Design

Get What You Need

Quad-Core AMD Opteron™ processors
with Direct Connect Architecture:

- Connect the CPU to memory, I/O, and other CPUs
- Offer improved system efficiency



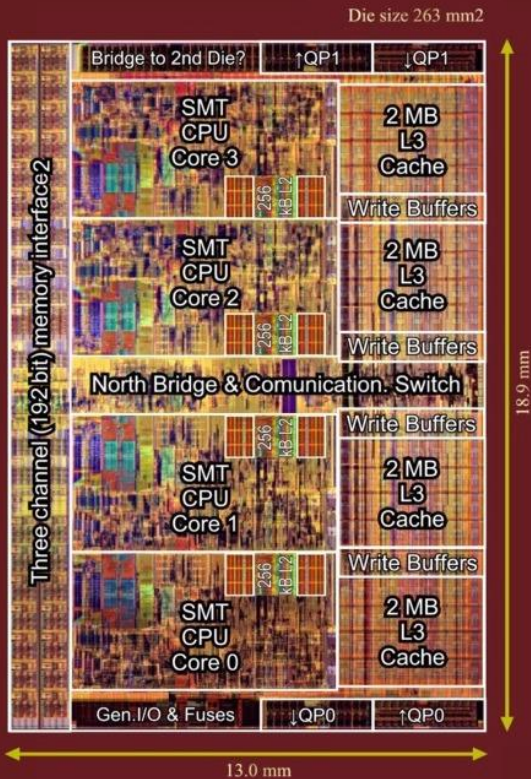
AMD (2007a)



Quad-Core

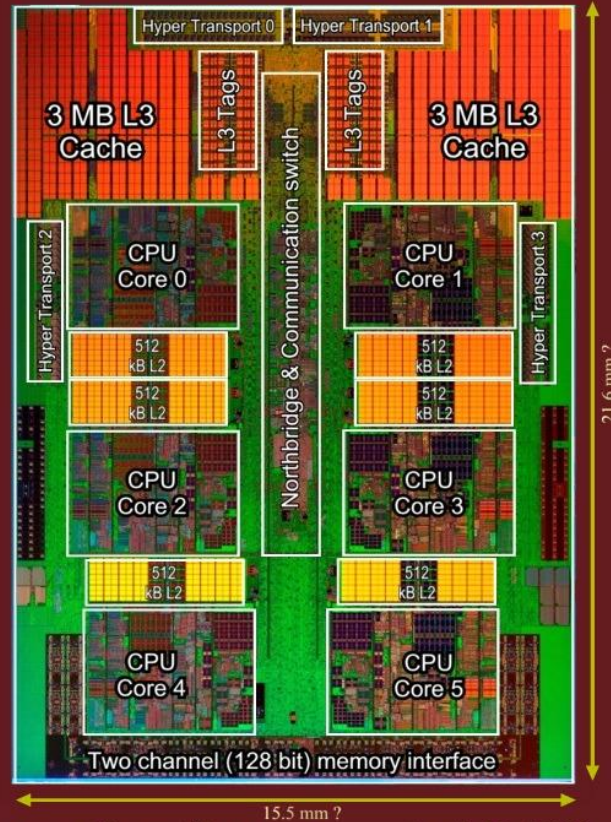
Nehalem, Istanbul, Shanghai

Intel four core Nehalem

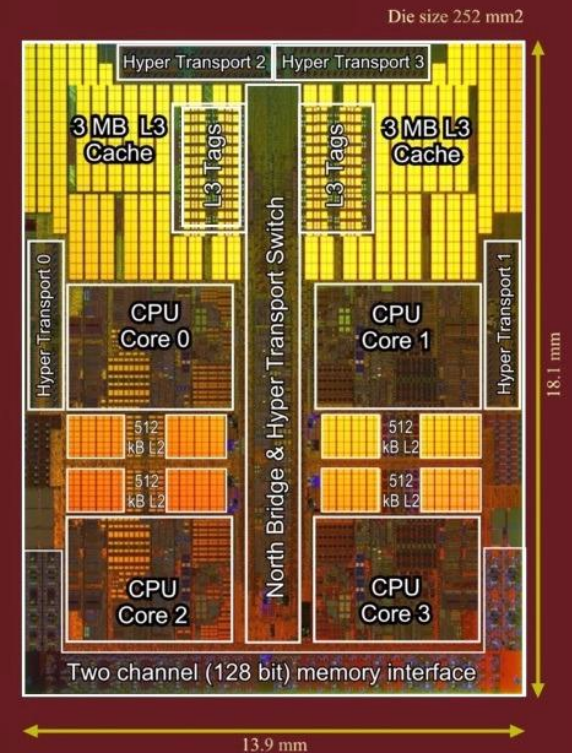


AMD six core Sao Paolo / Istanbul

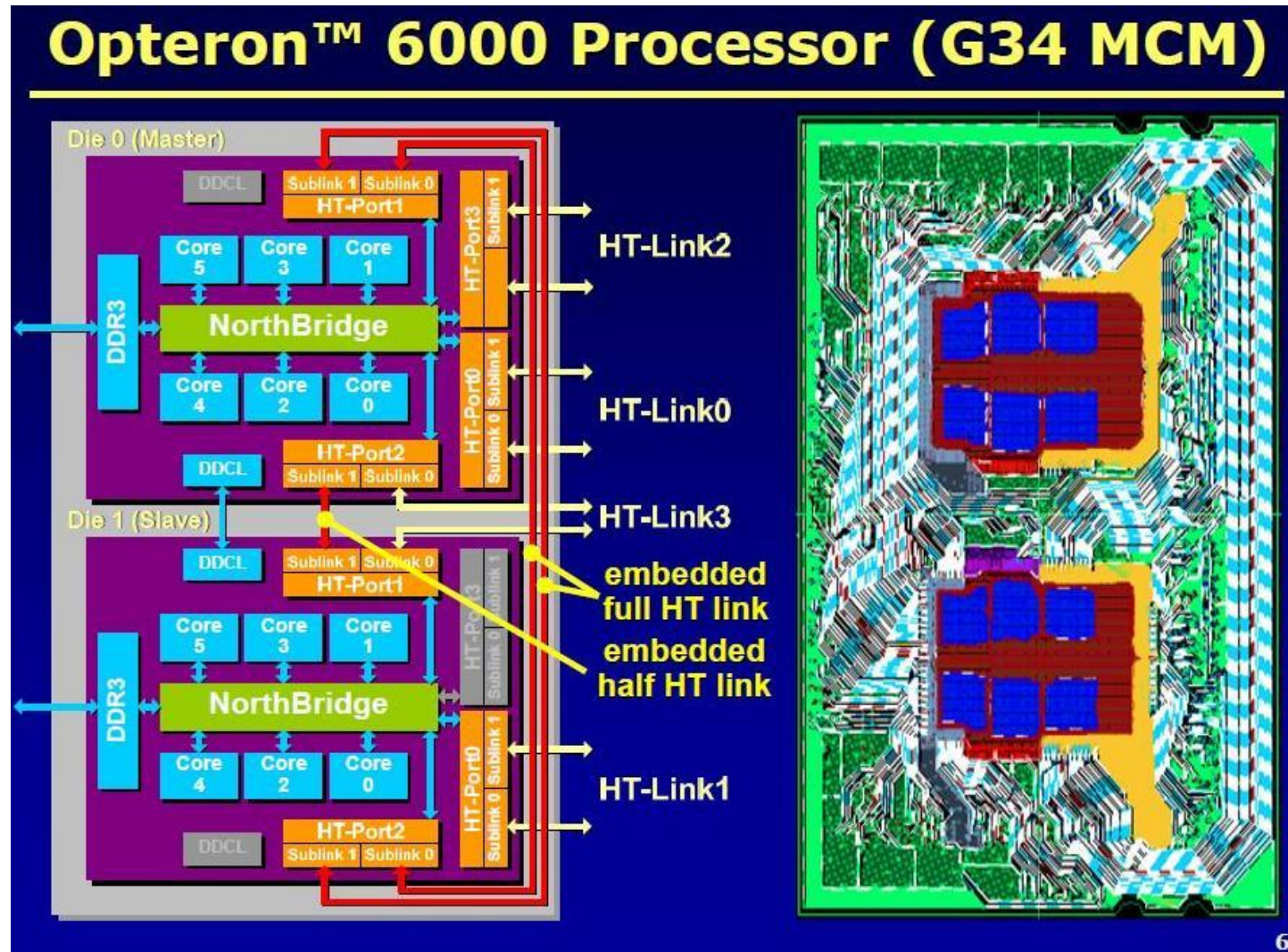
(No process scaling assumed) Die size 335 mm² ?



AMD four core Shanghai



AMD Mangy Cours (12 núcleos)



Processadores Many-Core

Redes-em-Chip

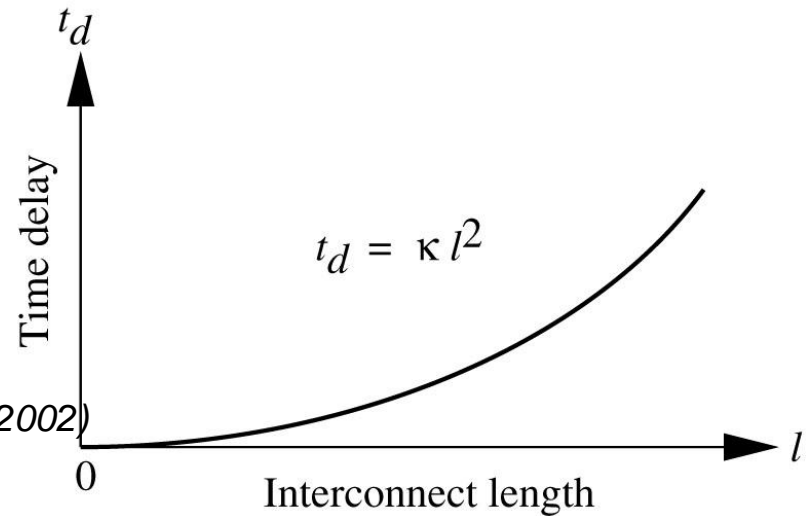
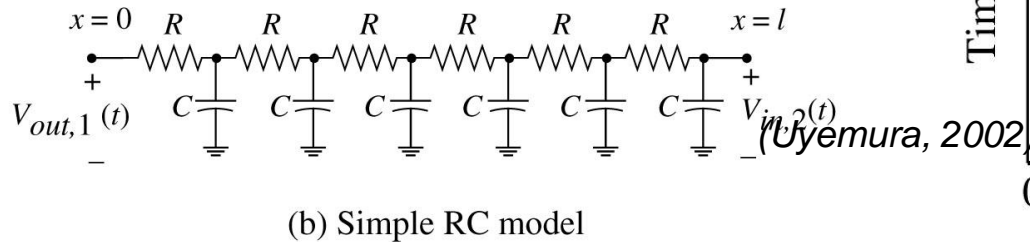
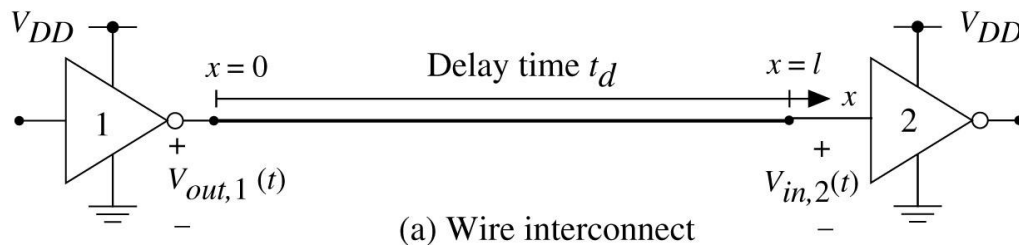
(Networks-on-Chip - NoCs)

Antes do Many-core...

- Processador multi-core:
 - 2 núcleos: barramento
 - 4 núcleos: barramento ou chave crossbar
 - 6 núcleos: barramento ou chave crossbar
 - 8 núcleos: barramento ou chave crossbar
 - 12 núcleos: barramento ou chave crossbar
 - 16 núcleos: barramento ou chave crossbar!?
- 48 núcleos: como interconectar!?

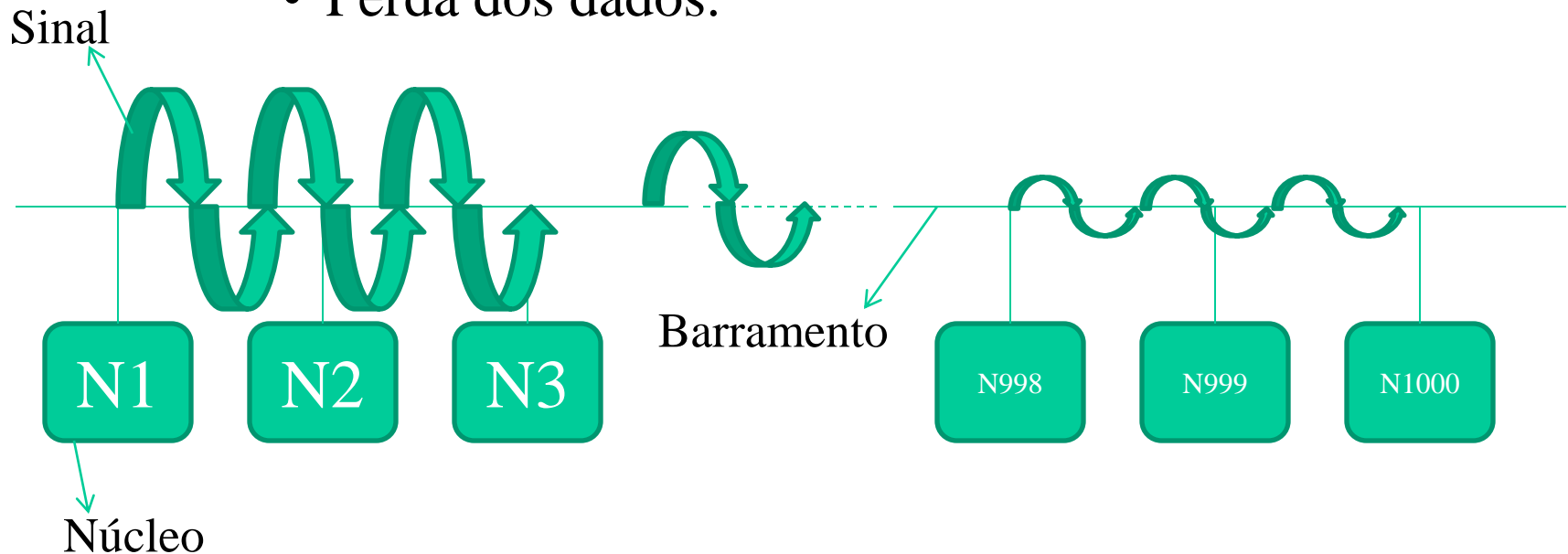
Processador Many-core

- O problema está no fio para interconectar os núcleos!



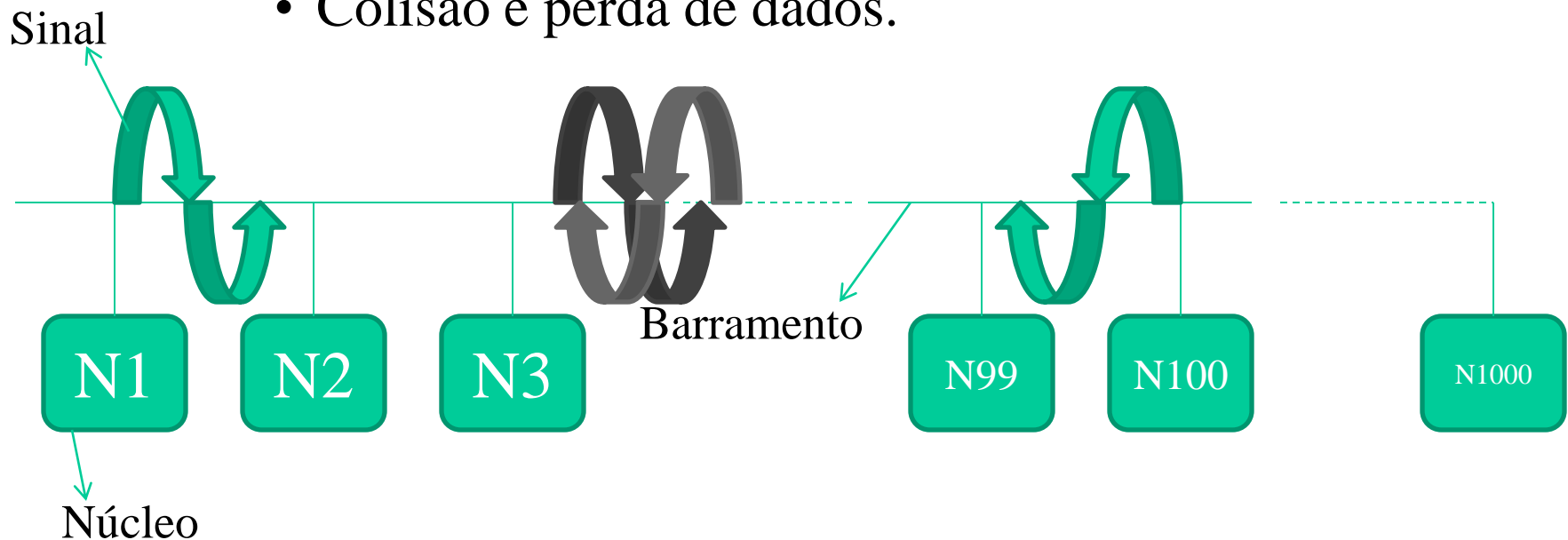
Processador Many-core

- O problema está no fio para interconectar os núcleos!
 - Problema 1: atenuação do sinal.
 - Perda dos dados.



Processador Many-core

- O problema está no fio para interconectar os núcleos!
 - Problema 2: núcleos distantes não escutam sinal.
 - Colisão e perda de dados.



Se o problema está no fio...

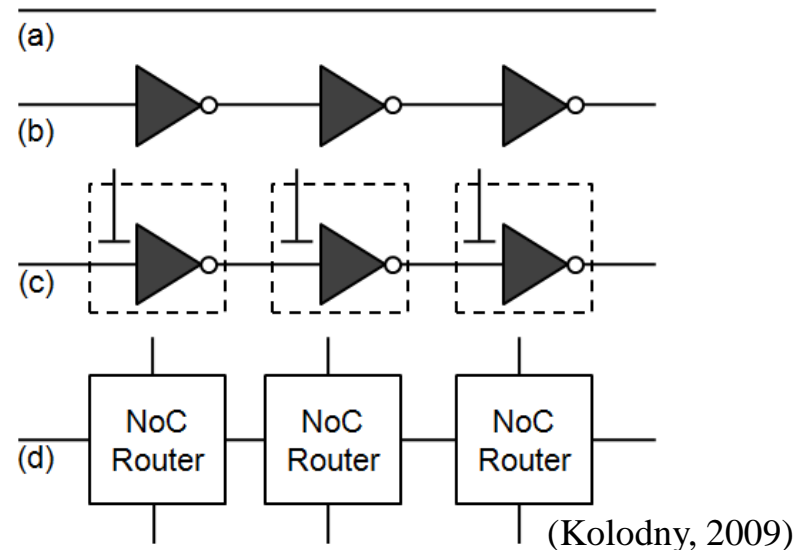
- Vamos eliminar a influência do fio!
 - Rede-em-Chip sem fio!
 - Rede-em-Chip óptica!
 - Rede-em-Chip reconfigurável!

Se o problema está no fio...

- Vamos eliminar a influência do fio!
 - Rede-em-Chip sem fio!
 - Rede-em-Chip óptica!
 - Rede-em-Chip reconfigurável!
 - Rede-em-Chip com fio, mas fio curto!

