T-20
2/4 de Junho de 2010

Sumário:

Multiprocessadores:

- Classificação de Flynn
- Arquitecturas SIMD (Single Instruction Multiple Data)
  - Instruções vectoriais
- Multiprocessadores de memória partilhada
  - *Simultaneous multithreading*, múltiplos *cores*
  - UMA (Uniform memory access time) e NUMA (Non-Uniform memory access time)
- Multiprocessadores de memória distribuída
- Computação usando GPUs (Graphical Processing Units)

Slides preparados pelo Prof. Hervé Paulino para a edição da cadeira de 2008/2009

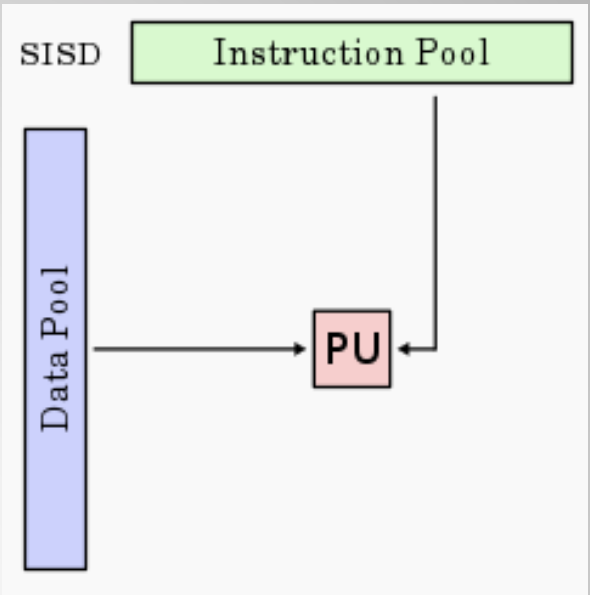
Taxonomia de Flynn (1966)

- Procura classificar todas as arquitecturas de computadores com base no processamento das instruções e dos dado

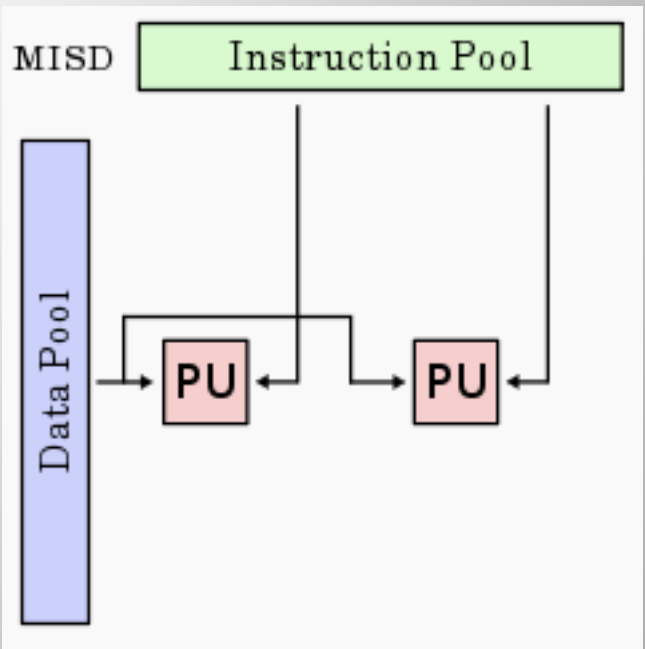
	Uma instrução	**Múltiplas instruções**
Um conjunto de dados	SISD (Von Neumann)	MISD (não interessante)
Múltiplos conjuntos de dados	SIMD	MIMD

SISD e MISD

- SISD - Single Instruction Single Data
  - Von Neumann

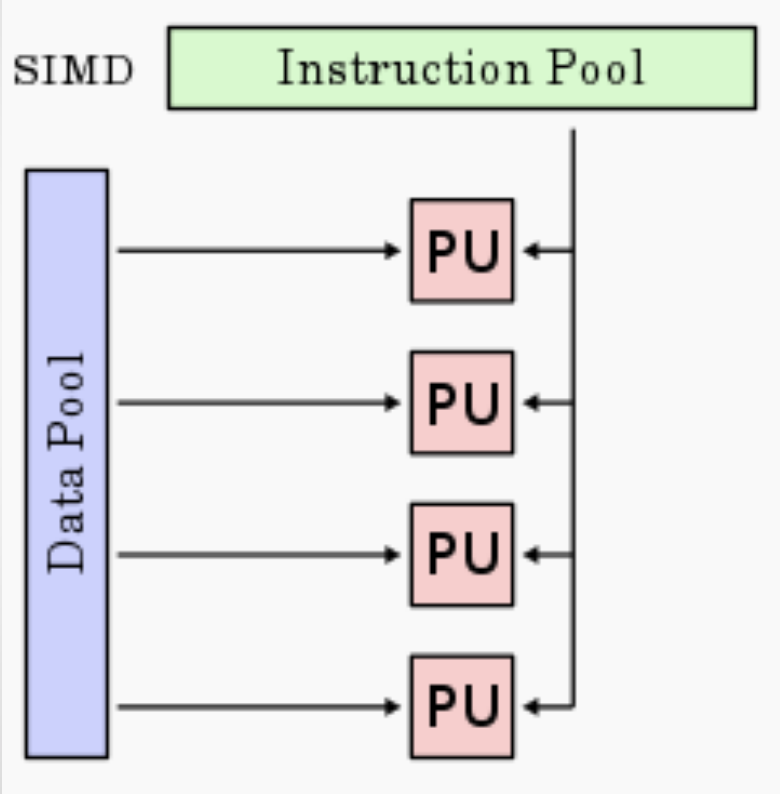


- MISD - Multiple Instruction Single Data
  - Não considerada



SIMD

- SIMD - Single Instruction Multiple Data
  - Uma instrução opera sobre múltiplos dados produzindo múltiplos resultados
  - Por oposição do tradicional: SISD - Single Instruction Single Data
    - Uma instrução opera sobre um único conjunto de dados produzindo um resultado



SIMD - Instrucções vectoriais

- Tradicional: operações sobre escalares
  - Exemplo: R3 ← R1 + R2

    Uma soma de dois escalares para obter um terceiro

- Vectorial: operações sobre grupos de escalares
  - Exemplo: V3 ← V1 + V2

    Uma soma de todos os elementos no vector V1 com os de V2 para obter um terceiro vector

  - Equivale a:

```
for ( i=0; i<N; i++ )
  V3[i] = V1[i] + V2[i];
```

Comparação (em pseudo-assembly)