Do Barramento ao roteador

- Evolução da **interconexão global** para NoC.
 - (a) fio longo dominado pela resistência,
 - (b) adição de repetidores ou buffers,
 - (c) repetidores se tornam latches,
 - (d) latches evoluem para roteadores de NoC.

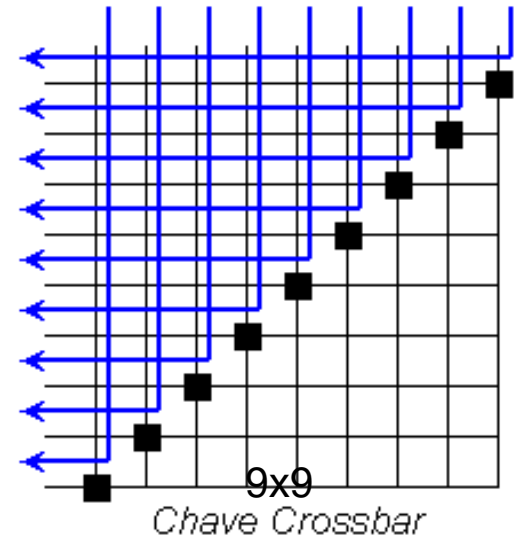
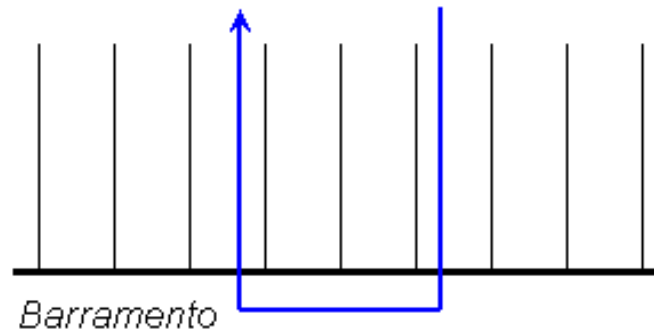


Rede-em-Chip

- Principais características:
 - Composta por roteadores,
 - Possui pacotes de rede,
 - Trabalha com protocolo de roteamento,
 - Possui diversas topologias,
 - Trabalha com Qualidade-de-Serviço (QoS),
 - É tolerante a falhas,
 - É escalável!

Rede-em-Chip

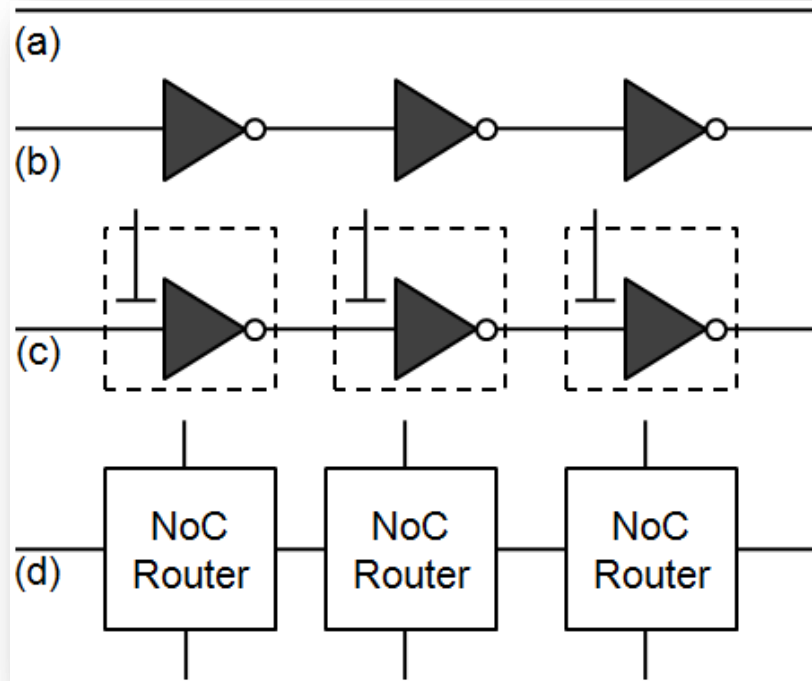
- Interconexões não escaláveis:
 - Barramento e Chave Crossbar.
 - Por que?



Qual a principal limitação do barramento?

Imaginem uma chave crossbar 99x99?
É viável? Por que?

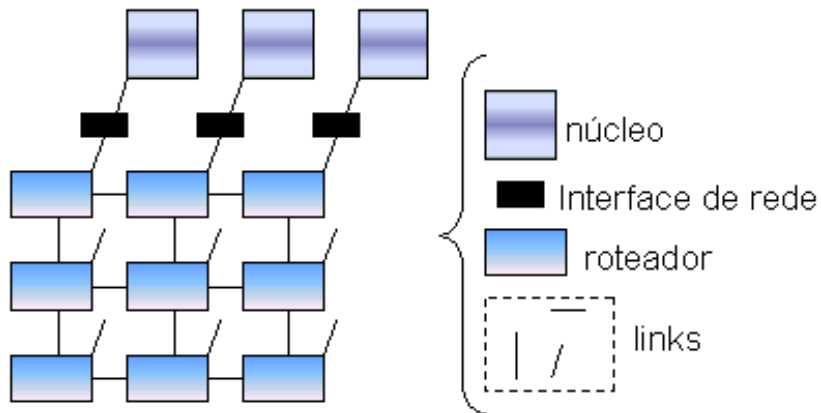
Origem da NoC?



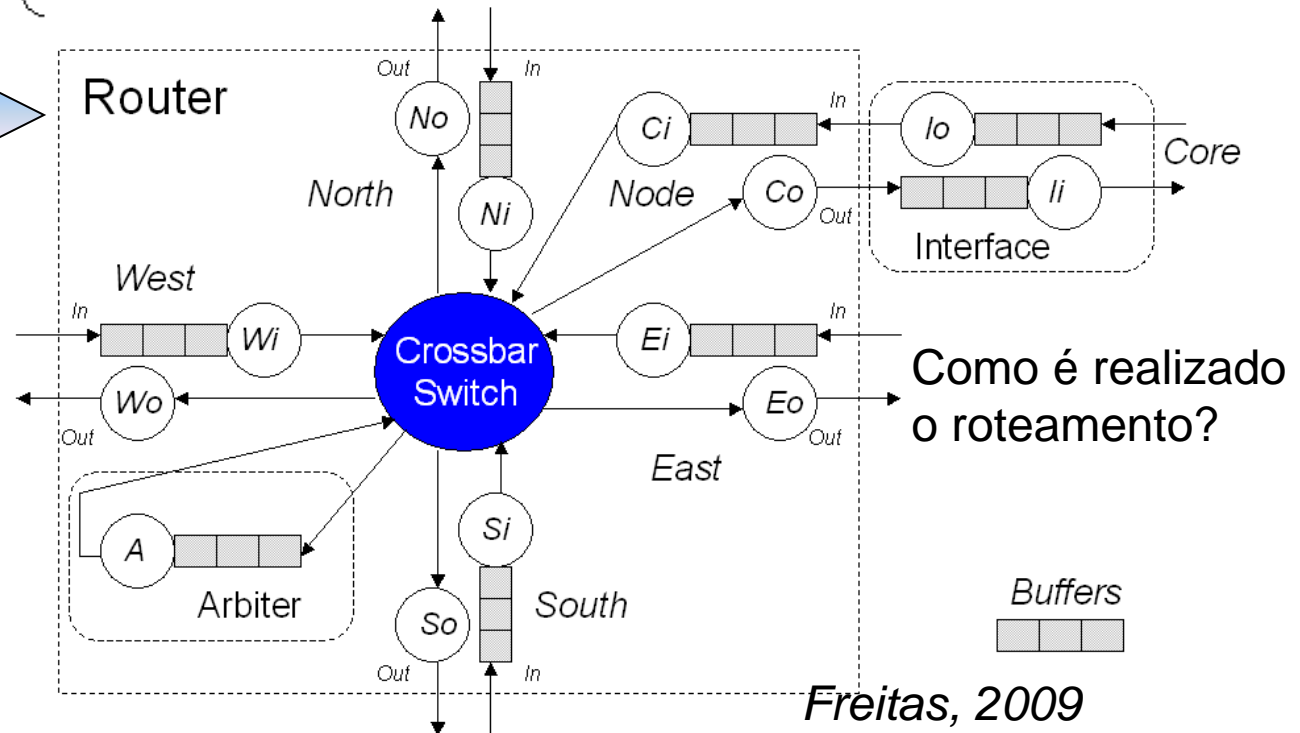
Kolodny, 2009

- a) Fio longo dominado pela resistência.
- b) Adição de repetidores ou buffers.
- c) Repetidores se tornam latches.
- d) Latches evoluem para roteadores de NoC.

Redes-em-Chip



Arquitetura típica de um roteador



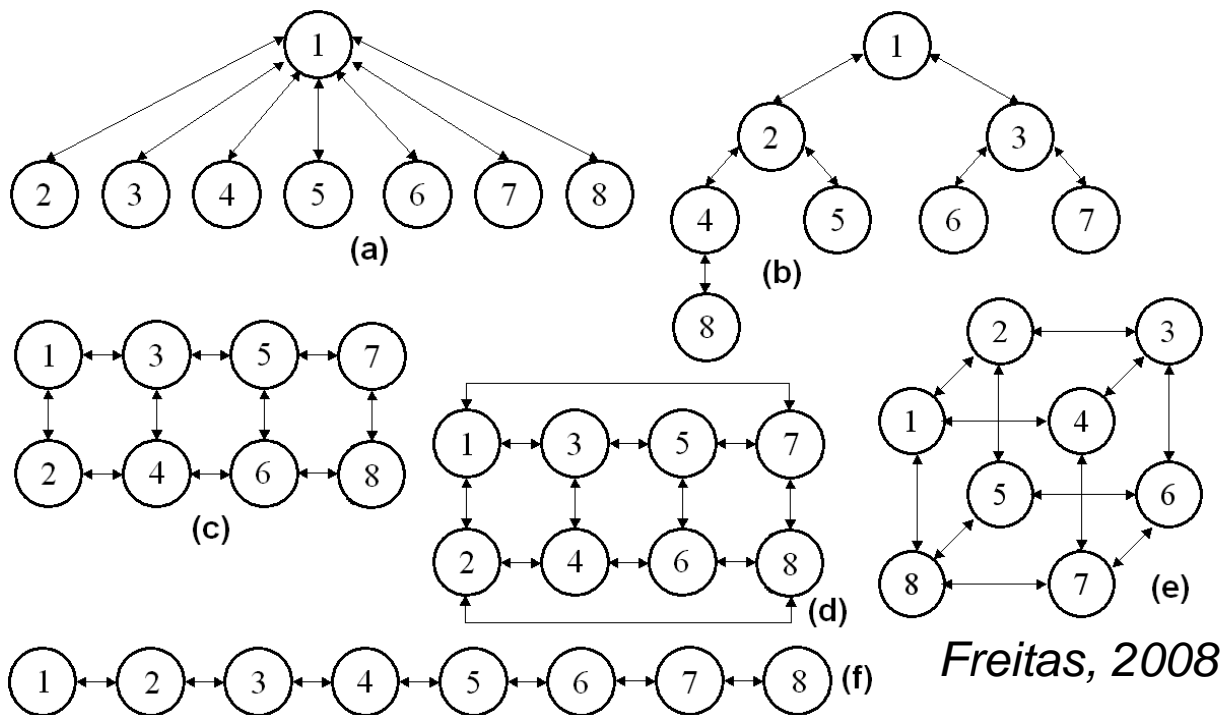
Tipos de Buffers

- **Buffers de entrada:** As técnicas de **arbitragem** são relativamente **simples**, possui uma melhor relação de área e potência, além de proporcionar um melhor **desempenho** para a chave crossbar.
- **Buffers de saída:** Em função de **N entradas** conectadas a cada um dos buffers de saída, a chave crossbar precisa ser **N vezes mais rápida**. A adoção de buffers de saída não é a mais adequada para alto desempenho. No entanto, existem vantagens em se tratando da **eliminação do bloqueio de pacotes** que não receberam permissão de envio porque **o primeiro pacote da fila ainda não teve liberação de uma determinada saída**. Este problema é conhecido como **head of the line blocking** e pode acontecer nas soluções com buffers de entrada.
- **Buffers de crosspoint:** Cada **ponto de conexão** da chave crossbar **possui um buffer**. É utilizada a técnica de roteamento chamada de self-routing. Neste caso, em cada crosspoint seria necessário além do buffer um decodificador para decisão de envio ou não do pacote. Esta solução aumenta o **tamanho** e a **potência** consumida da chave crossbar.

Preocupações no projeto de NoC

- *Deadlock*: é a representação de uma **dependência cíclica**. Neste caso, um pacote não consegue progredir e fica restrito a um subconjunto de estados ou roteadores.
- *Livelock*: é a representação de uma **contínua retransmissão** do pacote sem atingir o nó destino. Comum em protocolos de roteamento.
- *Starvation*: é a representação da **não alocação de um recurso** devido a **postergação indefinida** de acesso ao mesmo. Comum em protocolos de arbitragem.

Redes-em-Chip (topologias)



Freitas, 2008

(a) Estrela

(b) Árvore

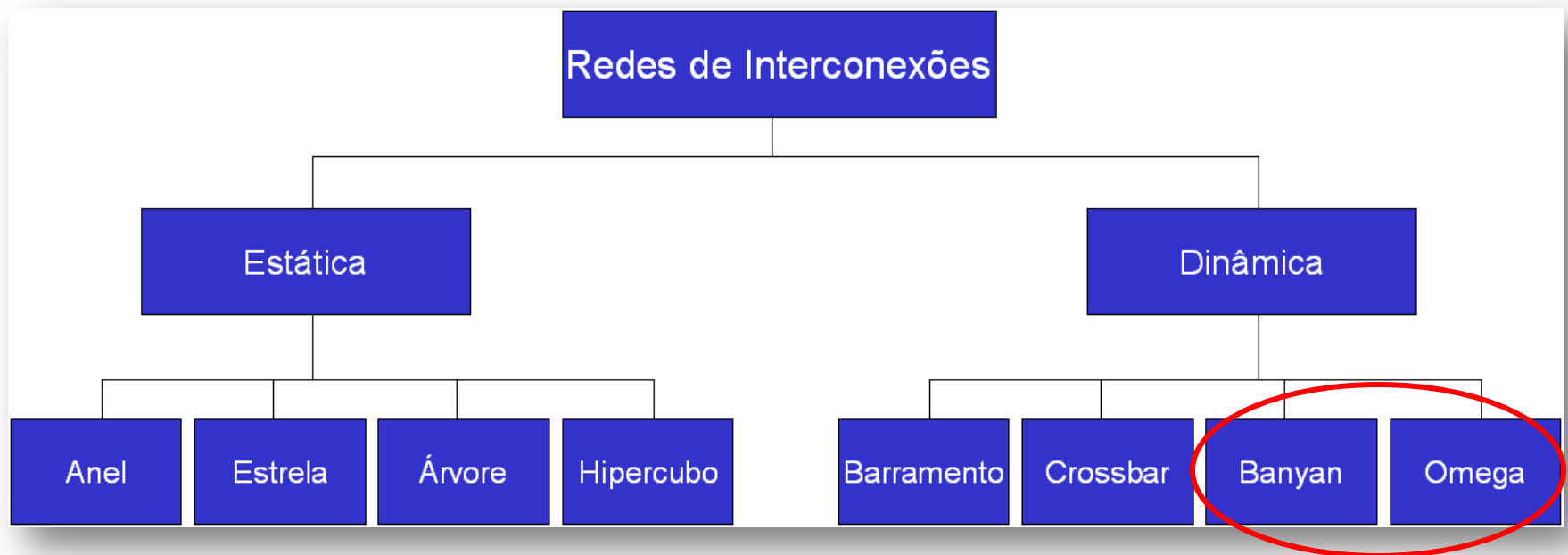
(c) Mesh

(d) Torus

(e) Hipercubo

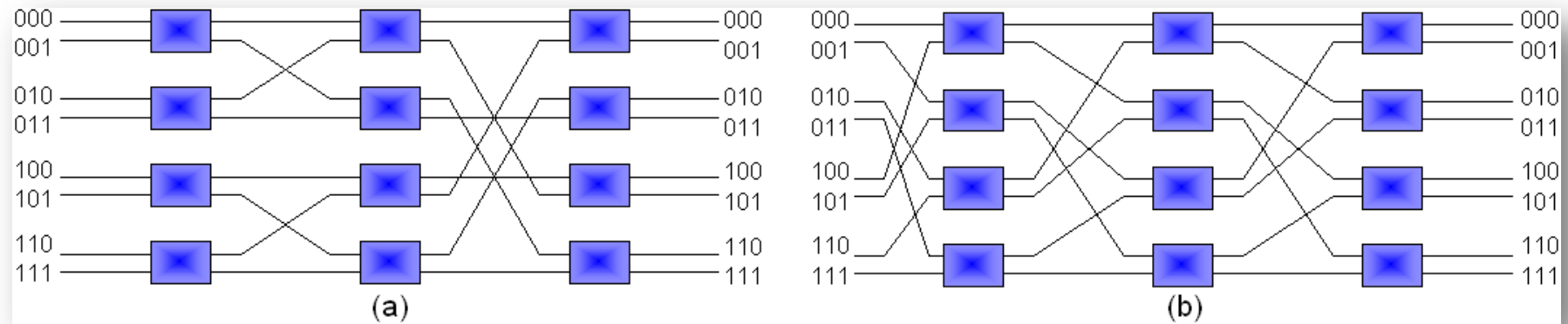
(f) Pipeline

Redes-em-Chip (Classificações)



Freitas, 2009

Redes Dinâmicas (multi-nível)



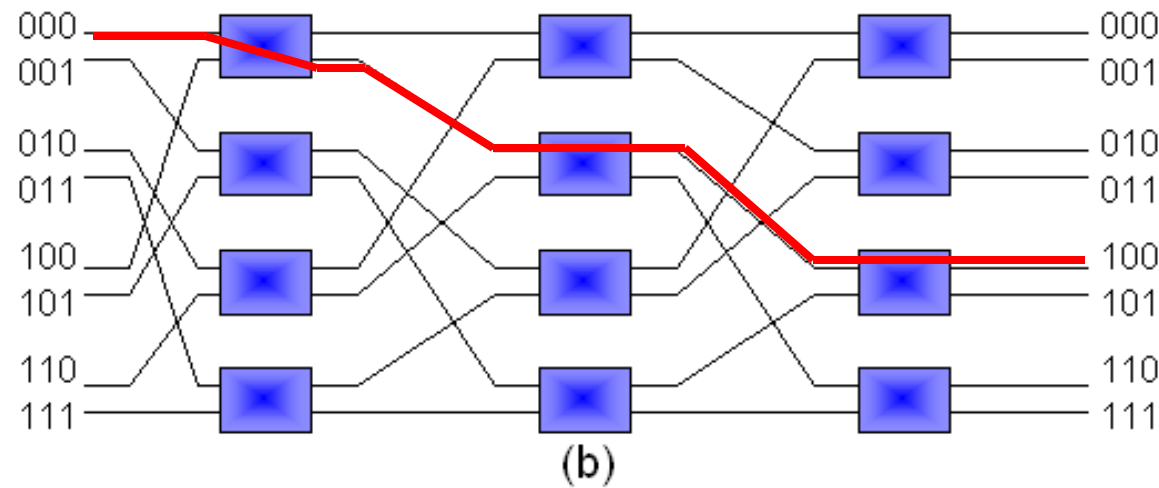
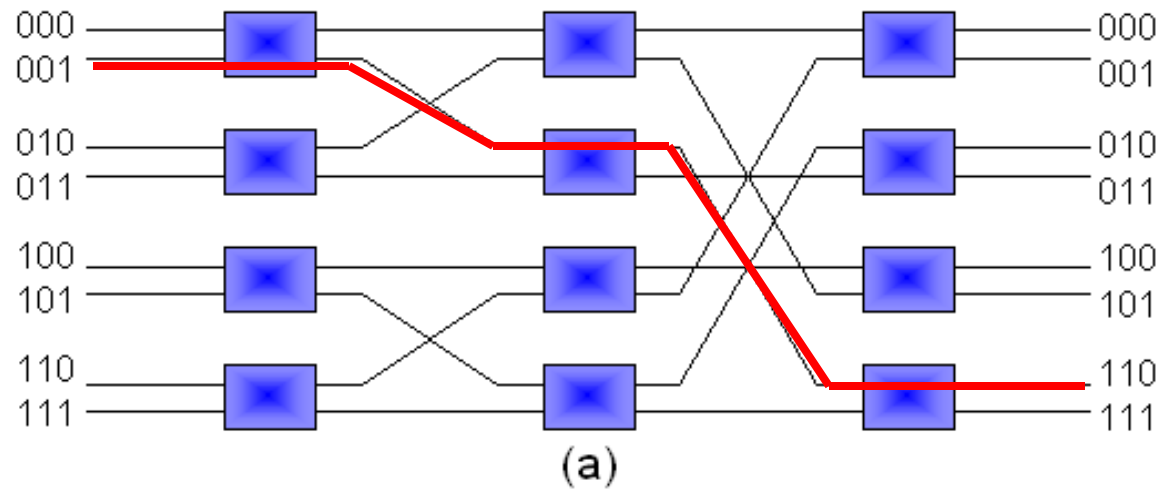
Banyan

Omega

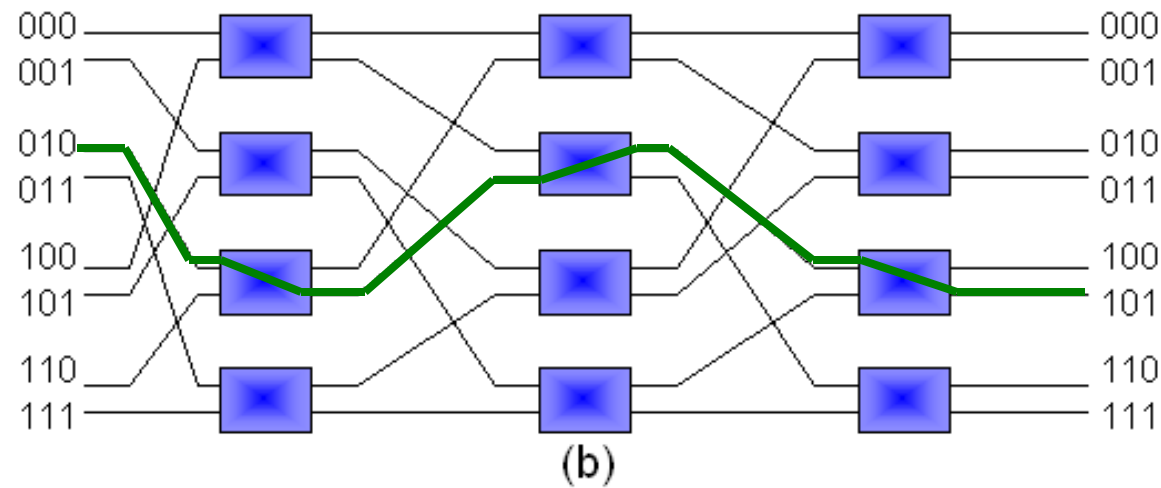
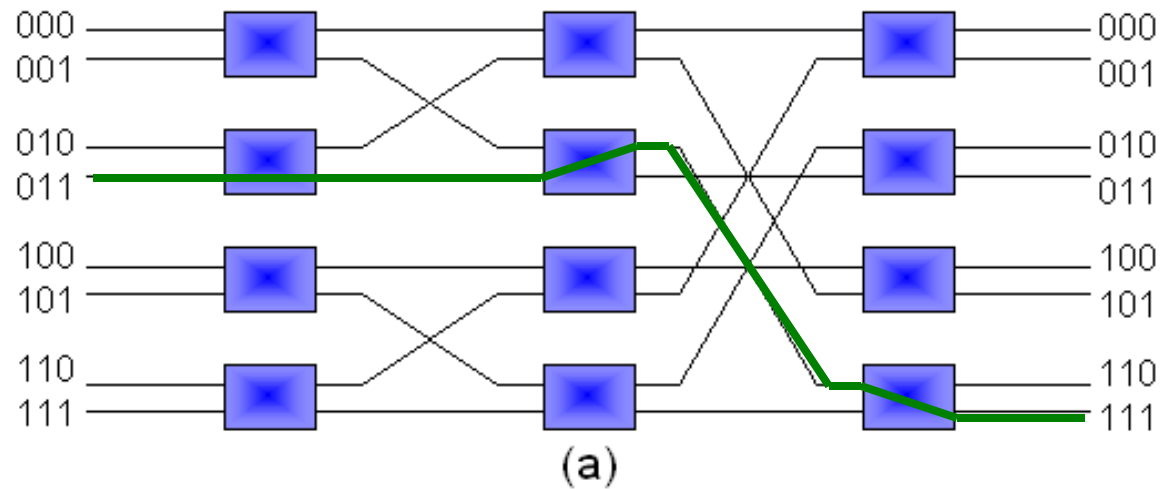
Freitas, 2009

Um único caminho entre entrada e saída faz com que o roteamento seja eficiente, podendo ser feito de forma descentralizada.

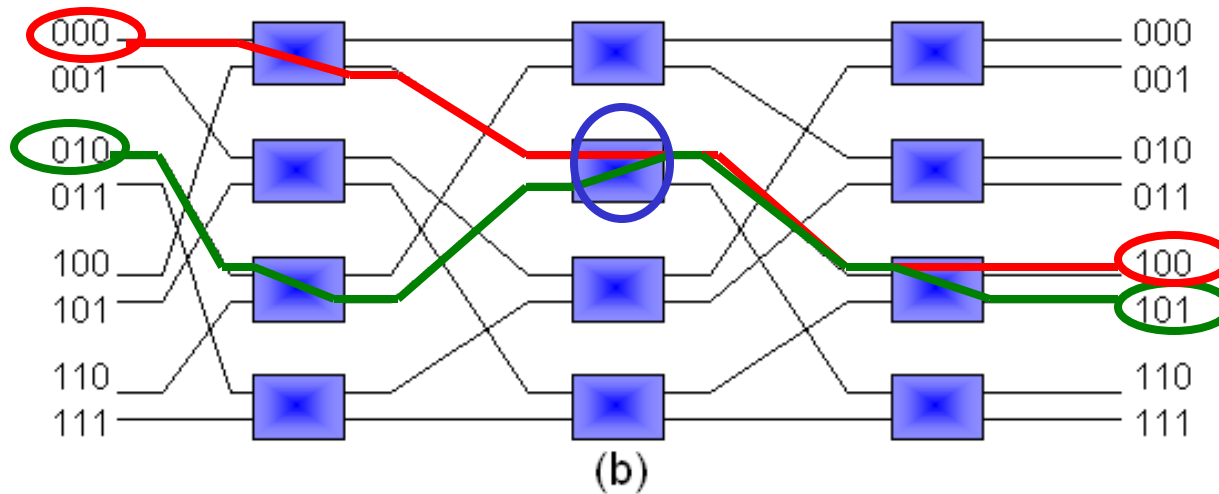
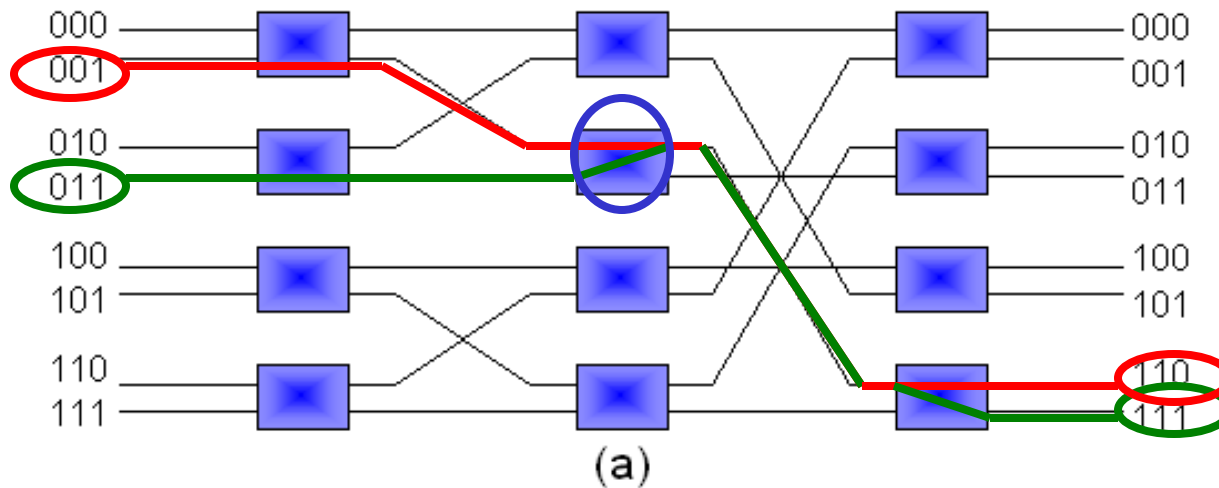
Redes Dinâmicas (multi-nível)



Redes Dinâmicas (multi-nível)



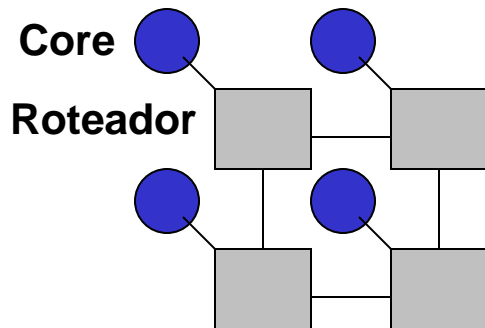
Redes Dinâmicas (multi-nível bloqueante)



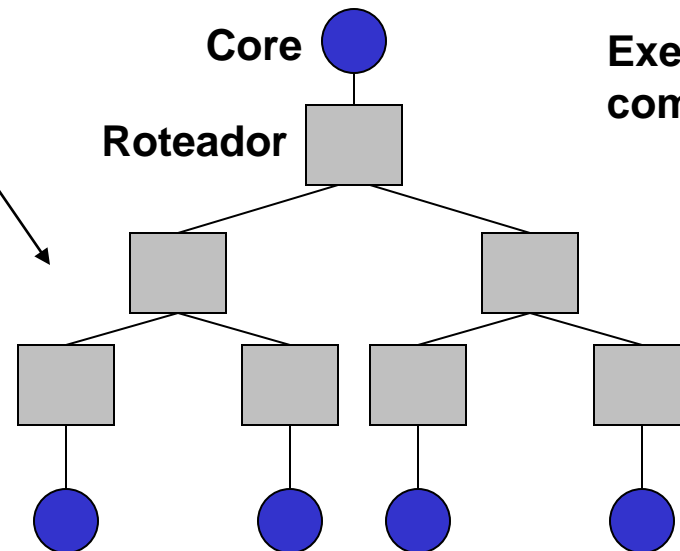
Redes-em-Chip (Classificações)

Topologia	Estratégia de Transferência	Método de Controle	Estrutura do Caminho
Estrela	Indireta	Roteamento centralizado	Dedicado
Árvore	Indireta	Roteamento Descentralizado	Dedicado
Mesh	Direta	Roteamento Descentralizado	Dedicado
Torus	Direta	Roteamento Descentralizado	Dedicado
Hipercubo	Direta	Roteamento Descentralizado	Dedicado
Pipeline	Direta	Roteamento Descentralizado	Dedicado

Freitas, 2009



Rede direta



Rede indireta

Exemplo compartilhado?

Viabilidade das NoCs

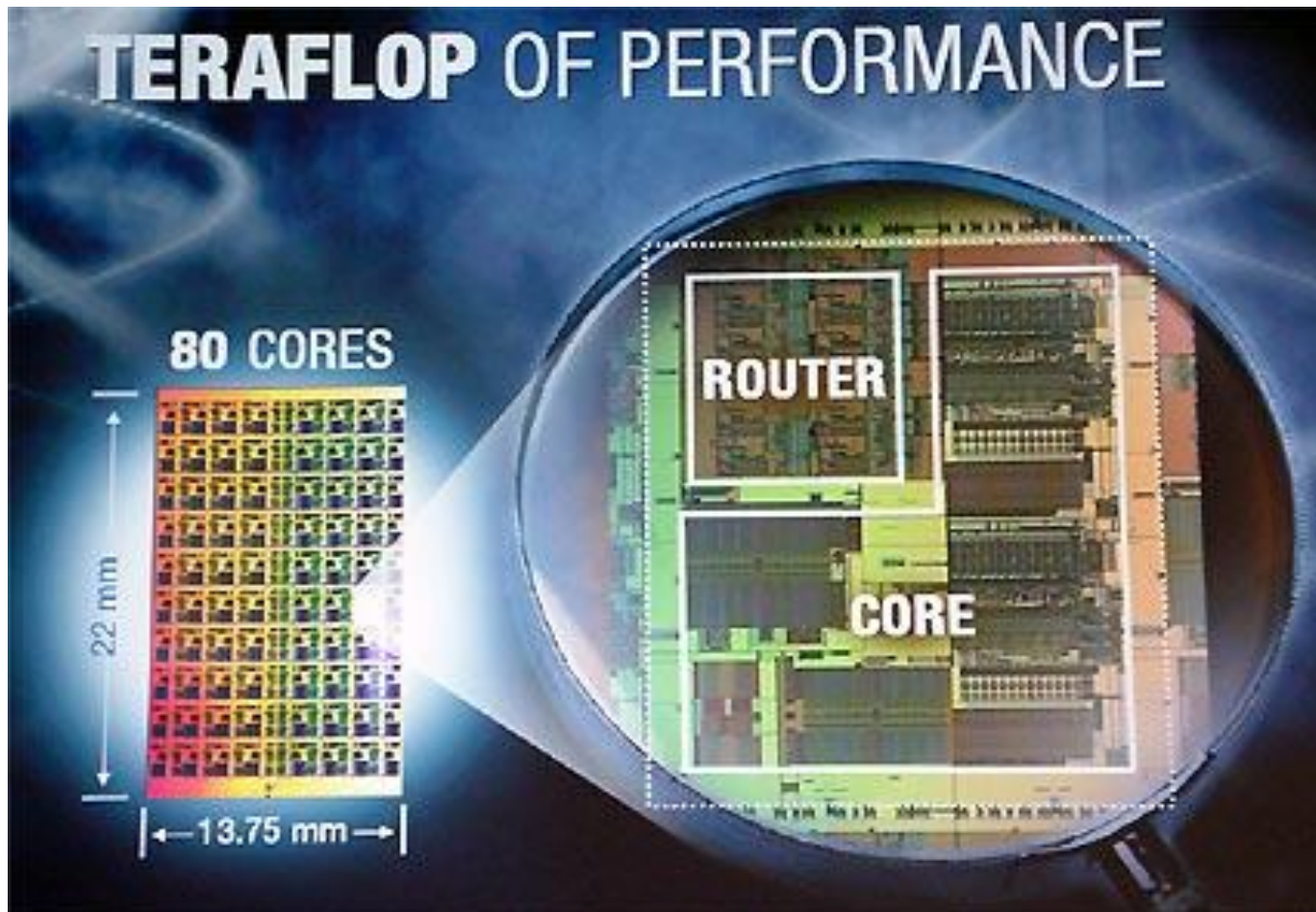
Tipo de Interconexão		Prós (+) e Contras (-)
Barramento	Fio	O aumento do fio aumenta a resistência degradando o desempenho. -
Chave Crossbar		O aumento do fio aumenta a resistência degradando o desempenho. -
Network-on-Chip		Os fios são ponto-a-ponto entre roteadores e o desempenho não degrada em função do aumento de nós. +
Barramento	Árbitro	O árbitro é um gargalo à medida que o número de nós aumenta. -
Chave Crossbar		O árbitro pode ser centralizado ou descentralizado e não é o fator principal para degradação do desempenho em função do aumento dos nós. +-
Network-on-Chip		As decisões de roteamento são distribuídas e não representam um gargalo. +
Barramento	Largura de banda	A largura de banda é limitada e compartilhada por todos os nós. -
Chave Crossbar		Cada interconexão é independente e a largura de banda de comunicação por conexão não é afetada pelas demais. +
Network-on-Chip		A largura de banda não é afetada pelo aumento da rede. +
Barramento	Latência	Latência é afetada pelo fio. +
Chave Crossbar		Latência é afetada pelo fio. +
Network-on-Chip		Latência é afetada pelas contenções em roteadores -
Barramento	Compatibilidade	Em sua maioria são compatíveis com qualquer IP (<i>Intellectual Property</i>) incluindo os softwares. +
Chave Crossbar		Em sua maioria são compatíveis com qualquer IP (<i>Intellectual Property</i>) incluindo os softwares. +
Network-on-Chip		São necessários adaptadores (<i>wrappers</i>) entre os IPs e os softwares precisam de sincronização em sistemas <i>multi-core</i> . -
Barramento	Complexidade	Conceitos simples e bem compreendidos. +
Chave Crossbar		Conceitos simples e bem compreendidos. +
Network-on-Chip		Projetistas precisam de uma reeducação em função dos novos conceitos. -

Adaptado de
Bjerregaard, 2006

Protocolos

- Políticas e estratégias de transporte de dados em uma NoC é de responsabilidade dos protocolos.
- A definição do protocolo descreve as principais características de funcionamento da rede.
- Os protocolos precisam ser capazes de:
 - Garantir a entrega de dados.
 - Confiabilidade da rede.
 - A melhor rota.
 - Melhor desempenho, entre outros.

Processadores many-Core



Intel, 2006

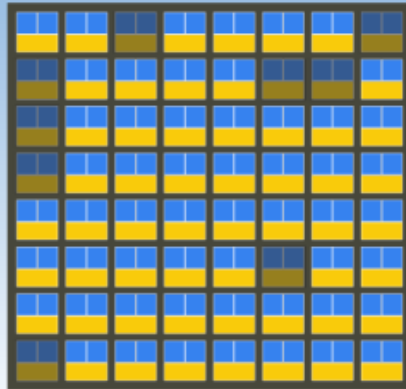
Processadores many-core

A tera-scale platform would intelligently activate only those cores that are needed at a particular time, while all others sleep. Intel's Teraflop Research Chip tests this capability.

(a)

Sleeping Cores

Active Cores



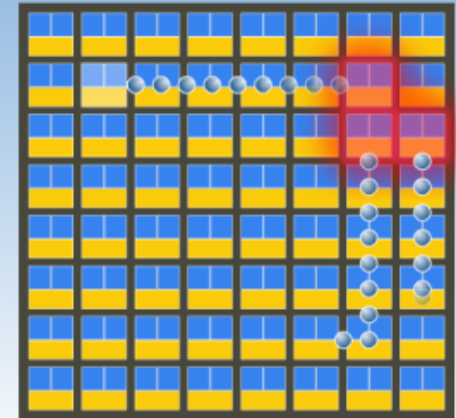
Heat can limit core speed. Peak performance could be increased by intelligently redistributing workloads, spreading them evenly across the chip to avoid any hot spots.