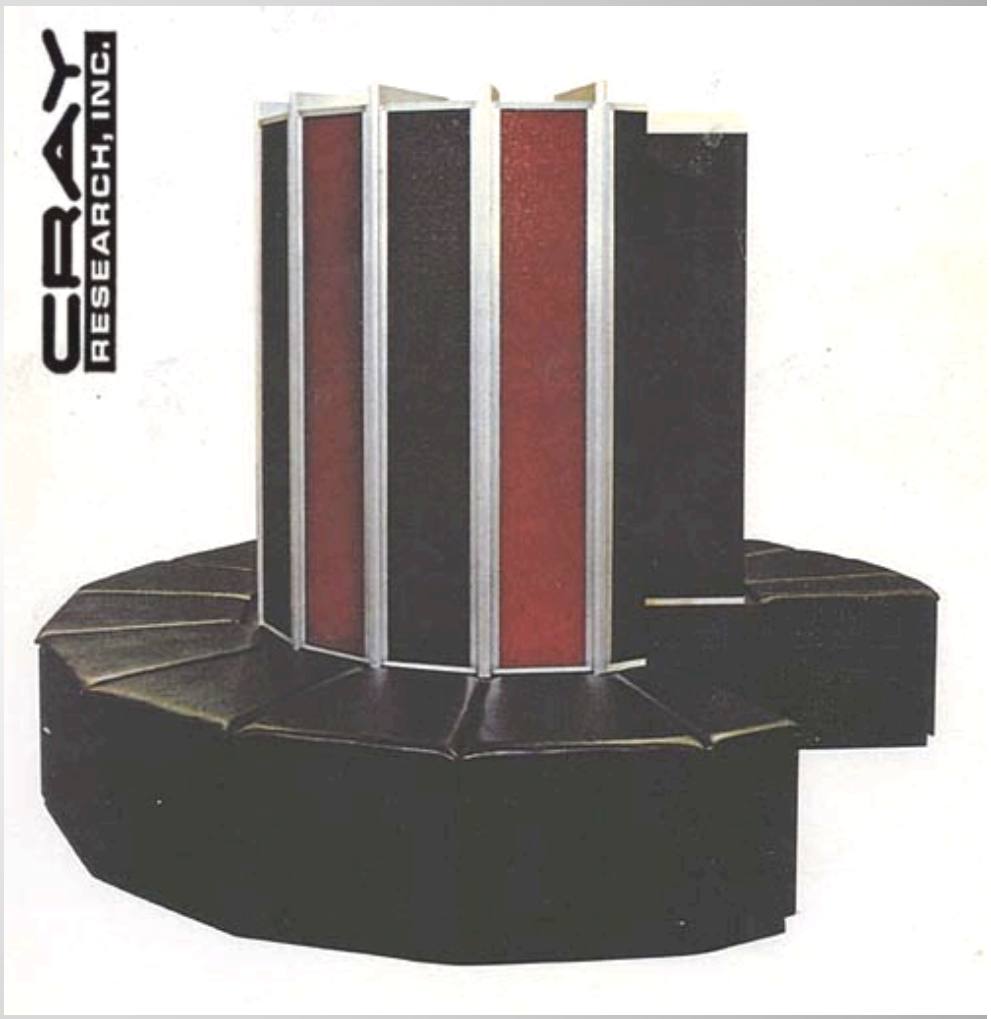
```
   mov R1, N
aloop:
   mov R2, [V1+R1]
   add R2, [V2+R1]
   mov [V3+R1], R2
   sub R1, 1
   jnz aloop
```

```
   ldv R1, [V1]
   ldv R2, [V2]
   addv R2, R1
   stv [V3], R2
```

- Menos instruções → menos fetchs
- Podem ser reduzidos (ou mesmo eliminados) os ciclos e existem menos dependências de dados

Exemplo comercial nos anos 70

- CRAY – 1 (1976)
  - 80 MHz
  - 8 registos de 64 bits
  - 8 registos vectoriais de 64 elementos de 64 bits
  - 6 unidades de execução



Vantagens/Desvantagens

- Vantagens
  - Menos instruções no programa
  - Menos fetch/decode
  - Paralelismo sobre grupos de valores
  - Menos acessos à memória
  - Menos problemas no aproveitamento do pipelining
- Desvantagens
  - Unidades de execução paralelas e dedicadas
  - Que código é realmente “vectorizável”?

Código vectorizável

- Muitos algoritmos dependem de vectores ou matrizes
  - Podem facilmente ser implementados usando instruções vectoriais
- Muitos algoritmos dependem da aplicação das mesmas operações a grupos de valores
  - Agrupamos estes valores em vectores

Hoje em dia

- Instruções vectoriais são incorporadas nas arquitecturas tradicionais
  - Intel IA-32 tem as extensões:
    - MMX (**M**ulti**M**edia e**X**tension), SSE (**S**treaming **S**IMD **E**xtensions), SSE2, SSE3, SSE4, AVX , …
  - AMD, além das extensões da Intel:
    - 3DNow!, SSE5
  - IBM/Freescale PowerPC tem:
    - Altivec

Intel SSE

- 8 registos de 128bits (XMM0 – XMM7)
  - Vistos como contendo: 2x64, 4x32, 8x16 ou 16x8
- Suporta load/store destes grupos de valores
- Suporta operações aritméticas (inteiros ou reais) e lógicas sobre estes registos:

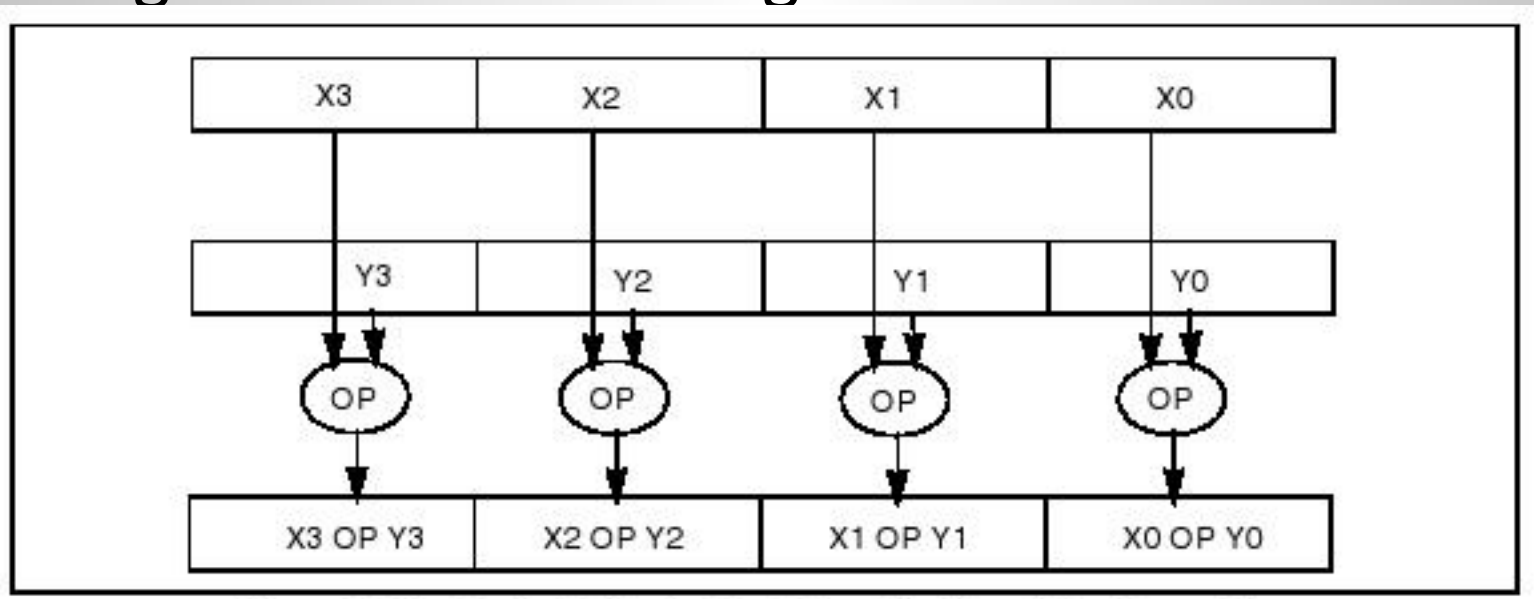
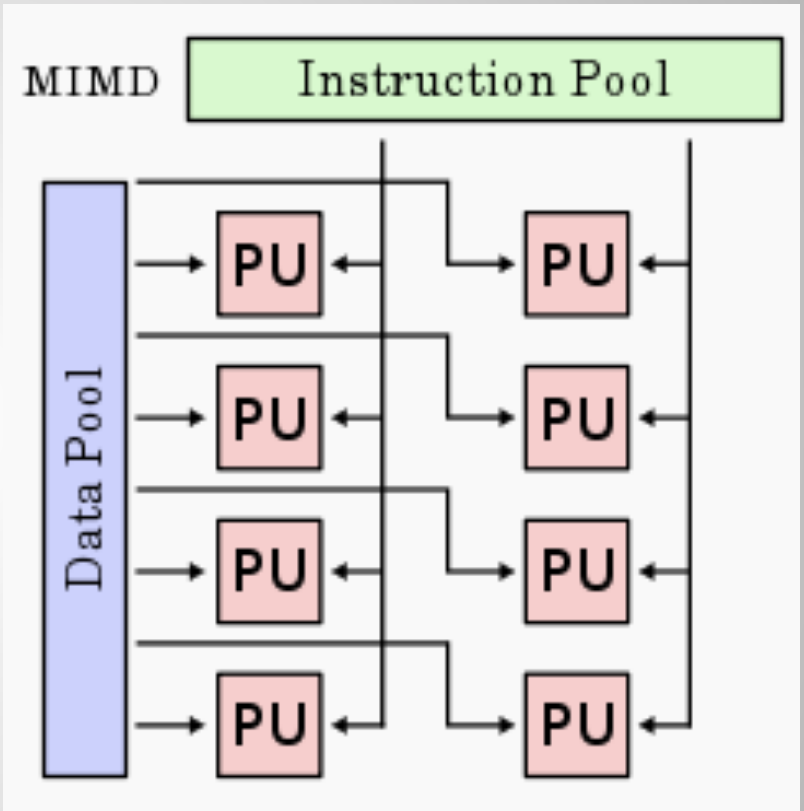