(b)

Hot Core

Normal Temp. Core

Workload

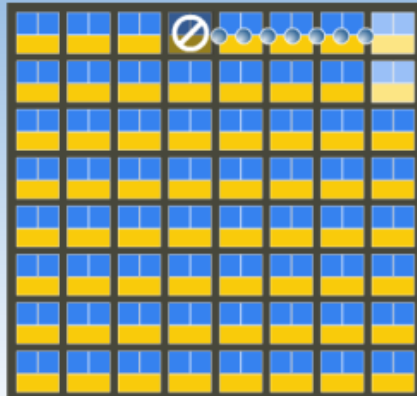


Having so many cores would allow a few to be held in reserve as "spares." If over time and use a core should wear out, a spare could immediately be brought online to maintain performance.

(c)

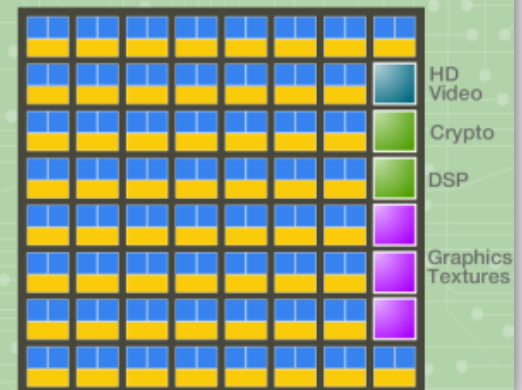
Spares held in Reserve

Problem Core



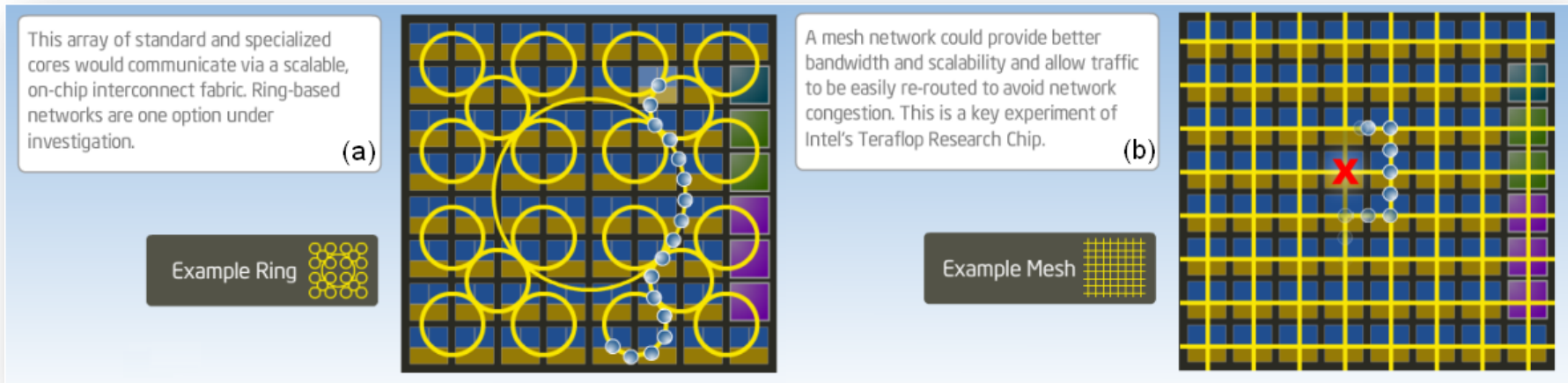
To expand functionality, some dedicated hardware engines could be integrated to accelerate multimedia, networking, security, or other tasks. The number and type could vary by target platform

(d)



Intel, 2007

Processadores many-core

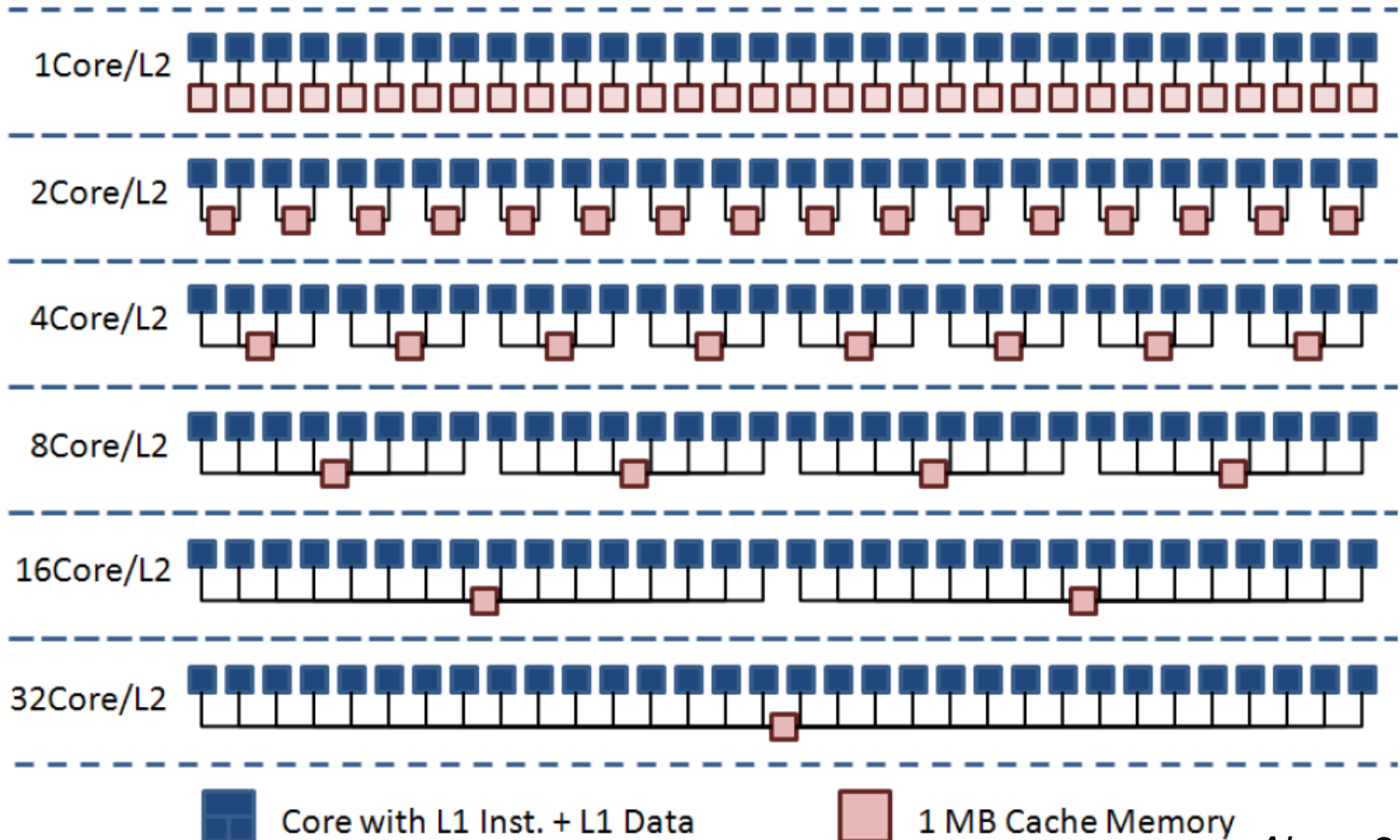


Intel, 2007

Por que a Intel estuda dois tipos de topologias?

Processadores many-core

Cache compartilhada



Arquiteturas paralelas

Conceitos Principais e Classificações

Arquiteturas monoprocessadas

- Os processos compartilham o mesmo processador.
- S.O. pode ser implementado com o conceito de monoprogramação ou multiprogramação.

Arquiteturas monoprocessadas

- Monoprogramação: recursos do computador alocados para uma única tarefa até o seu término.
- Multiprogramação: processador alterna a execução de vários processos.

Arquiteturas multiprocessadas

- Multiprocessada: vários elementos de processamento.
- Tipos de arquiteturas multiprocessadas:
 - Memória compartilhada
 - Memória distribuída

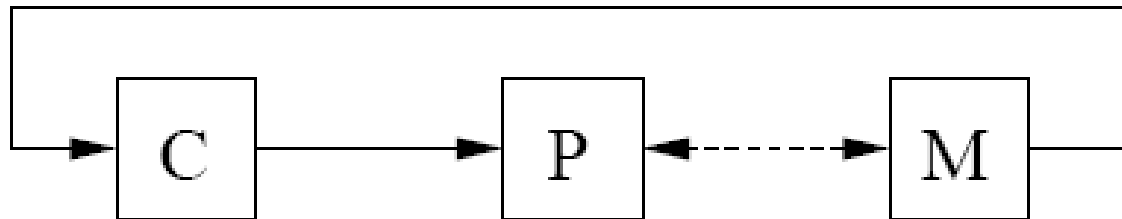
Classificações de Flynn

Classificação de Flynn (*Flynn, 1972*) segundo o
fluxo de instruções e fluxo de dados.

	SD (<i>Single Data</i>)	MD (<i>Multiple Data</i>)
SI (<i>Single Instruction</i>)	SISD Máquinas von Neumann	SIMD Máquinas Array
MI (<i>Multiple Instruction</i>)	MISD Sem representante até agora	MIMD Multiprocessadores e multicomputadores

De Rose (2003)

Classificações de Flynn

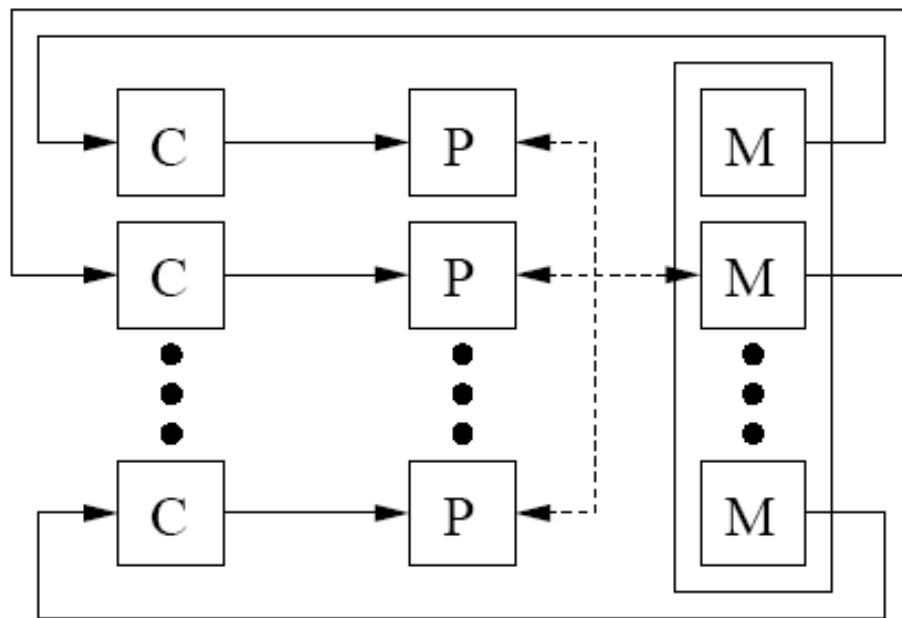


—→ Instruções
←---→ Dados

De Rose (2003)

SISD

Classificações de Flynn

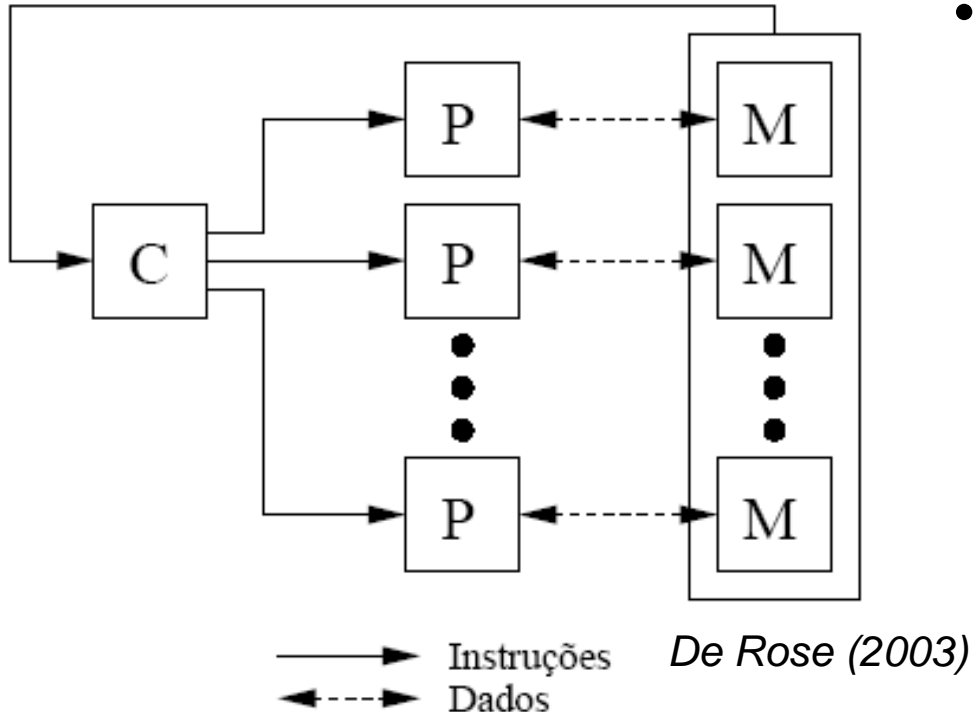


De Rose (2003)

—→ Instruções
←--- Dados

MISD

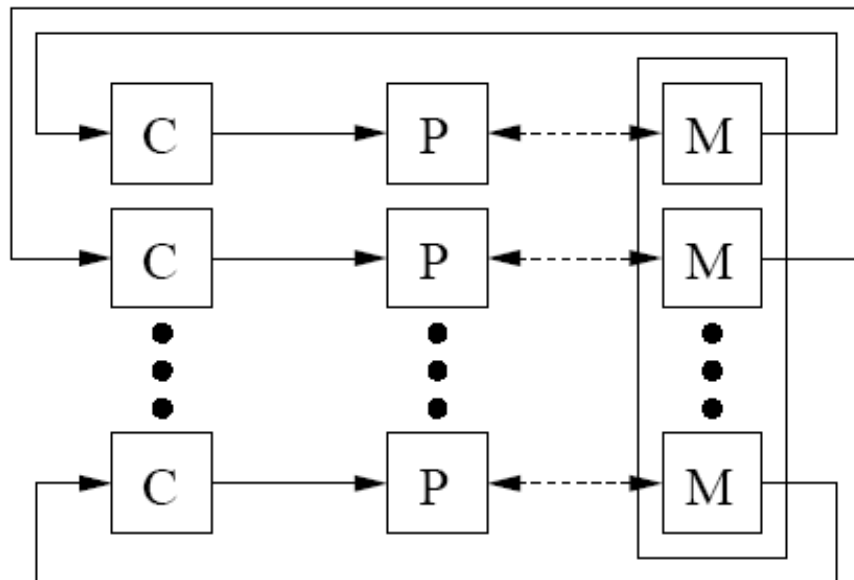
Classificações de Flynn



SIMD

- Processadores vetoriais:
 - Vetor é um conjunto de dados escalares do mesmo tipo, armazenados em memória.
 - Processamento vetorial ocorre quando executamos operações aritméticas ou lógicas sobre vetores.
 - Um processador escalar opera sobre um ou um par de dados.

Classificações de Flynn

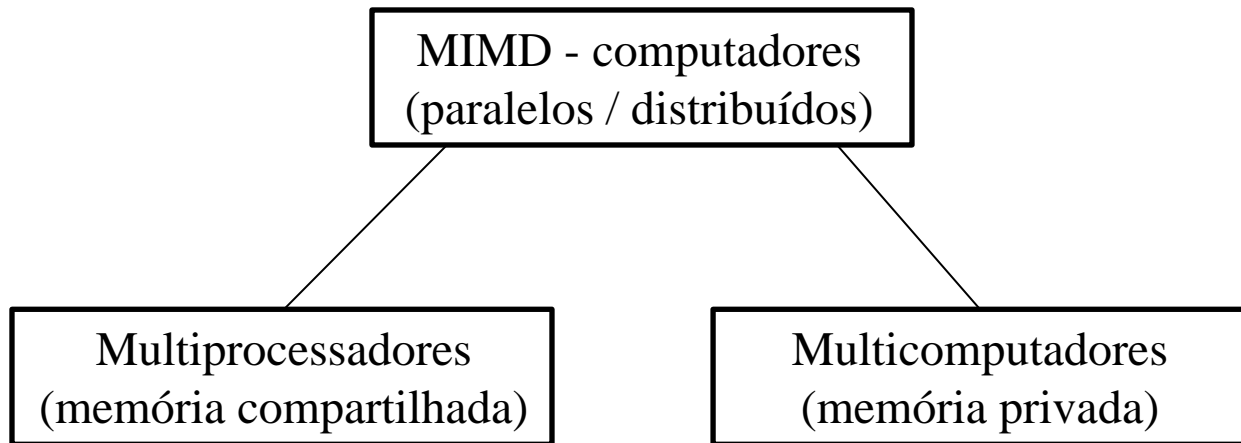


—→ Instruções
←--- Dados

De Rose (2003)

MIMD

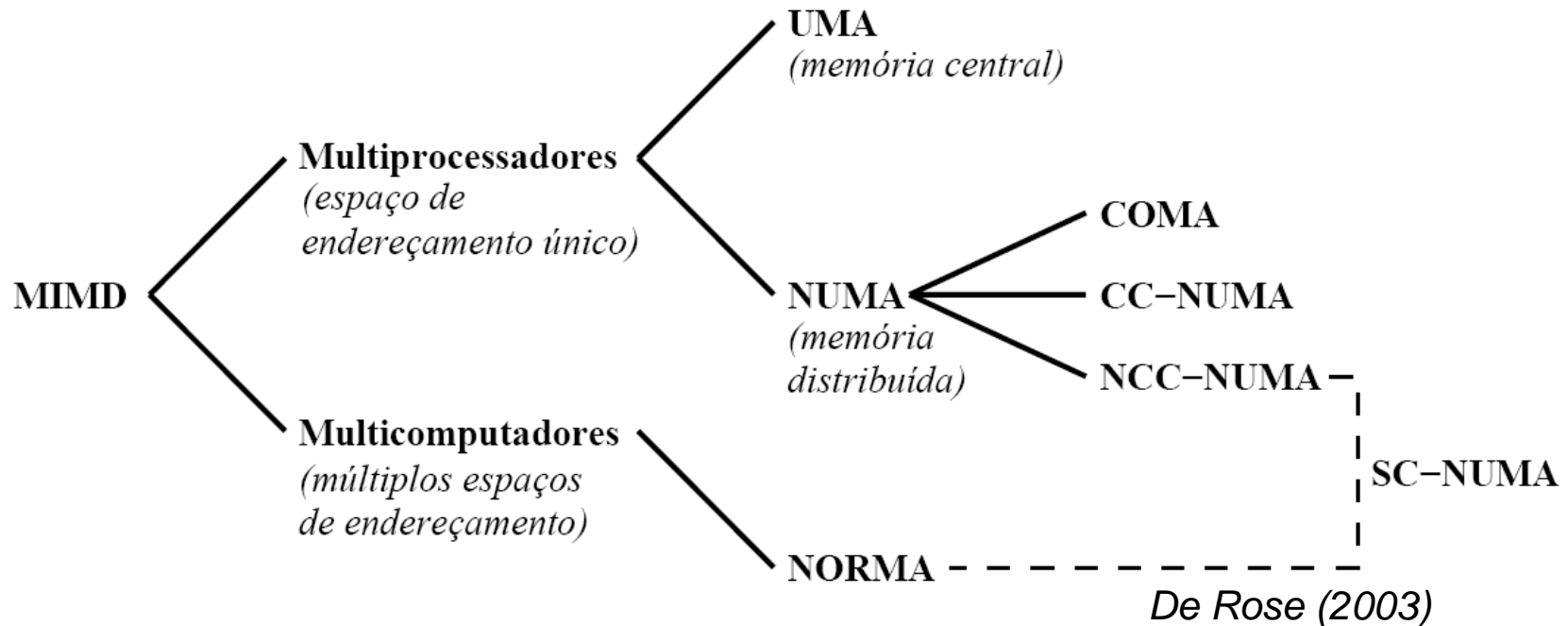
Subdivisão da classe MIMD



Classificações segundo compartilhamento de memória

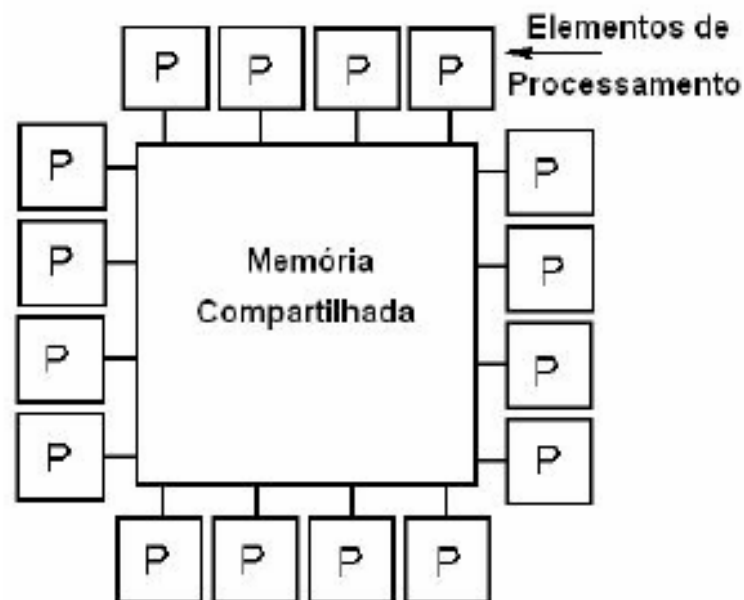
- UMA: Uniform Memory Access
- NUMA: Non-Uniform Memory Access
- CC-NUMA: Cache-Coherent NUMA
- NCC-NUMA: Non-Cache-Coherent NUMA
- SC-NUMA: Software-Coherent NUMA
- COMA: Cache-Only Memory Architecture
- NORMA: Non-Remote Memory Access

Classificações segundo compartilhamento de memória



A linha tracejada indica que através de um software que implemente coerência de *cache*, as máquinas NCC-NUMA e NORMA podem se transformar em máquinas SC-NUMA.

Memória Compartilhada



?

Memória Compartilhada

- Elementos de processamento compartilham a mesma memória.
- Programação realizada através de variável compartilhada. Maior facilidade na construção de programas paralelos.

Memória Compartilhada

- Neste tipo de arquitetura existe uma limitação de número de nós.
- Escalabilidade não é total.

Memória Compartilhada

- Programação em memória compartilhada é realizada com threads.
- Exemplos:
 - Pthreads
 - OpenMP

Memória Compartilhada

- O desempenho neste tipo de sistema é maior se consideramos no seu projeto o uso de memória cache.
- Surge o problema de coerência de cache.
- Solução em software ou em hardware.

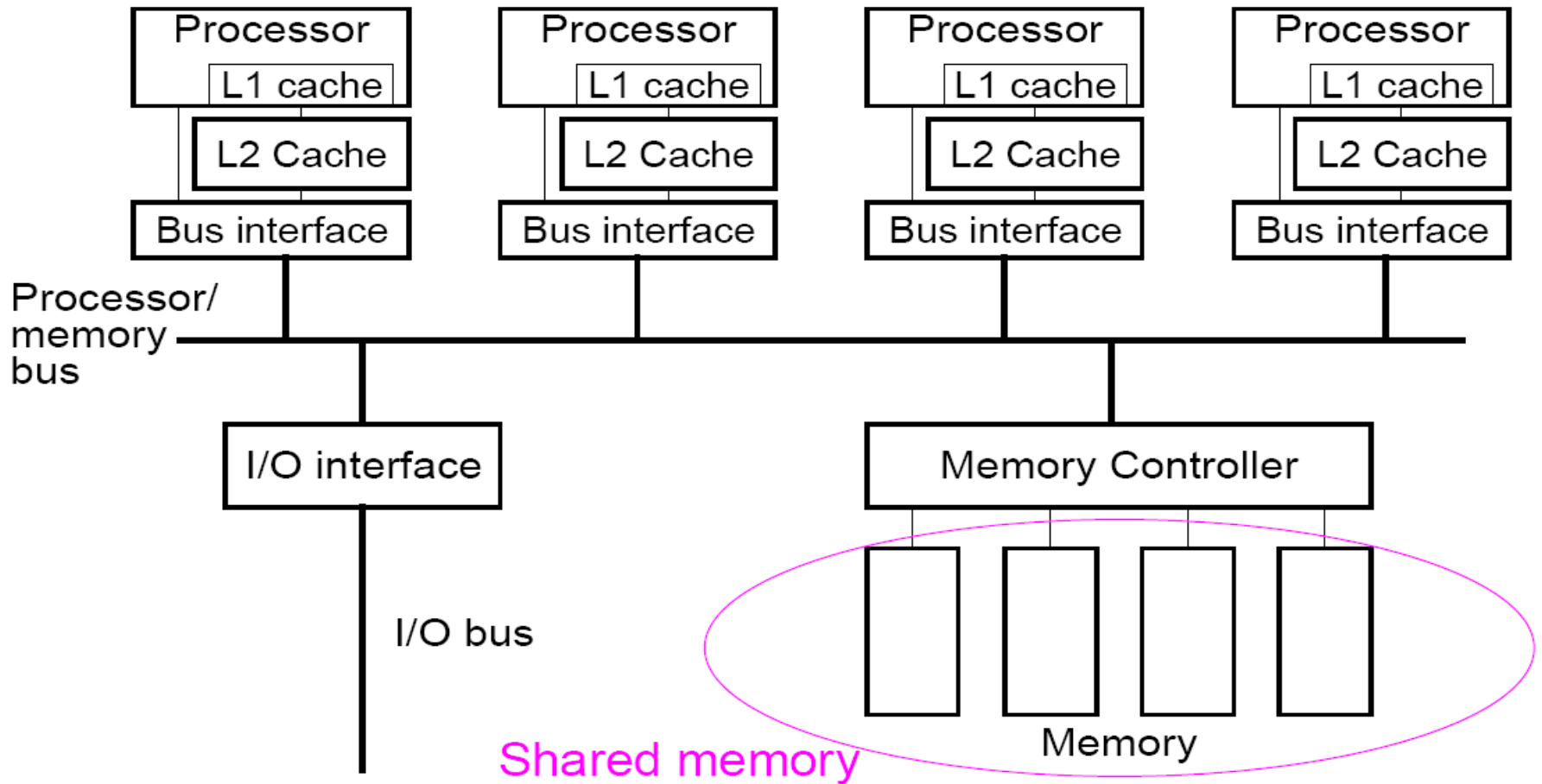
Memória Compartilhada

- As máquinas de memória compartilhada podem ser UMA ou NUMA.
 - UMA – Uniform memory access
 - NUMA – Non-uniform memory access

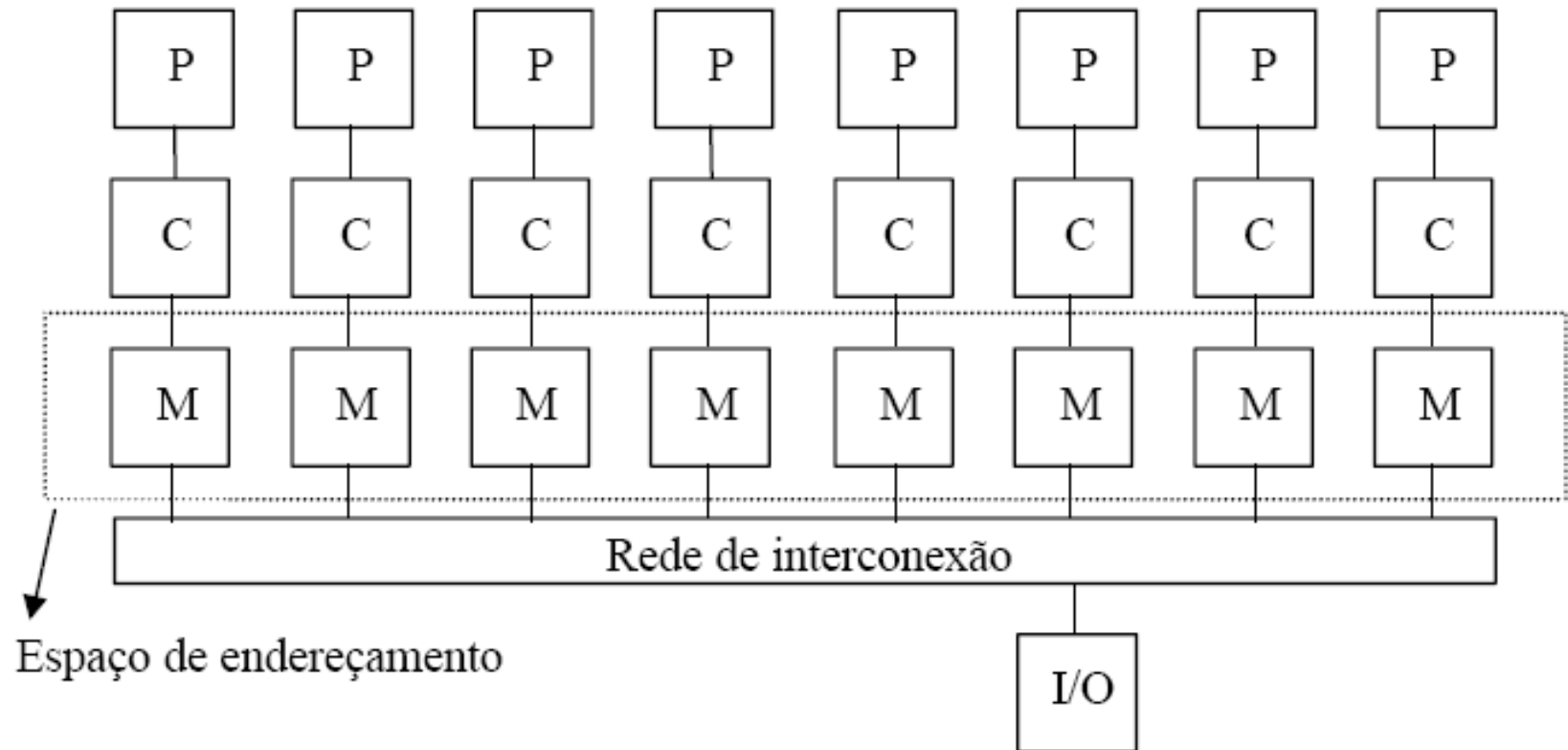
Memória Compartilhada

- Exemplos de arquiteturas de memória compartilhada:
 - Dual Pentium (UMA)
 - Quad Pentium (UMA)

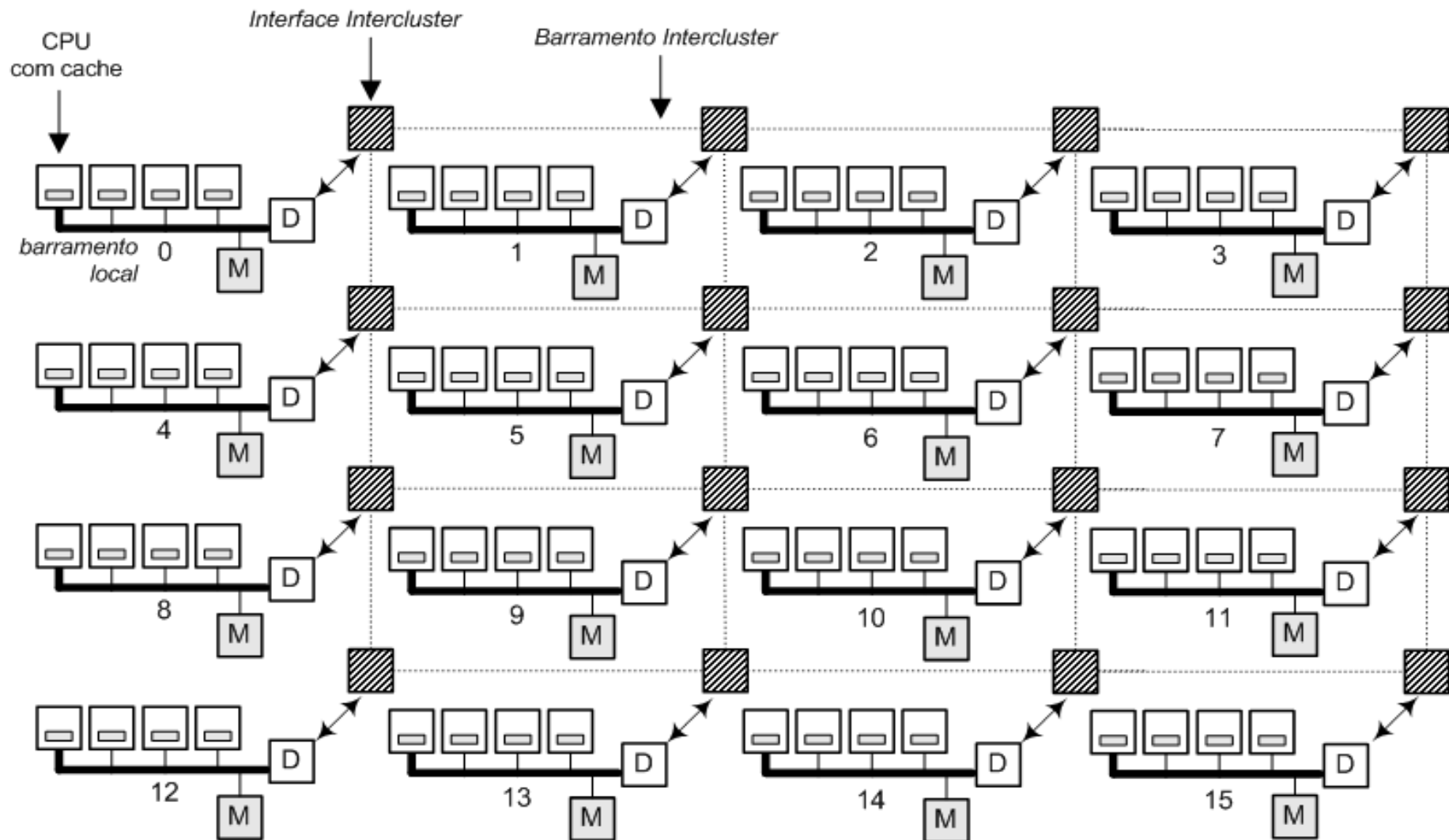
UMA



NUMA



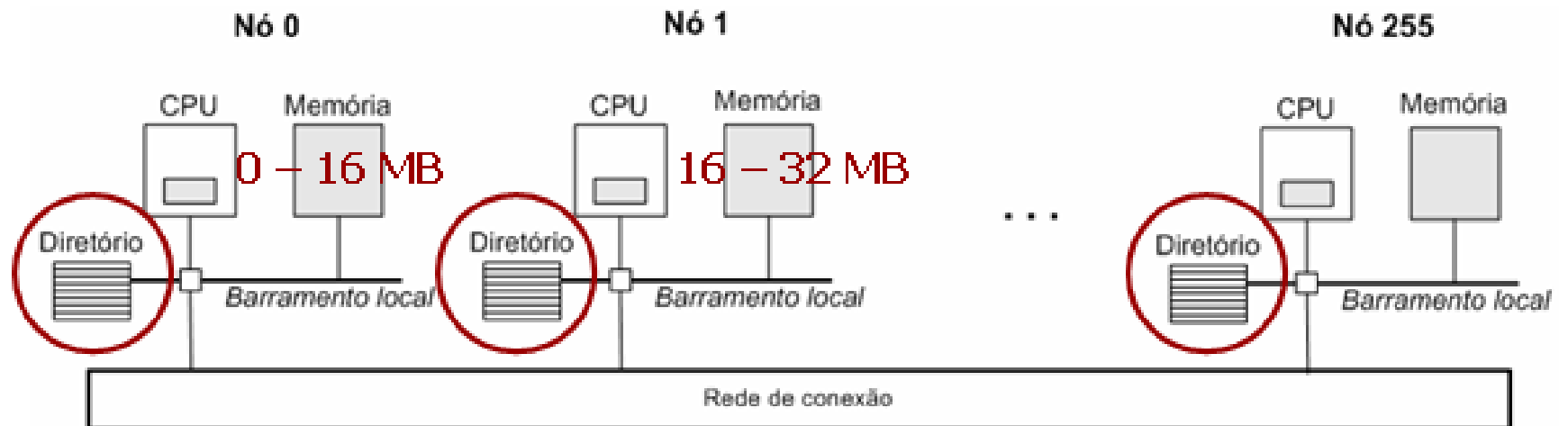
NUMA



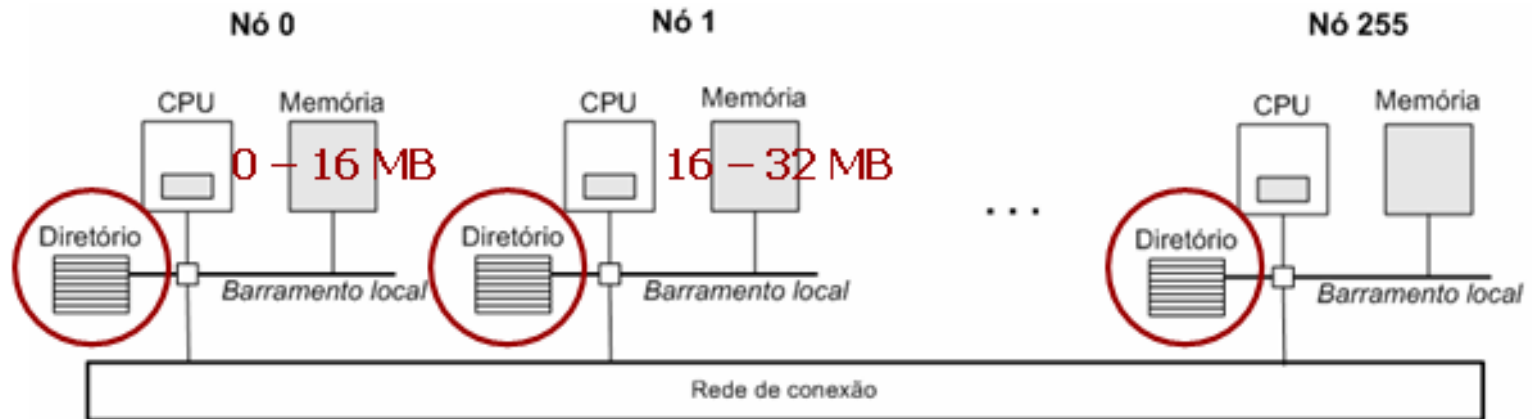
NUMA

- Existem dois tipos de arquiteturas NUMA:
 - NC-NUMA: NUMA que não utiliza cache
 - CC-NUMA: (Cache-Coherent NUMA) -
NUMA com cache (e coerência de cache)

CC-NUMA

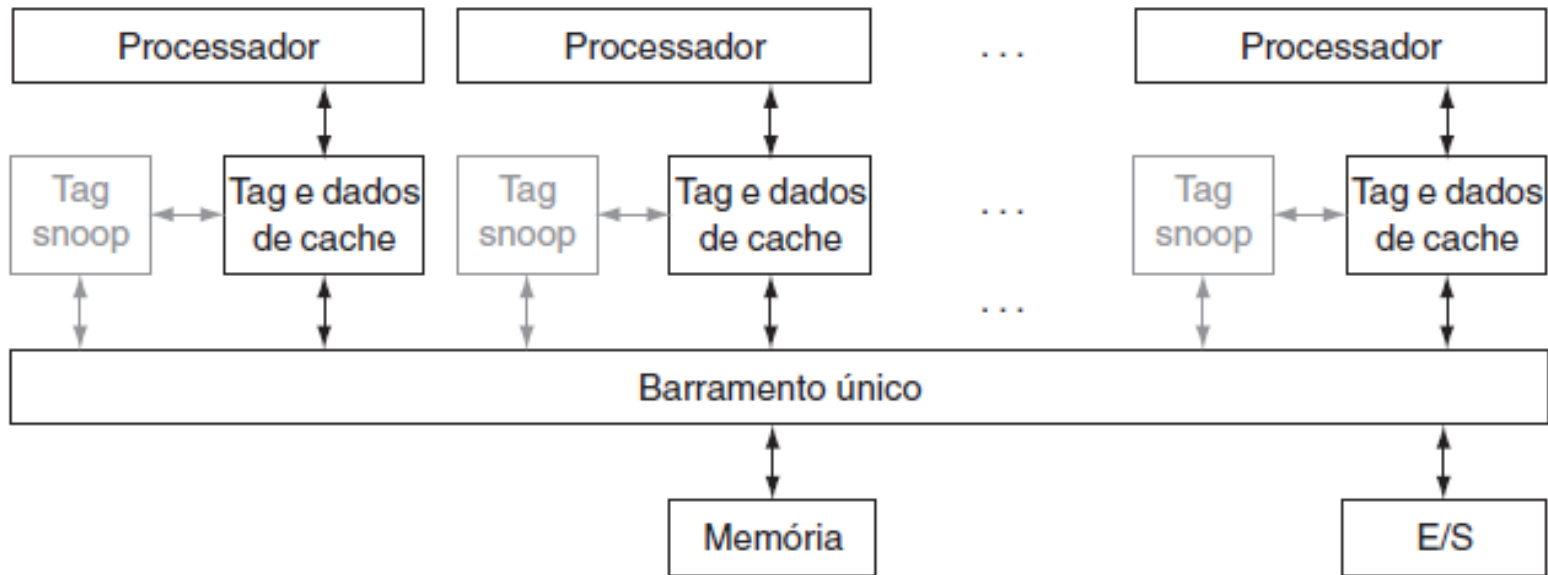


Coerência de cache (Diretório)



- Cada módulo de memória possui um diretório local que armazena as informações sobre onde as cópias dos blocos estão residentes.
- Os protocolos baseados em diretório enviam comandos de consistência seletivamente para aquelas caches que possuem uma cópia válida do bloco de dados compartilhado

Coerência de cache (Snooping)



Coerência de cache com snooping. Um **método** para manter coerência de cache em que todos os controladores de cache monitoram o barramento para determinar se possuem ou não uma cópia do bloco desejado.