Figure 10-5. Packed Single-Precision Floating-Point Operation

MIMD

- MIMD - Multiple Instruction Multiple Data
  - Múltiplas instruções operam sobre múltiplos dados e produzem múltiplos resultados
  - Sistemas paralelos/ distribuídos
    - Muitas classes destes sistemas …

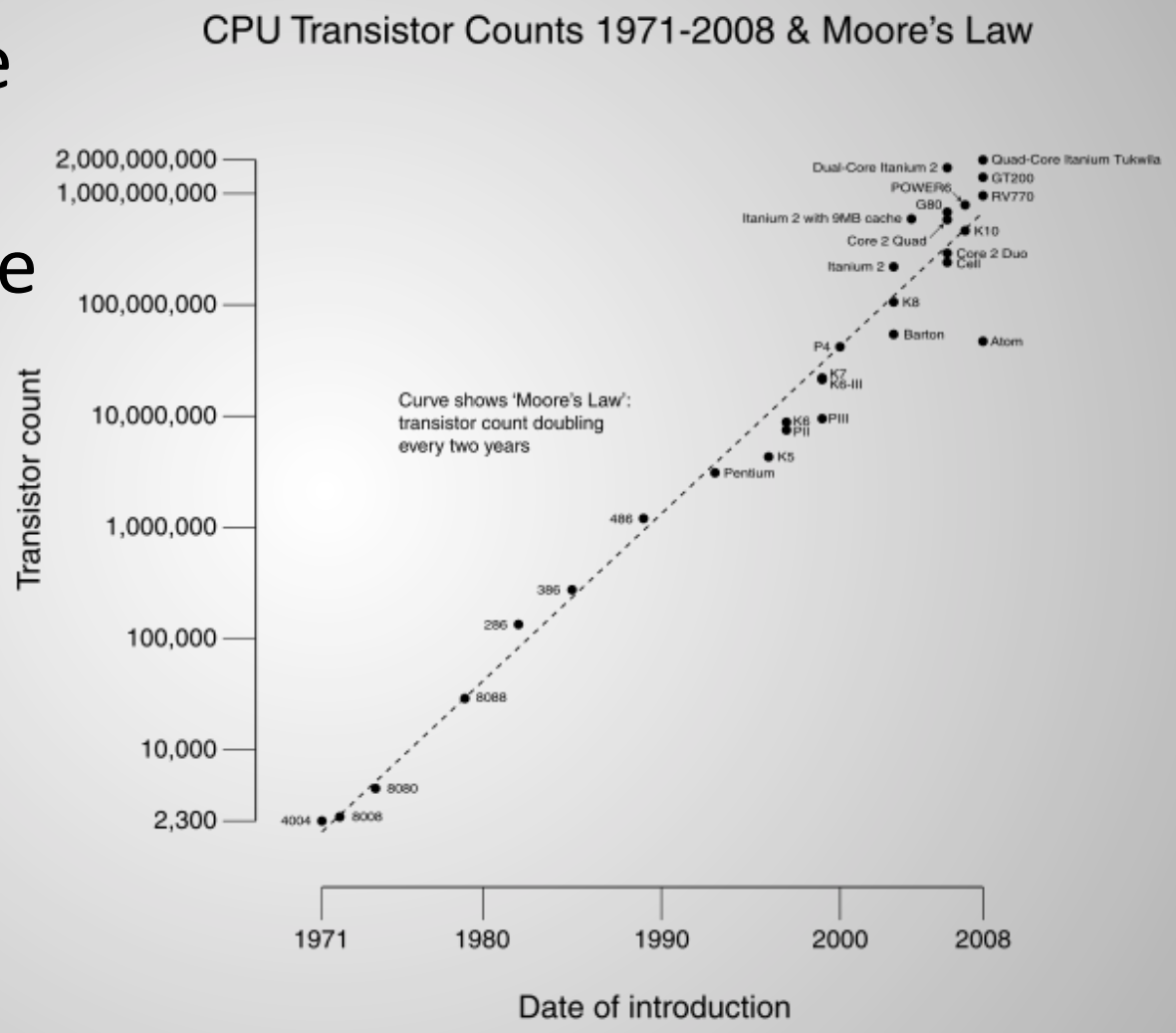


Tipos de arquitecturas MIMD

- Memória partilhada
  - Todas as unidades processadoras partilham a mesma memória
  - Exemplos:
    - Hyper-threading
    - Multicores
    - SMP (Symmetric Multi-processors)
- Memória distribuída
  - Sem memória partilhada: cada unidade processadora tem a sua própria memória
  - Exemplos:
    - MPP (Massively Parallel Processors)
    - DS (Distributed Systems)

Lei de Moore

- Lei de Moore será válida ainda durante algumas gerações



Velocidade de Relógio vs Paralelismo

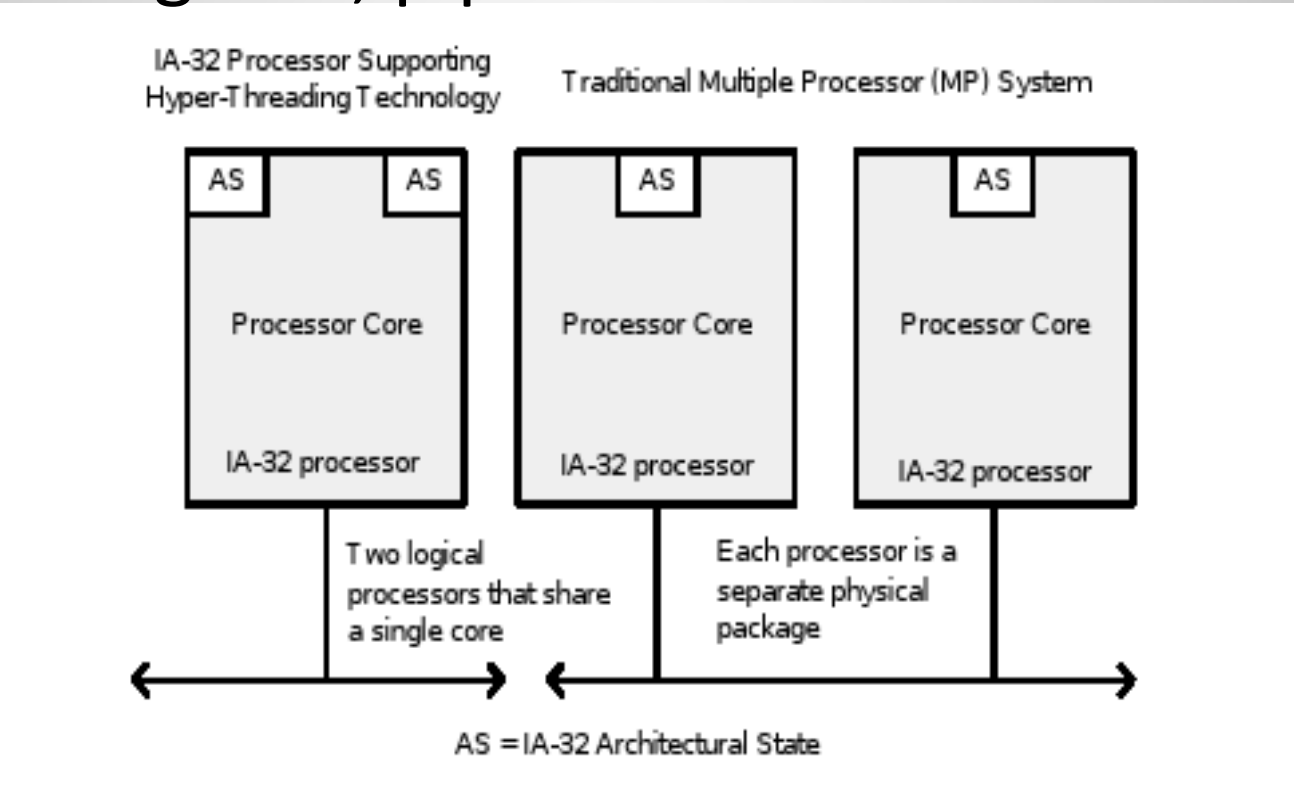
- Como tirar partido das consequências da Lei de Moore para aumentar a performance dos CPUs