Write-invalidate. Um tipo de protocolo de snooping em que o processador de escrita faz com que todas as cópias em outras caches sejam invalidadas antes de mudar sua cópia local, o que permite atualizar os dados locais até que outro processador os solicite.

Coerência de cache (MSI, MESI, MOESI)

- **Estados que cada bloco na memória cache possui:**
 - **Inválido (I):** bloco inválido na memória cache.
 - **Shared (S) ou Compartilhado:** bloco só foi lido e pode haver cópias em outras memórias cache.
 - **Modificado (M):** apenas esse cache possui cópia do bloco e a memória principal não está atualizada
 - ***Exclusivo (E):** Apenas esse cache possui cópia do bloco e a memória principal está atualizada.*
 - **Owner (O) :** Esse cache supre o dado em caso de leitura com falha no barramento uma vez que a memória não está atualizada. Outros caches podem ter cópia do dado.

Memória Distribuída

- Grupo de computadores autônomos (nós) que trabalham juntos como um recurso único.
- Os nós são interconectados através de redes de alto desempenho.

Memória Distribuída

- Escalabilidade absoluta e incremental.
- Alta disponibilidade.
- Excelente custo benefício.

Memória Distribuída

- Comunicação realizada através de passagem de mensagens.
 - MPI (Message Passing Interface) ou
 - PVM (Parallel Virtual Machine).
- Podemos usar o conceito de memória compartilhada. Software ou suporte em hardware.

Memória Distribuída

- Cluster ou aglomerado de computadores.
- Grids ou grades computacionais.
- São usados em gerenciadores de bancos de dados, com servidores WEB.
- São usados principalmente com processamento paralelo.

Cluster de Computadores



Cluster? de Computadores

Rank	Site	Computer/Year Vendor	Cores	R _{max}	R _{peak}	Power	
1	Oak Ridge National Laboratory United States	Jaguar - Cray XT5-HE Opteron Six Core 2.6 GHz / 2009 Cray Inc.	224162	1759.00	2331.00	6950.60	MPP
2	DOE/NNSA/LANL United States	Roadrunner - BladeCenter QS22/LS21 Cluster, PowerXCell 8i 3.2 Ghz / Opteron DC 1.8 GHz, Voltaire Infiniband / 2009 IBM	122400	1042.00	1375.78	2345.50	Cluster
3	National Institute for Computational Sciences/University of Tennessee United States	Kraken XT5 - Cray XT5-HE Opteron Six Core 2.6 GHz / 2009 Cray Inc.	98928	831.70	1028.85		MPP
4	Forschungszentrum Juelich (FZJ) Germany	JUGENE - Blue Gene/P Solution / 2009 IBM	294912	825.50	1002.70	2268.00	MPP
5	National SuperComputer Center in Tianjin/NUDT China	Tianhe-1 - NUDT TH-1 Cluster, Xeon E5540/E5450, ATI Radeon HD 4870 2, Infiniband / 2009 NUDT	71680	563.10	1206.19		Cluster
6	NASA/Ames Research Center/NAS United States	Pleiades - SGI Altix ICE 8200EX, Xeon QC 3.0 GHz/Nehalem EP 2.93 Ghz / 2009 SGI	56320	544.30	673.26	2348.00	MPP
7	DOE/NNSA/LLNL United States	BlueGene/L - eServer Blue Gene Solution / 2007 IBM	212992	478.20	596.38	2329.60	MPP
8	Argonne National Laboratory United States	Blue Gene/P Solution / 2007 IBM	163840	458.61	557.06	1260.00	MPP
9	Texas Advanced Computing Center/Univ. of Texas United States	Ranger - SunBlade x6420, Opteron QC 2.3 Ghz, Infiniband / 2008 Sun Microsystems	62976	433.20	579.38	2000.00	Cluster
10	Sandia National Laboratories / National Renewable Energy Laboratory United States	Red Sky - Sun Blade x6275, Xeon X55xx 2.93 Ghz, Infiniband / 2009 Sun Microsystems	41616	423.90	487.74		Cluster

R_{max}: desempenho máximo alcançado usando Linpack.

Novembro de 2009

R_{peak}: desempenho de pico teórico.

Valores em Tera Flops.

Cluster? de Computadores

Rank	Site	Computer
1	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIx 2.0GHz, Tofu interconnect Fujitsu
2	National Supercomputing Center in Tianjin China	Tianhe-1A - NUDT TH MPP, X5670 2.93Ghz 6C, NVIDIA GPU, FT-1000 8C NUDT
3	DOE/SC/Oak Ridge National Laboratory United States	Jaguar - Cray XT5-HE Opteron 6-core 2.6 GHz Cray Inc.
4	National Supercomputing Centre in Shenzhen (NSCS) China	Nebulae - Dawning TC3600 Blade, Intel X5650, Nvidia Tesla C2050 GPU Dawning
5	GSIC Center, Tokyo Institute of Technology Japan	TSUBAME 2.0 - HP ProLiant SL390s G7 Xeon 6C X5670, Nvidia GPU, Linux/Windows NEC/HP
6	DOE/NNSA/LANL/SNL United States	Cielo - Cray XE6 8-core 2.4 GHz Cray Inc.
7	NASA/Ames Research Center/NAS United States	Pleiades - SGI Altix ICE 8200EX/8400EX, Xeon HT QC 3.0/Xeon 5570/5670 2.93 Ghz, Infiniband SGI
8	DOE/SC/LBNL/NERSC United States	Hopper - Cray XE6 12-core 2.1 GHz Cray Inc.
9	Commissariat a l'Energie Atomique (CEA) France	Tera-100 - Bull bullx super-node S6010/S6030 Bull SA
10	DOE/NNSA/LANL United States	Roadrunner - BladeCenter QS22/LS21 Cluster, PowerXCell 8i 3.2 Ghz / Opteron DC 1.8 GHz, Voltaire Infiniband IBM

Rmax: desempenho máximo alcançado usando Linpack.

Junho de 2011

Rpeak: desempenho de pico teórico.

Valores em Tera Flops.

Jaguar – Cray XT5-HE Opteron Six Core 2.6 GHz



Cluster? de Computadores

Rank	Site	Computer
1	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu
2	National Supercomputing Center in Tianjin China	NUDT YH MPP, Xeon X5670 6C 2.93 GHz, NVIDIA 2050 NUDT
3	DOE/SC/Oak Ridge National Laboratory United States	Cray XT5-HE Opteron 6-core 2.6 GHz Cray Inc.
4	National Supercomputing Centre in Shenzhen (NSCS) China	Dawning TC3600 Blade System, Xeon X5650 6C 2.66GHz, Infiniband QDR, NVIDIA 2050 Dawning
5	GSIC Center, Tokyo Institute of Technology Japan	HP ProLiant SL390s G7 Xeon 6C X5670, Nvidia GPU, Linux/Windows NEC/HP
6	DOE/NNSA/LANL/SNL United States	Cray XE6, Opteron 6136 8C 2.40GHz, Custom Cray Inc.
7	NASA/Ames Research Center/NAS United States	SGI Altix ICE 8200EX/8400EX, Xeon HT QC 3.0/Xeon 5570/5670 2.93 Ghz, Infiniband SGI
8	DOE/SC/LBNL/NERSC United States	Cray XE6, Opteron 6172 12C 2.10GHz, Custom Cray Inc.
9	Commissariat a l'Energie Atomique (CEA) France	Bull bullx super-node S6010/S6030 Bull
10	DOE/NNSA/LANL United States	BladeCenter QS22/LS21 Cluster, PowerXCell 8i 3.2 Ghz / Opteron DC 1.8 GHz, Voltaire Infiniband IBM

Jaguar?

Rmax: desempenho máximo alcançado usando Linpack.

Novembro de 2011

Rpeak: desempenho de pico teórico.

Valores em Tera Flops.

Cluster? de Computadores

			Cores	Rmax (TF)	Rpeak (TF)	Power (kW)	
73	NCSA United States	Abe - PowerEdge 1955, 2.33 GHz, Infiniband, Windows Server 2008/Red Hat Enterprise Linux 4 / 2007 Dell	9600	68.48	89.59		Cluster
74	University of Southampton United Kingdom	iDataPlex, Xeon E55xx QC 2.26 GHz, Infiniband, Windows HPC2008 R2 / 2009 IBM	8000	66.68	72.32	222.62	Cluster
75	NASA/Ames Research Center/NAS United States	Columbia - SGI Altix 1.5/1.6/1.66 GHz, Voltaire Infiniband / 2008 SGI	13824	66.57	82.94		MPP
76	NACAD/COPPE/UFRJ Brazil	Galileu - Sun Blade x6048, Xeon X5560 2.8 Ghz, Infiniband QDR / 2009 Sun Microsystems	6464	64.63	72.40	430.00	Cluster
77	Barcelona Supercomputing Center Spain	MareNostrum - BladeCenter JS21 Cluster, PPC 970, 2.3 GHz, Myrinet / 2006 IBM	10240	63.83	94.21		Cluster
78	IBM Poughkeepsie Benchmarking Center United States	BladeCenter QS22/LS21 Cluster, PowerXCell 8i 3.2 Ghz / Opteron DC 1.8 GHz, Infiniband / 2008 IBM	7200	63.25	80.93	138.00	Cluster
79	Clemson University United States	Palmetto - PowerEdge 1950/SunFire X2200/iDataPlex Cluster Intel 53xx/54xx 2.33Ghz, Opteron 2.3 Ghz, Myrinet 10G / 2009 Dell/Sun/IBM	8120	60.67	74.70	285.00	Cluster

Única da América do Sul no Top500

Novembro de 2009

Cluster? de Computadores

31	KTH - Royal Institute of Technology Sweden	Lindgren - Cray XT6m 12-Core 2.1 GHz / 2011 Cray Inc.	36384	237.20	305.63	658.35
32	Universitaet Aachen/RWTH Germany	Bullx B500 Cluster, Xeon X56xx 3.06Ghz, QDR Infiniband / 2011 Bull SA	25448	219.84	270.54	
33	Institute of Process Engineering, Chinese Academy of Sciences China	Mole-8.5 - Mole-8.5 Cluster Xeon L5520 2.26 Ghz, nVidia Tesla, Infiniband / 2010 IPE, Nvidia, Tyan	33120	207.30	1138.44	
34	INPE (National Institute for Space Research) Brazil	Tupã - Cray XT6 12-core 2.1 GHz / 2010 Cray Inc.	30720	205.10	258.05	
35	DOE/SC/Oak Ridge National Laboratory United States	Jaguar - Cray XT4 QuadCore 2.1 GHz / 2008 Cray Inc.	30976	205.00	260.20	1580.71
36	Sandia National Laboratories United States	Sandia/Cray Red Storm - Cray XT3/XT4 / 2009 Cray Inc.	38208	204.20	284.00	2506.00
37	NOAA/Oak Ridge National Laboratory United States	Gaea - Cray XT6-HE, Opteron 6100 12C 2.1GHz / 2010 Cray Inc.	30912	194.40	259.66	610.70

Junho de 2011

Cluster? de Computadores

164	Internet Service China	xSeries x3650M3, Xeon X56xx 2.53 GHz, GigE / 2011 IBM	11604	65.39	117.43	351.99
165	IBM Thomas J. Watson Research Center United States	NNSA/SC Blue Gene/Q Prototype 1 / 2010 IBM	8192	65.35	104.86	38.80
166	Leibniz Rechenzentrum Germany	SuperMIG - BladeCenter HX5, WM Xeon 10-core, 2.4 GHz, Infiniband / 2011 IBM	8000	64.86	76.80	195.00
167	NACAD/COPPE/UFRJ Brazil	Galileu - Sun Blade x6048, Xeon X5560 2.8 Ghz, Infiniband QDR / 2009 Oracle	6464	64.63	72.40	430.00
168	Service Provider Japan	xSeries x3650M2 Cluster, Xeon QC E55xx 2.26 Ghz, GigE / 2011 IBM	14432	64.60	130.87	633.20
169	Georgia Institute of Technology United States	Keeneland - HP ProLiant SL390s G7 Xeon 6C X5660 2.8Ghz, nVidia Fermi, Infiniband QDR / 2010 Hewlett-Packard	6048	63.92	188.09	94.40
170	Barcelona Supercomputing Center Spain	MareNostrum - BladeCenter JS21 Cluster, PPC 970, 2.3 GHz, Myrinet / 2006 IBM	10240	63.83	94.21	

Junho de 2011

Cluster? de Computadores

46	Los Alamos National Laboratory United States	Xtreme-X 1320H-LANL, Opteron 12 Core 2.30 GHz, Infiniband QDR / 2011 Appro	37056	230.60	340.92	540.4
47	Universitaet Aachen/RWTH Germany	Bullx B500 Cluster, Xeon X56xx 3.06Ghz, QDR Infiniband / 2011 Bull	25448	219.84	270.54	
48	UCSD/San Diego Supercomputer Center United States	Xtreme-X GreenBlade GB512X, Xeon E5 (Sandy Bridge - EP) 8C 2.60GHz, Infiniband QDR / 2011 Appro	12608	218.10	262.25	252.2
49	INPE (National Institute for Space Research) Brazil	Cray XT6 12-core 2.1 GHz / 2010 Cray Inc.	30720	205.10	258.05	
50	Sandia National Laboratories United States	Cray XT3/XT4 / 2009 Cray Inc.	38208	204.20	284.00	2506.0
51	NOAA/Oak Ridge National Laboratory United States	Cray XT6-HE, Opteron 6100 12C 2.1GHz / 2010 Cray Inc.	30912	194.40	259.66	610.0
52	Japan Atomic Energy Agency (JAEA) Japan	BX900 Xeon X5570 2.93GHz, Infiniband QDR / 2009 Fujitsu	17072	191.40	200.08	831.0

Novembro de 2011

Cluster? de Computadores

287	Gaming Company Ireland	Cluster Platform 3000 BL460c G7, Xeon X5660 6C 2.80 GHz, 10G Ethernet / 2011 HP	8064	64.94	90.32	
288	Defence Australia	BladeCenter HX5, Xeon E7-4870 10C 2.40 GHz, Infiniband QDR / 2011 IBM	9280	64.86	89.09	226.2
289	Leibniz Rechenzentrum Germany	BladeCenter HX5, Xeon E7-4870 10C 2.40 GHz, Infiniband QDR / 2011 IBM	8000	64.86	76.80	195.0
290	NACAD/COPPE/UFRJ Brazil	Sun Blade x6048, Xeon X5560 2.8 Ghz, Infiniband QDR / 2009 Sun	6464	64.63	72.40	430.0
291	Financial Institution () Belgium	iDataPlex, Xeon E55xx QC 2.26 GHz, GigEthernet / 2010 IBM	16704	64.60	151.47	480.2
292	Service Provider Japan	xSeries x3650M2 Cluster, Xeon QC E55xx 2.26 Ghz, GigE / 2011 IBM	14432	64.60	130.87	633.2
293	IT Service Provider (B) United States	BladeCenter HS22 Cluster, Xeon E5540 4C 2.53 GHz, Gigabit Ethernet / 2011 IBM	11400	64.49	115.37	360.4

Novembro de 2011

Cluster de Computadores



Escalabilidade

- O que pode impactar no ganho de desempenho de uma aplicação paralela reduzindo sua escalabilidade?
 - Rede. Por que?
 - Balanceamento de carga. Por que?
 - Regiões sequenciais dos programas. Por que?

Lei de Amdahl

- Uma aplicação pode ter uma grande região de instruções que devem ser executadas sequencialmente.
 - G: ganho de desempenho.
 - T_s : trecho da região sequencial.
 - T_p : trecho da região paralela.
 - n : número de processadores

$$G(n) = \frac{T_s + T_p}{T_s + \frac{T_p}{n}} = \frac{1}{T_s + \left(\frac{1 - T_s}{n}\right)} \leq \frac{1}{T_s}$$

Lei de Amdahl

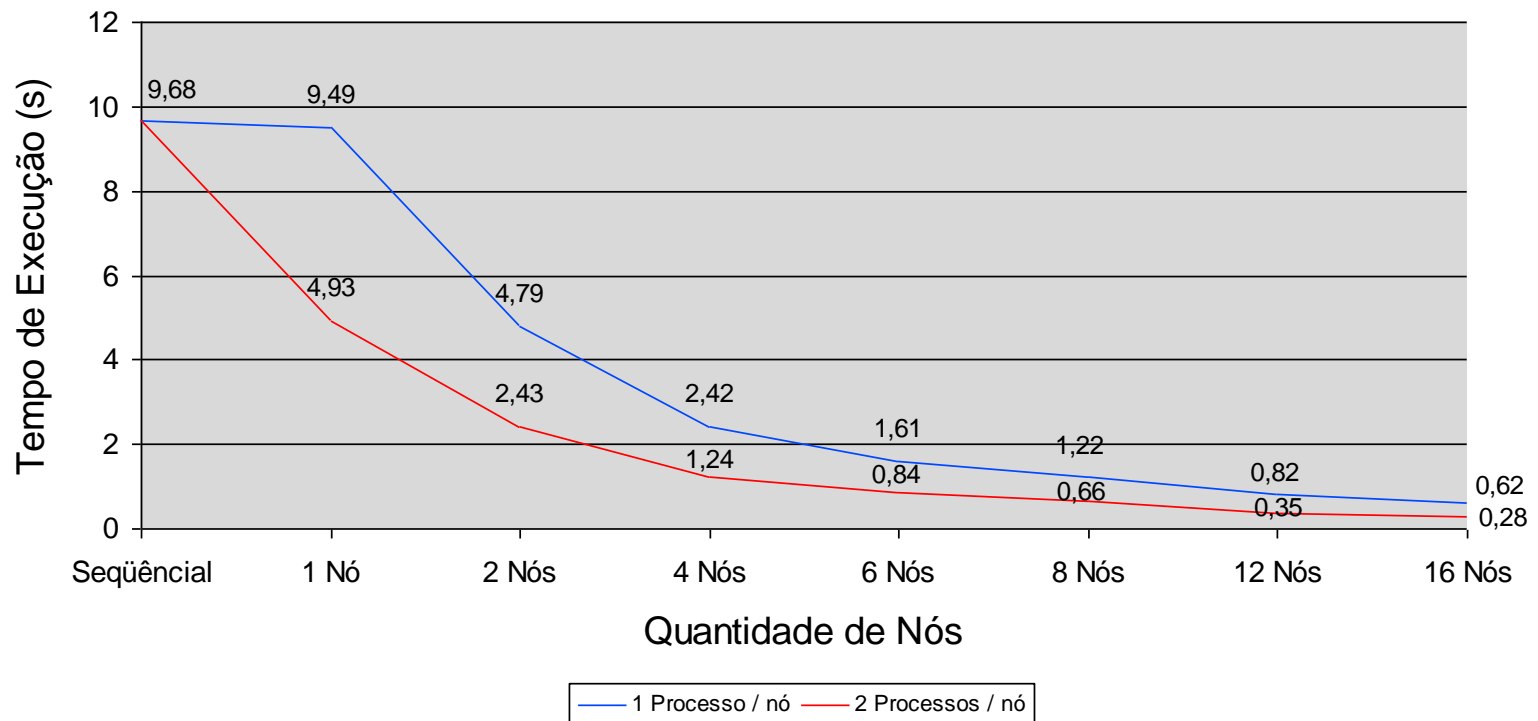
- Um programa possui 60% do código puramente sequencial.
- 40% pode ser paralelizado.
- Dois núcleos serão utilizados.
- Portanto: $1/(0.6 + 0.4/2) = 1/0.8 = 1.25$
- Ganho de desempenho será de 25%.
- Se o número de núcleos tende ao infinito.
- Portanto: $1/0.6 = 1.67$
- Ganho de desempenho de 67% no máximo.

Análise de Eficiência

- Tempo sequencial = 0.05 segundos
- Tempo paralelo = 0.019 segundos
- Quantidade de processadores = 32
- Speedup ideal = 32
- Speedup obtido = (tempo sequencial/tempo paralelo)
 $= 0.05/0.019 = 2.63$
- Eficiência = (Speedup obtido / Qtd processadores) = $0.05 / (0.019 * 32) = 0.0822 * 100\% = 8.22\%$
- Ou seja, o speedup alcançado é cerca de 12.16 vezes menor do que o speedup ideal.
- A eficiência de 8.22% mostra que a solução paralela na verdade é ineficiente.

Avaliação de desempenho

Resultados de Desempenho

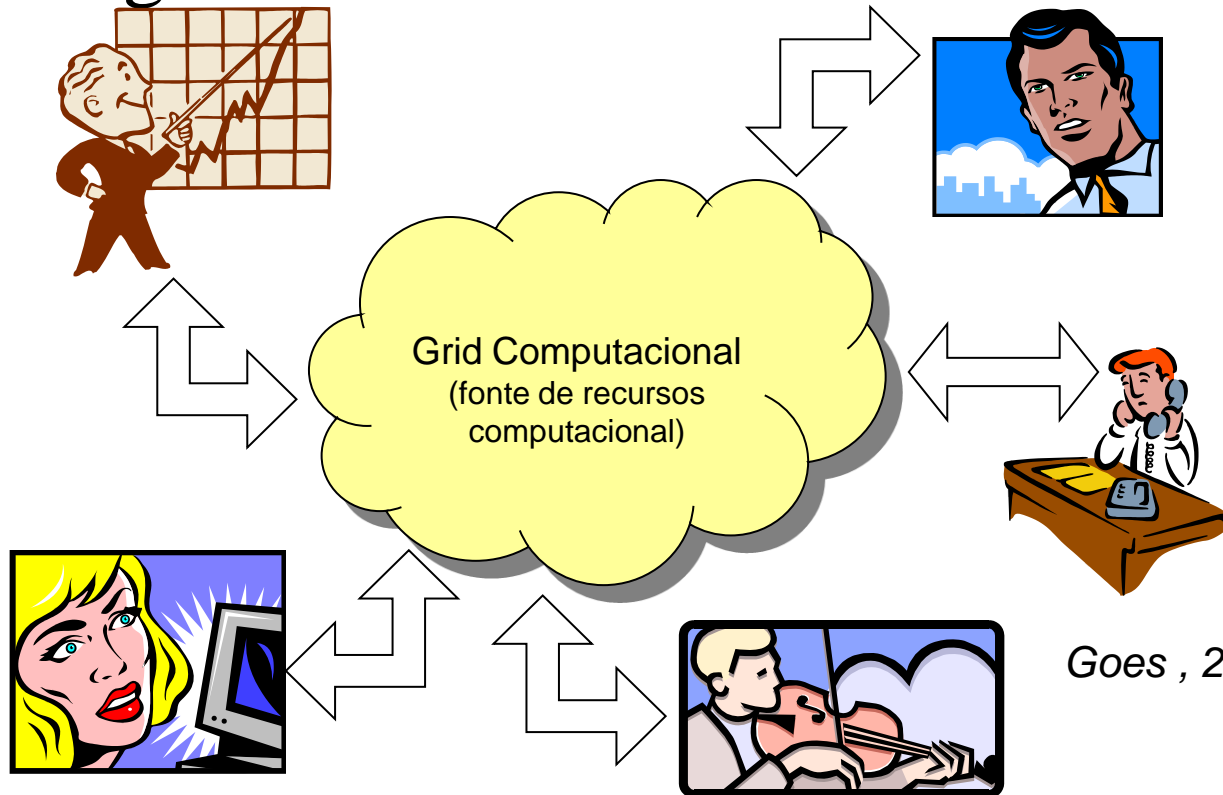


Grid Computacional

- Uma plataforma para execução de aplicações paralelas
 - Amplamente distribuída
 - Heterogênea
 - Compartilhada
 - Sem controle central
 - Com múltiplos domínios administrativos


Grid Computacional

- Analogia com rede elétrica



Goes , 2005

Grid Computacional

- SMPs
 - MPPs
 - NOWs
 - Grids
- 
- acoplamento
- distribuição

- SMP: Symmetric Multiprocessor (memória compartilhada)
- MPP: Massively Parallel Processors
- NOW: Network of Workstations

Grid Computacional

- TeraGrid
 - 4 centros de supercomputação norte-americanos
 - Cada centro com milhares de processadores *dedicados* ao TeraGrid
 - Canais de altíssima velocidade (40 GBits/s)
 - Poder agregado de 13,6 TeraFlops
- SETI@home
 - Ciclos ociosos de 1.6 milhões de processadores espalhados em 224 países
 - Computa em média a uma velocidade de 10 Teraflops
- Grid5000
 - Instrumento científico para estudo de sistemas paralelos e distribuídos de larga escala.
 - O objetivo inicial era alcançar 5000 processadores, atualizado para núcleos e alcançado no inverno de 2008-2009.
 - São 9 sites na França e 1 site no Brasil na cidade de Porto Alegre (UFRGS).
 - 112 núcleos na UFRGS, 14 nós DELL Power Edge 1950, 2 Intel Xeon Quad Core 1.6GHz.

Computação em Grid

- Características das soluções para computação em Grid:
 - Globus: conjunto de serviços que facilitam computação em grade, podendo ser utilizados para submissão e controle de aplicações, descoberta de recursos, movimentação de dados e segurança.
 - Condor: é um sistema que objetiva fornecer grande quantidade de poder computacional a médio e longo prazo utilizando recursos ociosos na rede. Os autores salientam que o Condor objetiva vazão e não desempenho.
 - MyGrid: foca a execução de aplicações paralelas Bag-of-Tasks (tarefas independentes executadas em qualquer ordem). Aplicações Bag-of-Tasks se adequam melhor a heterogeneidade e dinamicidade do grid.

Exploração do Paralelismo através de GPUs

Conceitos

Graphic Processing Units (GPUs)



Graphic Processing Units (GPUs)

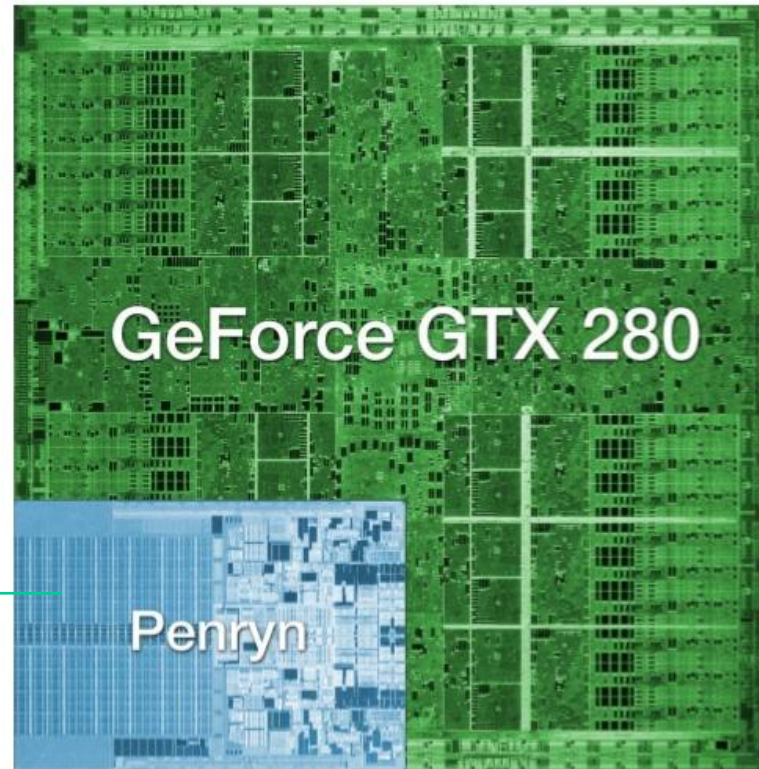
- Aplicações gráficas eram executadas em CPUs.
- Aceleradores gráficos surgiram para operar sobre pixels baseados em pipelines fixos/dedicados.
 - Algoritmos denominados shaders melhoram a qualidade dos gráficos gerados.
 - Shaders: algoritmos de propósito geral que utilizam vértices e pixels como entrada e saída.
- GPUs são processadores gráficos com pipelines programáveis capazes de suportar os shaders.
- Nova terminologia: GPGPU ou General-Purpose GPU, já que os shaders são basicamente algoritmos de propósito geral.
 - GPUs se tornaram uma alternativa para processamento paralelo de propósito geral.

Graphic Processing Units (GPUs)



- Área proporcional dos componentes básicos em um chip de uma CPU comparativamente ao chip de uma GPU (NVIDIA, 2009b).

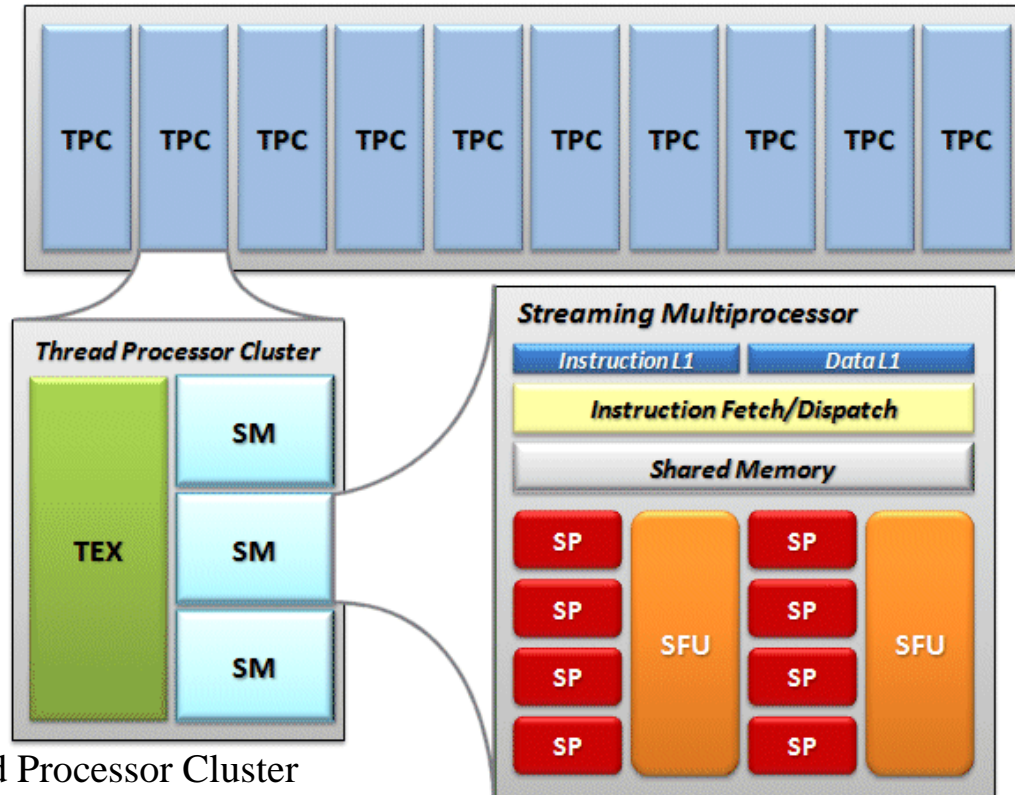
Graphic Processing Units (GPUs)



Codinome para núcleo
que compõe o Core 2 e
o Xeon.

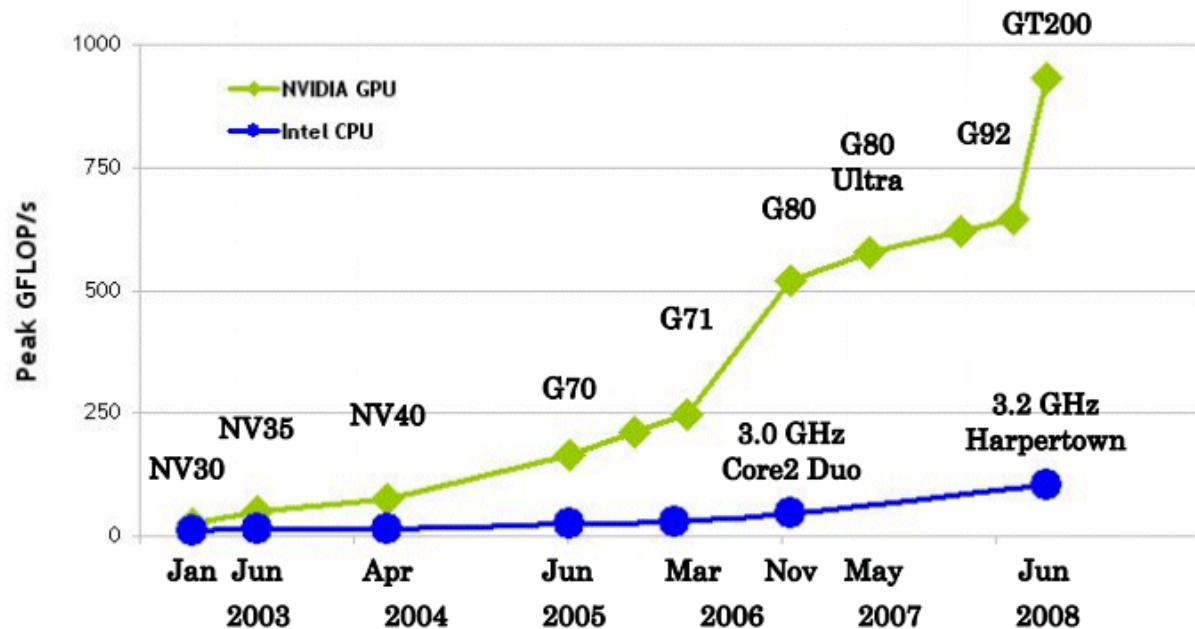
- Área do *chip* de uma GPU (GeForce GTX 280, *chip* GT200) e de uma CPU (Penryn, da Intel)

Graphic Processing Units (GPUs)



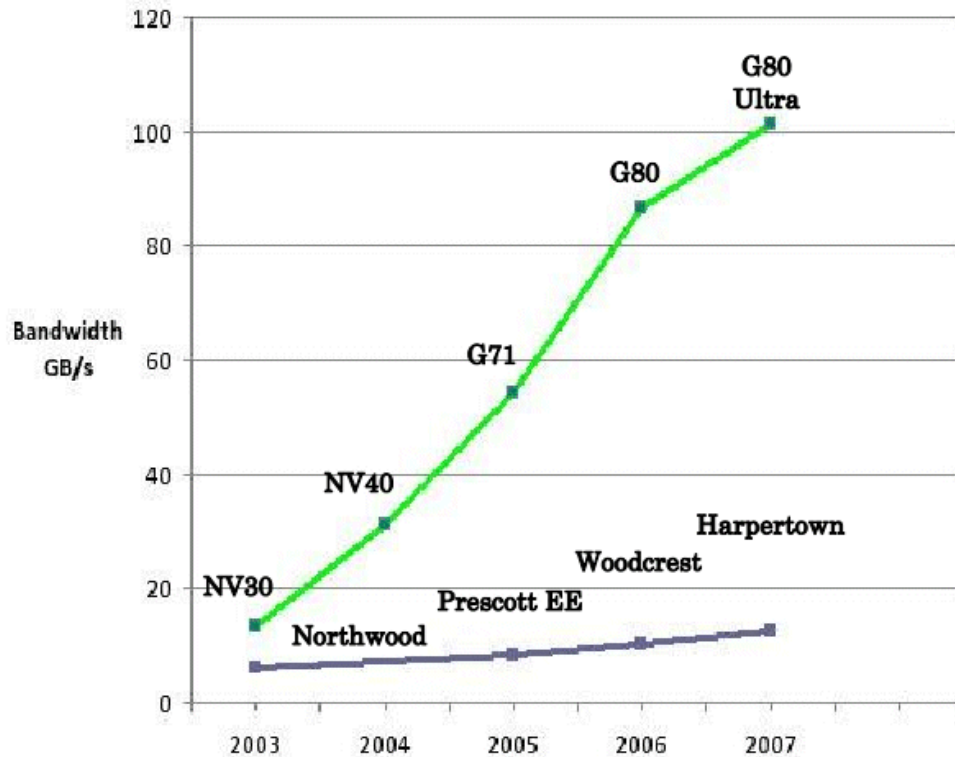
- TPC: Thread Processor Cluster
- SM: Streaming Multiprocessor
- SP: Streaming Processor
- SFU: Super Function Unit
- SIMT: Single Instruction Multiple Thread (semelhante ao SIMD)

Graphic Processing Units (GPUs)