- Até à pouco tempo
  - Aumento da velocidade de relógio
    - Precisa de muita energia → maior consumo
    - Gera calor → requer unidades de dissipação

- Actualmente
  - Paralelismo
  - Mais memória cache on-chip

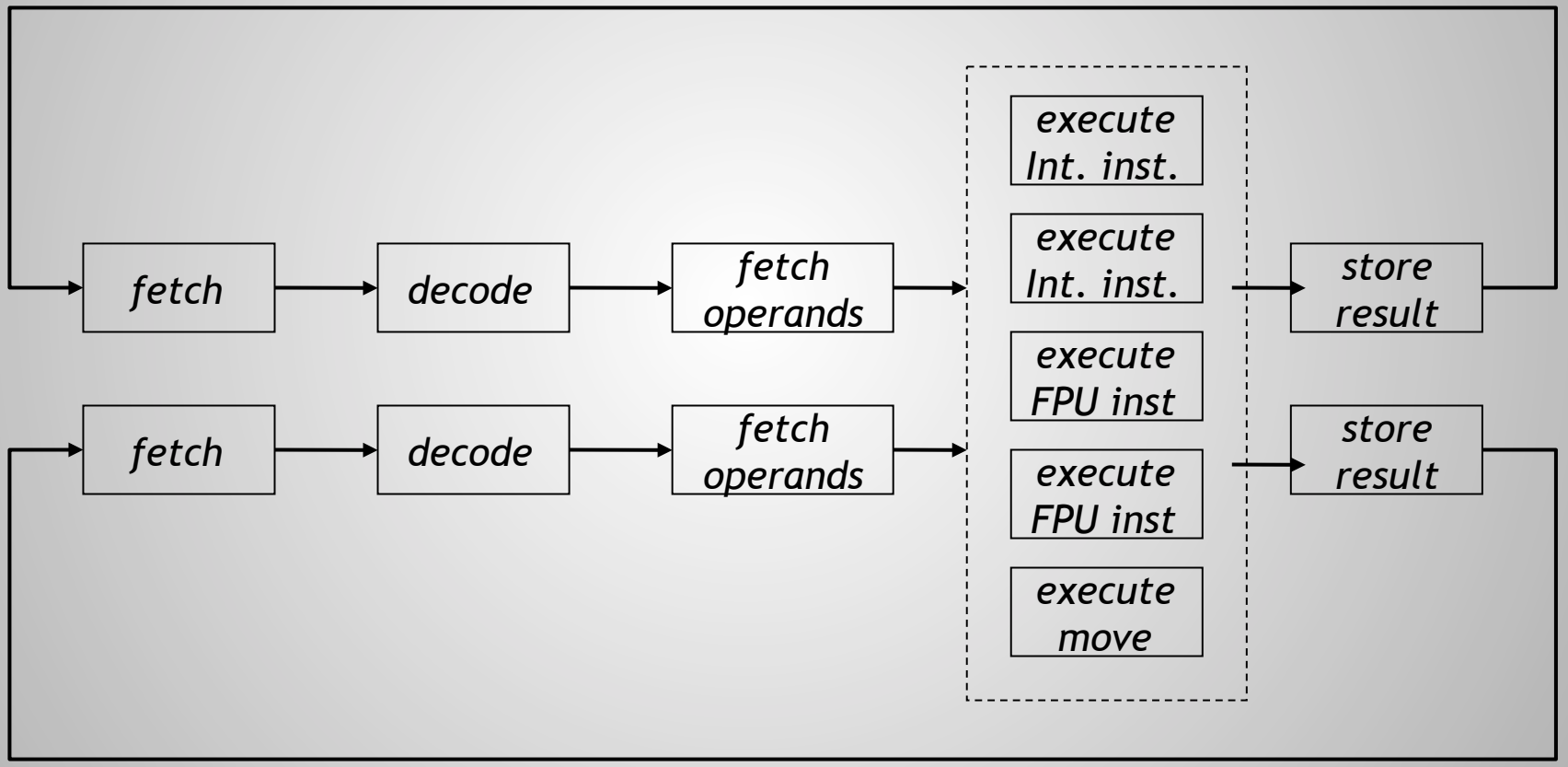
Hyper-threading

- Tecnologia usada nos processadores Intel para "oferecerem" 2 processadores lógicos
- Duplica registos, pipelines e PIC



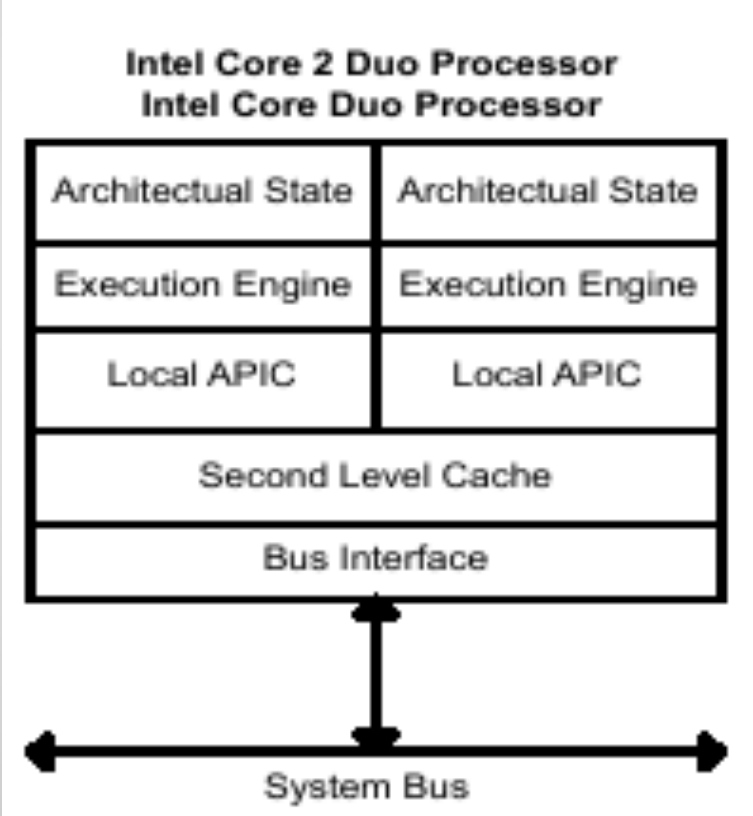
Hyper-threading

- Vários pipelines com vários unidades de execução



Multi-cores

- Tecnologia que permite a coexistência que vários núcleos ("core") de execução num só pacote

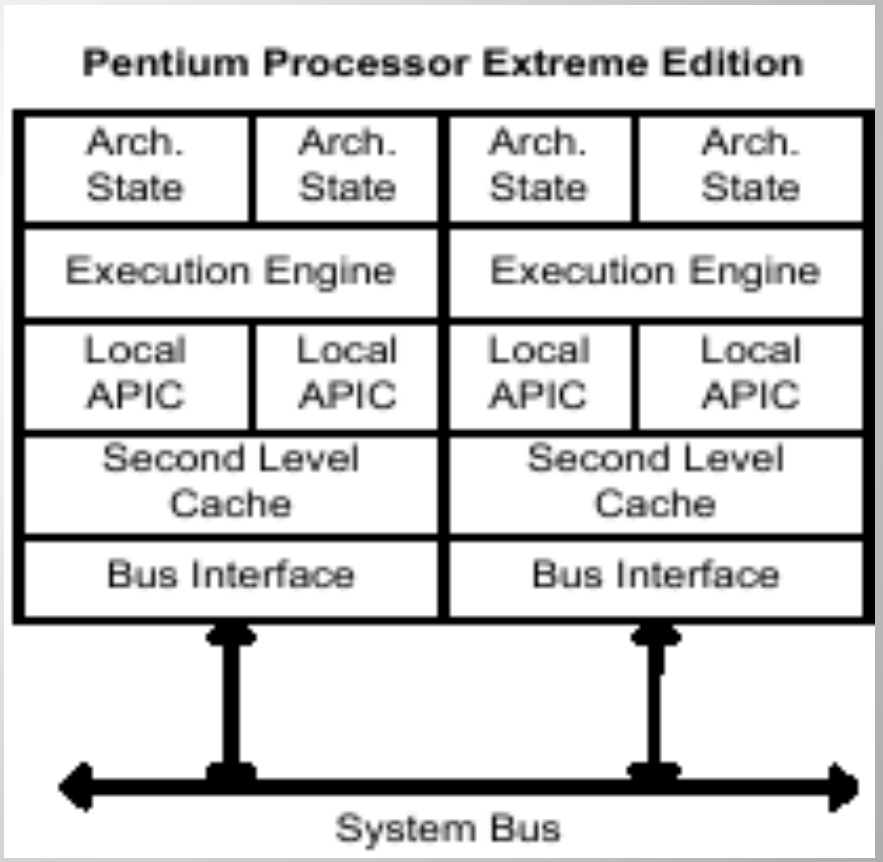


Multi-cores

- Intel
  - Pentium D – primeiro processador com 2 cores
  - Pentium Extreme Edition – 2 cores com hyper-threading
  - Core Duo – 2 cores
  - Core 2 Duo – 2 cores
  - Code 2 Quad – 4 cores
  - I7 – 4 cores com hyper-threading

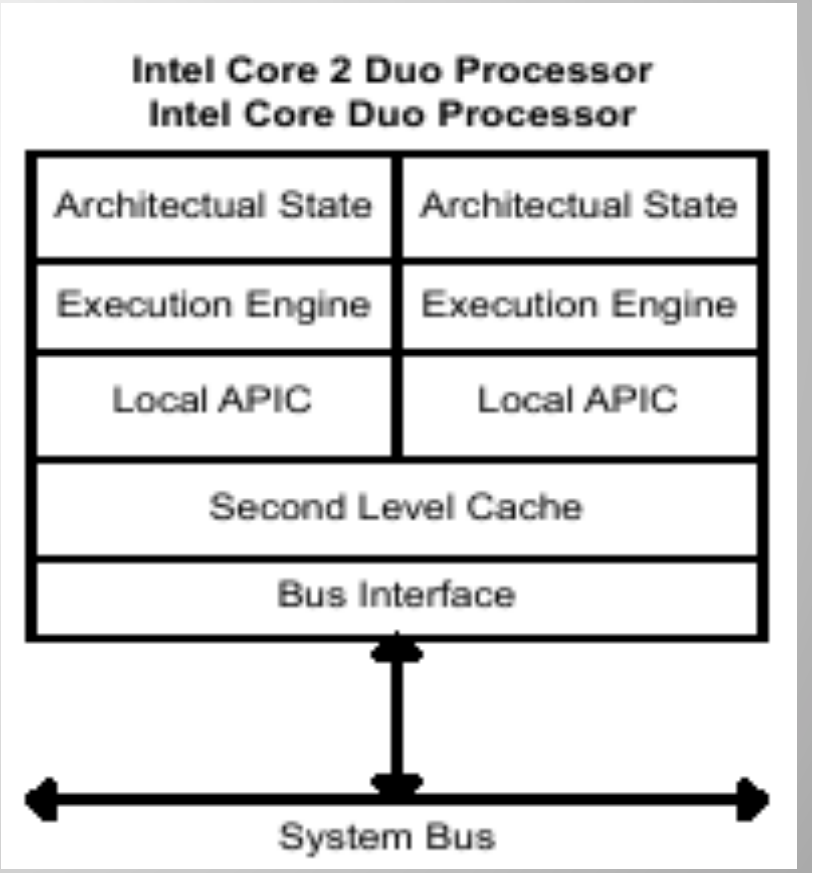
Intel Pentium Extreme Edition

- 2 cores com hyper-threading
  - Cada core tem caches L1 (dados e instruções) e L2 dedicadas



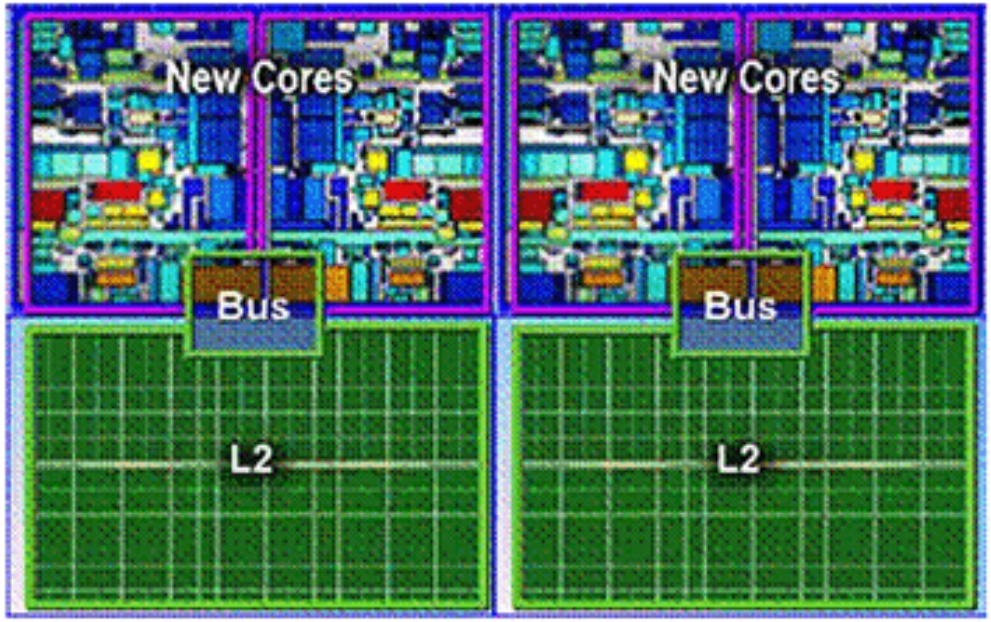
Intel Core 2 Duo

- 2 cores
  - Cada core tem duas caches L1
    - Dados – 32 Kbytes
    - Instruções – 32 Kbytes
  - A cache L2 é partilhada
    - 2 a 4 Mbytes
  - Caches L1 e L2 write-back e non- inclusive