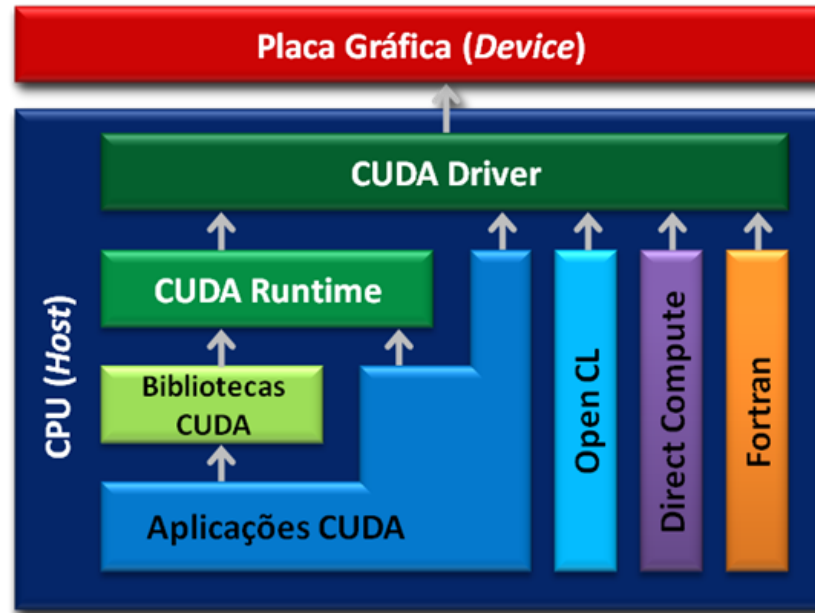
- Evolução da performance aritmética máxima de GPUs e CPUs entre 2003 e 2008 (NVIDIA, 2009)

Graphic Processing Units (GPUs)



- Evolução da largura de banda de GPUs e CPUs, entre 2003 e 2007 (NVIDIA, 2009)

Graphic Processing Units (GPUs)

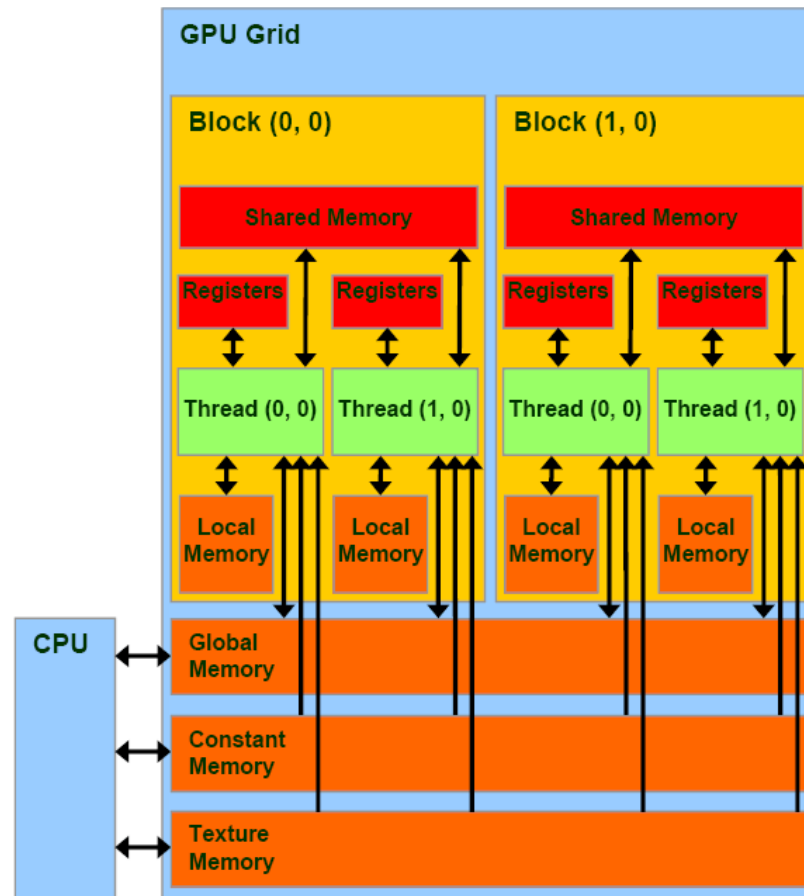


- CUDA: Computer Unified Device Architecture
 - Driver para a primeira camada de abstração do hardware.
 - API CUDA Run time library que executa sobre a API do driver.

Graphic Processing Units (GPUs)

- Um programa em CUDA tem a seguinte sequencia de operações:
 - O host inicializa um vetor com dados.
 - O array é copiado da memória do host para a memória do dispositivo CUDA.
 - O dispositivo CUDA opera sobre o vetor de dados.
 - O array é copiado de volta para o host.

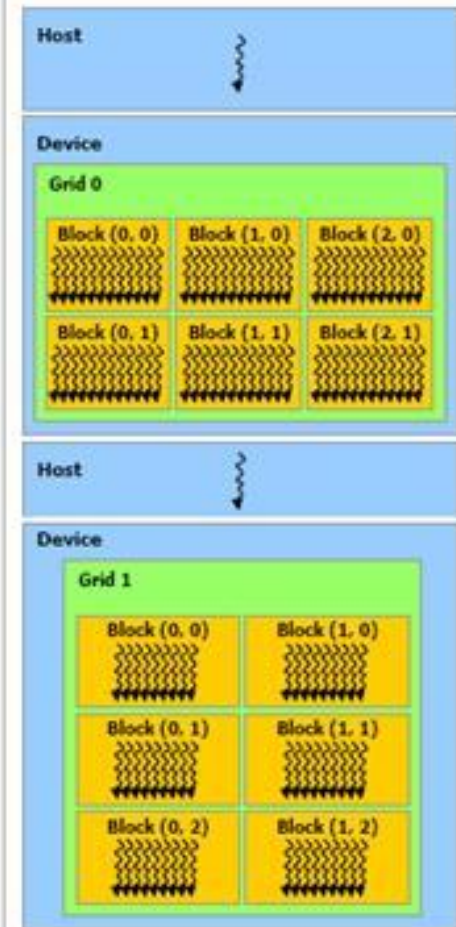
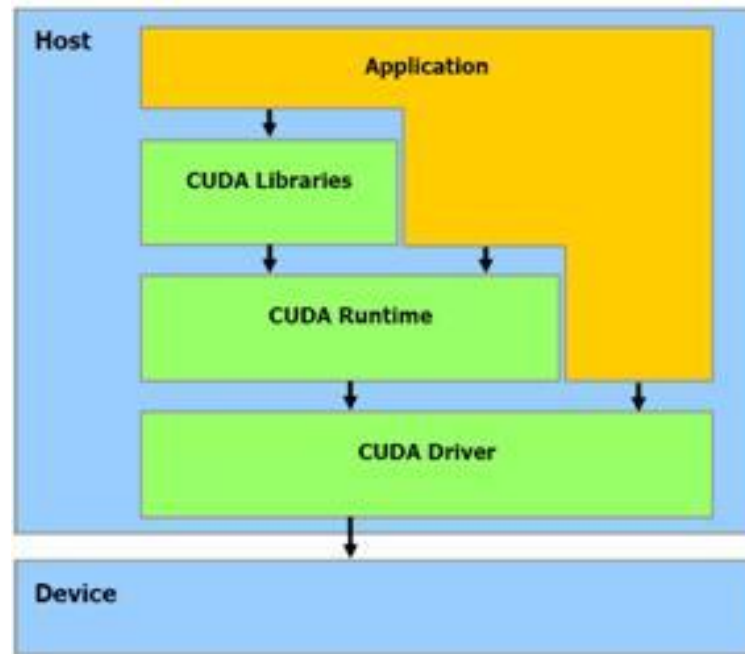
Graphic Processing Units (GPUs)



- Hierarquia de Memória CUDA (NVIDIA, 2009)

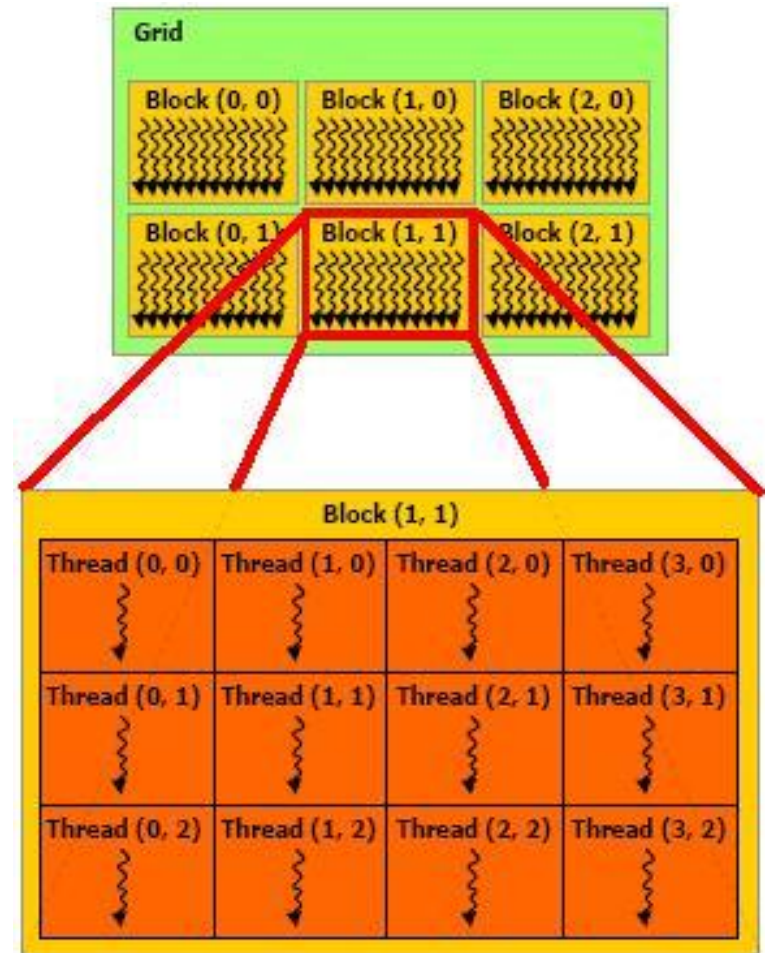
Graphic Processing Units (GPUs)

- Bloco: unidade de organização das threads.
- Grid: Unidade básica onde estão distribuídos os blocos.



Graphic Processing Units (GPUs)

- Estrutura de um programa:
 - Definir funções em linguagem C (kernels)
 - Executados N vezes por N threads em paralelo
 - SIMT



Graphic Processing Units (GPUs)

- A API CUDA introduz 4 extensões à linguagem C:
 - Qualificadores do tipo de função (lógica de execução, GPU ou CPU).
 - Qualificadores do tipo de variável (onde está armazenada, GPU ou CPU).
 - Uma nova sintaxe de chamada de função para configurar blocos e grid.
 - Quatro variáveis internas para acessar índices e dimensões dos blocos, grid e threads.

Graphic Processing Units (GPUs)

- Qualificadores de tipo de função:
 - `__device__`: define função que será executada na GPU.
 - `__global__`: define o que a plataforma CUDA chama de kernel, ou seja, a função chamada a partir da CPU que será executada na GPU.
 - `__host__`: define função que será executada na CPU e só pode ser chamada a partir da CPU.

Graphic Processing Units (GPUs)

- Qualificadores de tipo de variável:
 - `__device__`: define variável que reside na memória global da GPU. Acessíveis por todas as threads de um grid e pela CPU.
 - `__constant__`: a diferença está no fato de que a variável reside na memória constante da GPU.
 - `__shared__`: variáveis que residem na memória compartilhada da GPU, são acessíveis apenas pelas threads de um mesmo bloco. Tempo de vida igual ao tempo de vida do bloco.

Graphic Processing Units (GPUs)

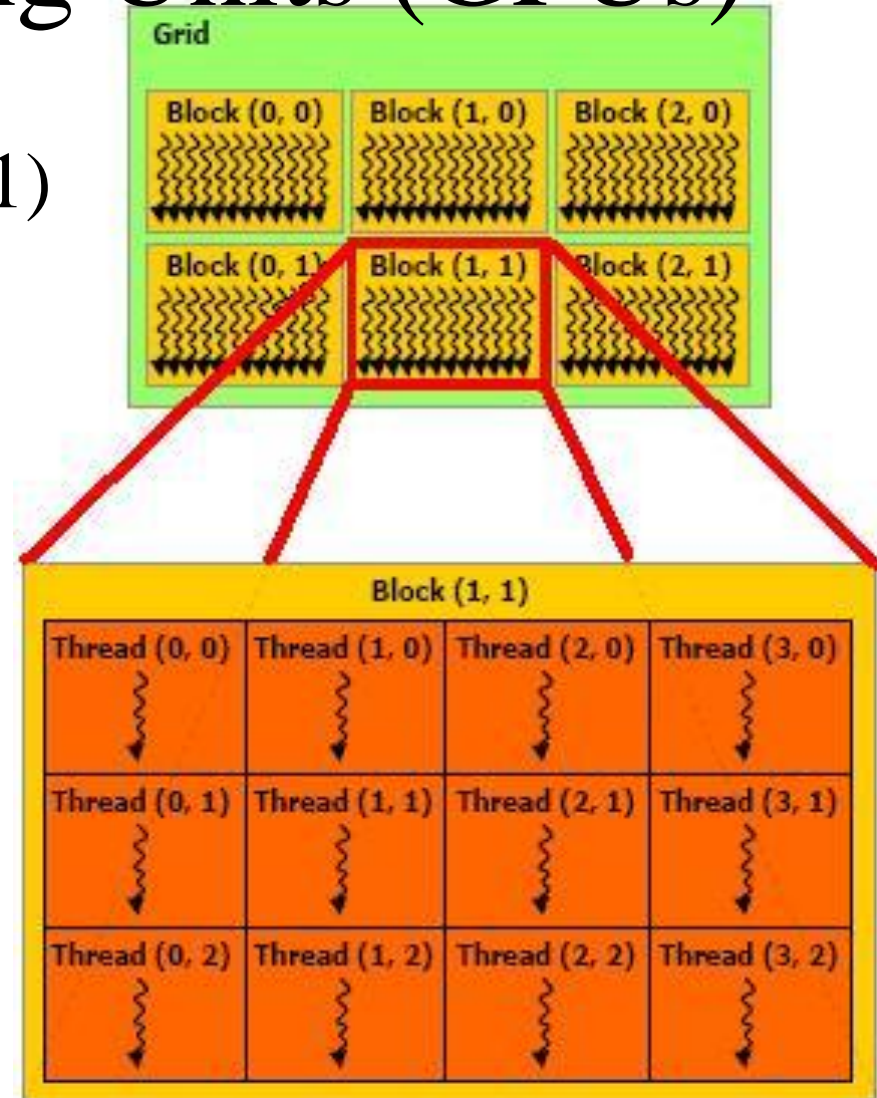
- Sintaxe da chamada de função:
 - Informar no código as dimensões do grid e do bloco.
 - Entre o nome da função e os argumentos usar um array bidimensional onde constam as dimensões do grid e do bloco.
 - Delimitado pelos caracteres <<< e >>>.

Graphic Processing Units (GPUs)

- Variáveis auxiliares:
 - Acesso aos índices das threads por dimensões: threadIdx.x, threadIdx.y, threadIdx.z
 - Acesso ao identificador de cada bloco dentro do grid: blockIdx.x, blockIdx.y
 - Acesso aos índices dos blocos de threads: blockDim.x, blockDim.y, blockDim.z
 - Acesso aos valores de dimensões de grids: gridDim.x, gridDim.y

Graphic Processing Units (GPUs)

- Thread (1,2) do bloco (1,1)
 - threadIdx.x = 1
 - threadIdx.y = 2
 - blockIdx.x = 1
 - blockIdx.y = 1
 - blockDim.x = 4
 - blockDim.y = 3
 - gridDim.x = 3
 - gridDim.y = 2



Graphic Processing Units (GPUs)

- Produto escalar em CUDA (VASCONCELOS, 2009)

- 1. int main(){
- 2.
- 3. cudaMalloc((void**) &d_A, mem_size);
- 4. cudaMalloc((void**) &dev, sizeof(unsigned int));
- 5.
- 6. cudaMemcpy(d_A, h_A, mem_size, cudaMemcpyHostToDevice);
- 7. cudaMemcpy(d_B, h_B, mem_size, cudaMemcpyHostToDevice);
- 8.
- 9. mult<<<blocos, threads>>>(d_C, d_A, d_B);
- 10. soma<<<blocos, threads>>>(d_C, d_Result);
- 11.
- 12. cudaMemcpy(host, dev, sizeof(unsigned int), cudaMemcpyDeviceToHost);
- 13. cudaFree(d_A);
- 14. }
- 15.

Alocação de mem. na GPU

Cópia p/ mem. da GPU

Cópia dados p/ CPU
Liberar memória.

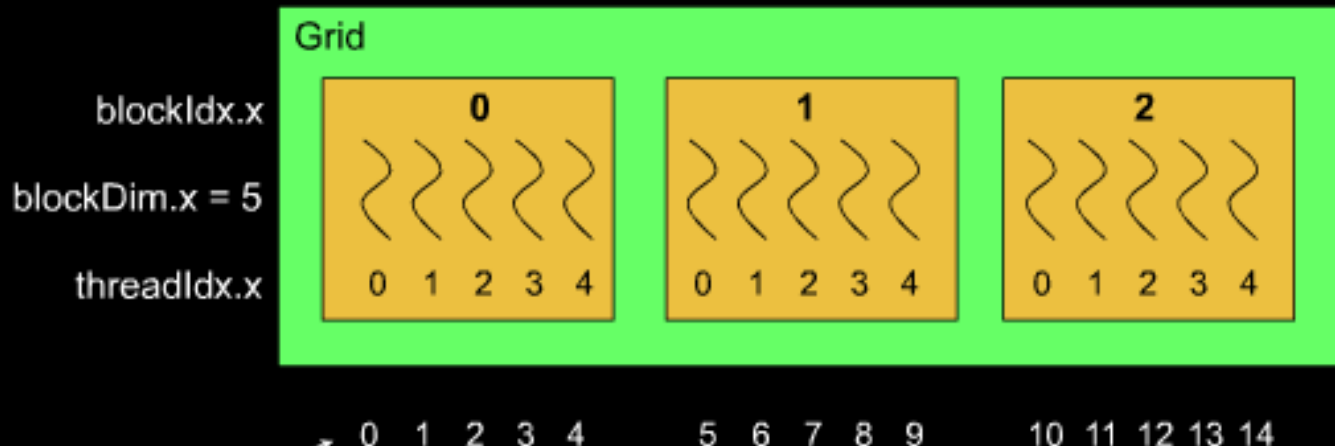
- 16. __global__ void mult(unsigned int* C, unsigned int* A, unsigned int* B) {
- 17. int i = blockDim.x * blockIdx.x + threadIdx.x;
- 18. C[i] = A[i] * B[i];
- 19. }
- 20.
- 21. __global__ void soma(unsigned int* C, unsigned int* total) {
- 22. unsigned int tid = threadIdx.x;
- 23. unsigned int i = blockDim.x * blockIdx.x + threadIdx.x;
- 24.
- 25. __shared__ unsigned int x[VECTOR_LENGTH/THREAD_BLOCKS];
- 26. x[tid] = C[i];
- 27. __syncthreads();
- 28.
- 29. for(int s=blockDim.x/2; s>0; s=s/2) {
- 30. if(tid < s) x[tid] += x[tid + s];
- 31. __syncthreads();
- 32. }
- 33.
- 34. if (tid == 0) atomicAdd(total, x[0]);
- 35. }

Graphic Processing Units (GPUs)

Unique Thread IDs

Basics

- Built-in variables are used to determine unique thread IDs
 - Map from local thread ID (threadIdx) to a global ID which can be used as array indices



$$\text{blockIdx.x} * \text{blockDim.x} + \text{threadIdx.x}$$