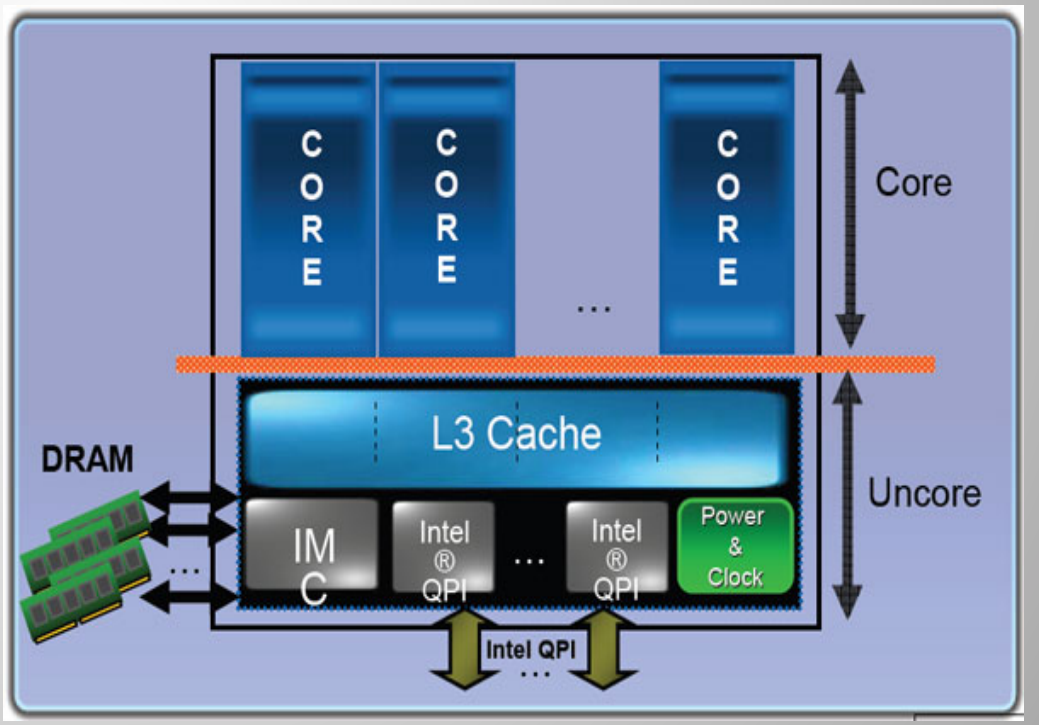
Multi-cores

- Intel Code 2 Quad – 4 cores
  - Na realidade são 2 Core 2 Duo num só pacote
  - Cada core tem 1 caches L1 (dados e instruções) dedicadas
  - Cada 2 cores partilha



Intel I7

- Intel I7 - 4 cores com hyper-threading
  - 8 processadores virtuais
  - Cada core tem caches L1 e L2 dedicadas
    - L1 dados e instruções – cada com 64 Kbytes
    - L2 - 256 Kbytes
    - Caches L1 e L2 write-back e non-inclusive
  - Caches L3 partilhada write-back e inclusive
    - Os dados que estão em L1 ou L2 têm de estar em L3



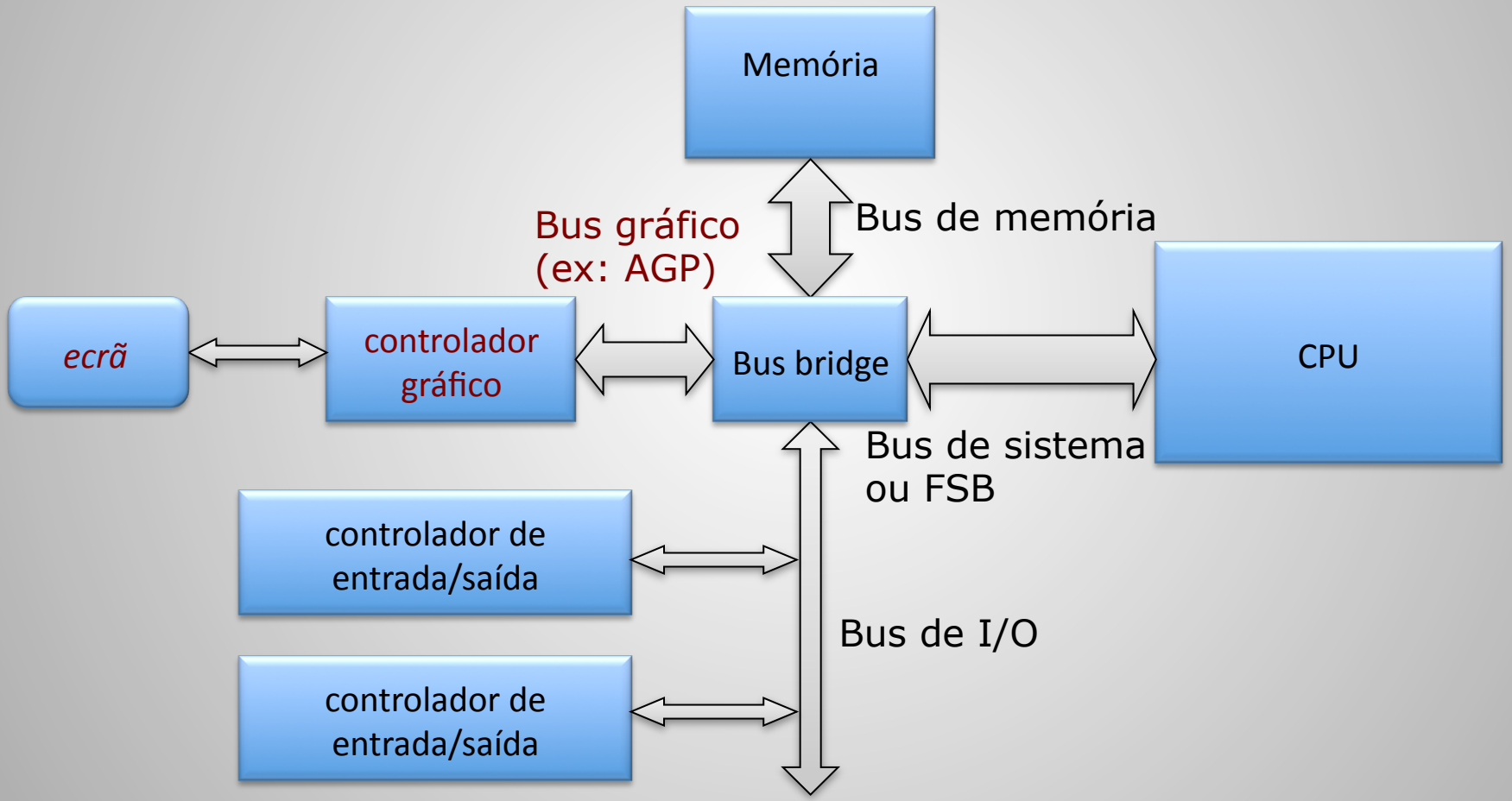
Transferências CPU - Memória video

- Supondo uma resolução 1024x768, com 3 bytes de cor por pixel:
  - Memória vídeo = 3x1024x768 = 2,25 Mbytes
  - Para apresentar uma imagem completamente nova no ecrã é necessário transferir 2,25 MBytes
  - Se forem 10 imagens/s (10 frames/s) = 22,5 MB/s

- Grandes taxas de transferência, muita ocupação dos buses e do CPU
  - O controlador gráfico é “promovido” na arquitectura

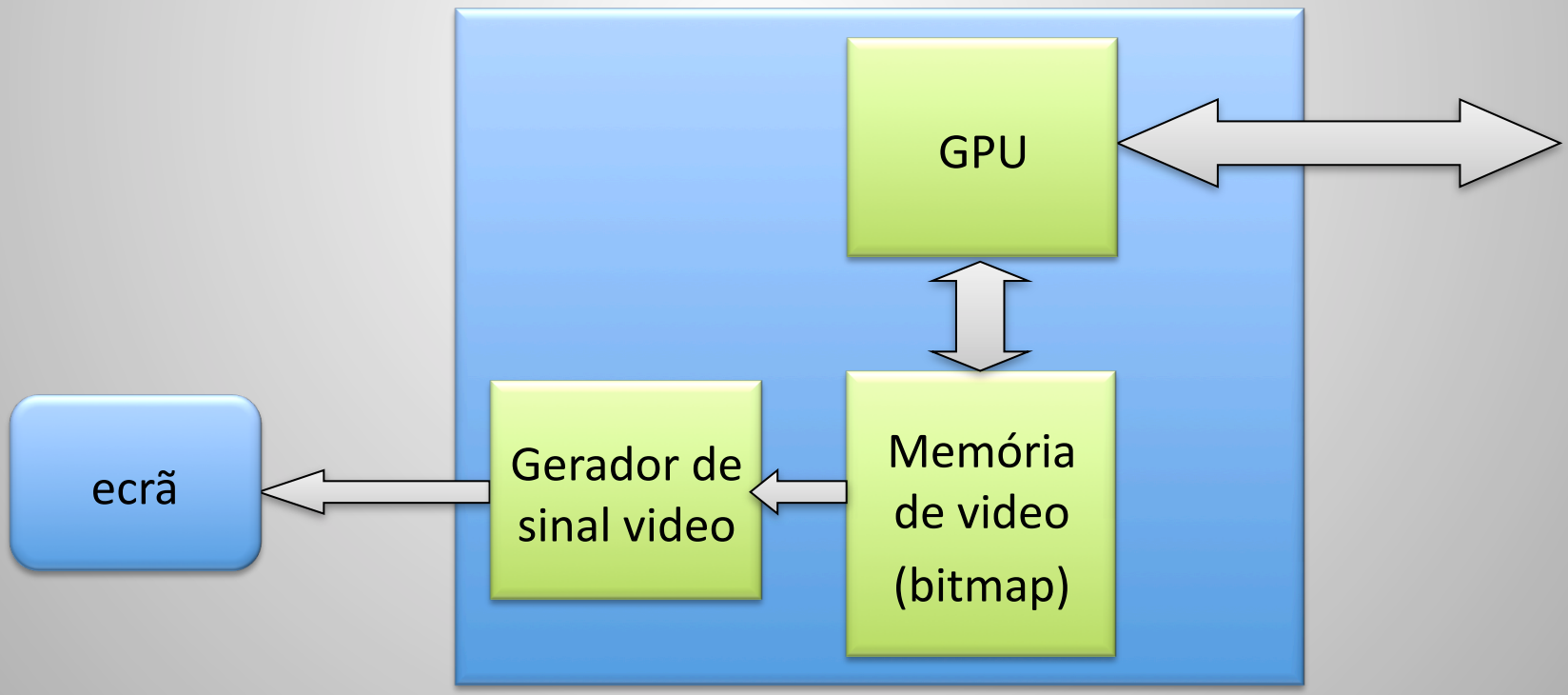
Arquitecturas de computadores

- Bus dedicado de alta velocidade



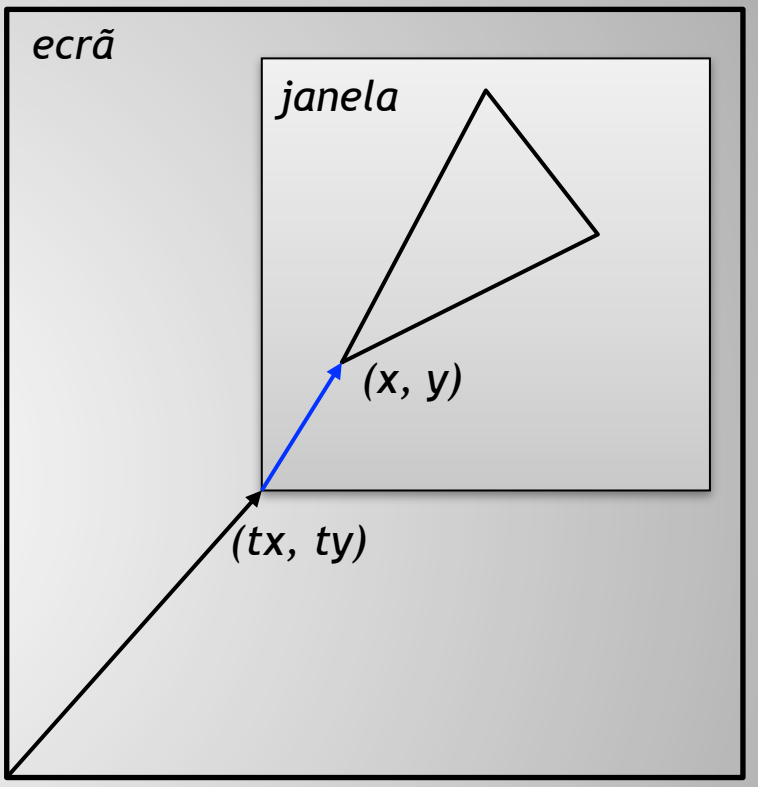
Coprocessador gráfico

- O controlador gráfico passa a ser "acelerado"
  - Inclui uma unidade co-processadora que ajuda o CPU na manipulação de gráficos
  - GPU - Graphics Processing Unit



"Aceleração" 2D

- O GPU suporta comandos para desenho 2D
  - Em vez do CPU "pintar" os pixeis, manda a GPU fazer essas operações
    - Poupa o CPU e o BUS
  - Exemplos de comandos suportados: linhas, figuras geométricas (elipses, polígonos, etc), e também copia/move zonas da imagem…
  - Estas operações podem também incluir operações de translação, rotação e escala
    - Operações vectoriais



Posição do triângulo na janela (x,y)
Posição da janela (tx, ty)
Posição do trângulo no ecrã (x+tx, y+ty)

Fases no desenho 3D

- Descrição da cena
  - Os objectos 3D
    - Exemplo: por aproximação de triângulos 3D
  - Descrição das superfícies:
    - Cor e textura
  - Descrição da cena:
    - Posição dos objectos e da iluminação
- Projectar a cena 3D no plano do ecrã (2D)
  - Modelo da máquina fotográfica

  (resolver: posições de cada objecto e o aspecto das partes visíveis)

“Aceleração” 3D

- O GPU recebe a descrição de toda a cena

- O GPU implementa um pipeline de operações:
  1. Aplica todas as transformações nos objectos para os posicionar e efectua a transformação 3D para 2D
     - Translações, rotações e escala
  2. Calcula o “aspecto” de cada face, tendo em conta as características dos objectos e da luz que lhe incide
  3. Identificando as partes visíveis e calculando a cor de cada pixel no ecrã

GPU actuais – um segundo computador

- São quase/ou tão complexos como o CPU
- Possuem um ISA bastante rico e eficiente
- Possuem dezenas de unidades processadoras, também chamadas cores, SIMD
- Exemplo: nVidia C1060

  1 GPU @ 600 MHz

  240 unidades processadoras (cores)

  4096 MByte de memória

  936 MFlops

GPGPUs

- General Purpose GPUs
- Usar a capacidade de processamento dos GPUs para fazer cálculo não visual
- Existem linguagens e biblotecas que permitem tal programação:
  - CUDA
    - Desenvolvido pela NVIDIA
    - Existem implementação para C
  - OpenCL (Open Computing Language)
    - Tentativa de definir uma linguagem para a desenvolvimento de aplicações em ambientes heterógeneos (CPUs + GPUs)
    - Desenvolvida pela Apple com apoio da AMD, Intel e NVIDIA

Sistemas multi-processador

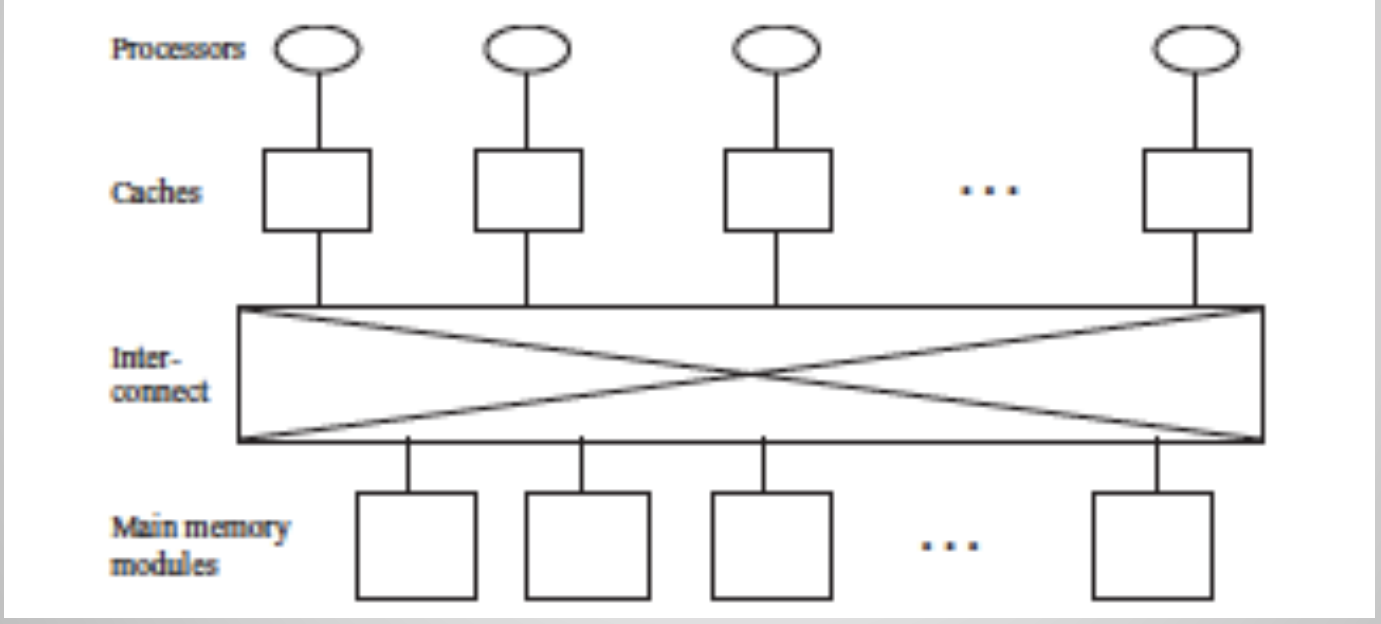
- Multiprocessadores de memória partilhada
  - UMA (Uniform Memory Access time) – qualquer CPU tem acesso a todos as células de memória no mesmo tempo
  - NUMA (Non-Uniform Memory Access time) – os tempos de acesso partes distintas da RAM não são os mesmos para um dado CPU
- Multiprocessadores de memória distribuída
  - Sistema constituído por múltiplos computadores completos (nós)
  - Em cada nó:
    - Só há acesso directo á memória local
    - Comunica-se com os outros nós usando um dispositivo de entrada/saída

SMP

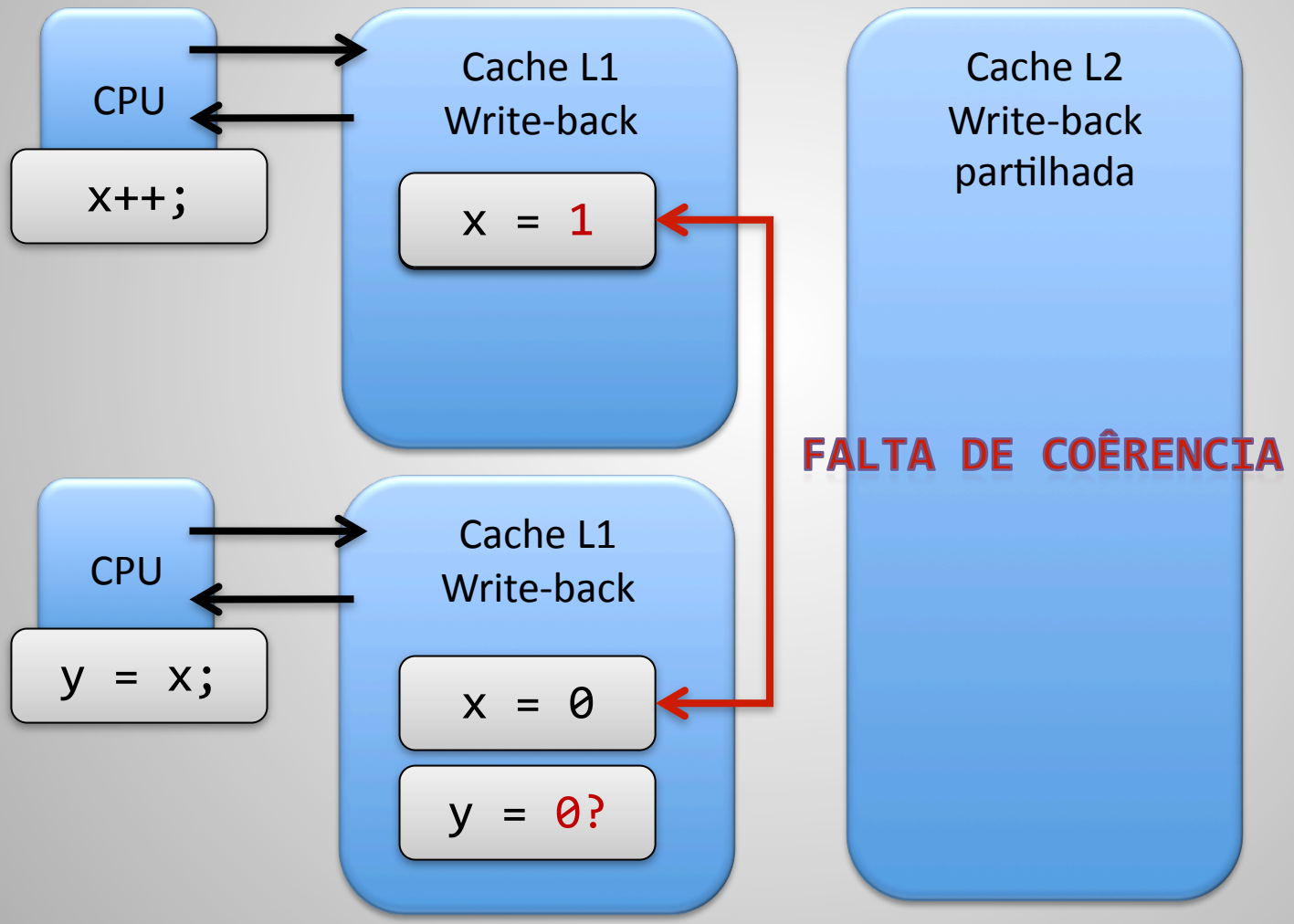
- Vantagens
  - Todos os CPU partilham o programa e os dados
  - Fácil e eficiente partilhar as alterações nos dados
  - O SO tira partido destas arquitecturas
- Desvantagens
  - Congestionamento no acesso à memória
  - As caches podem aliviar esse problema, mas colocam o problema das escritas poderem tornar as caches incoerentes
  - Impraticável para grande escala (muitos CPUs)

SMP

- Todos os CPUs têm acesso a qualquer módulo de memória no mesmo tempo



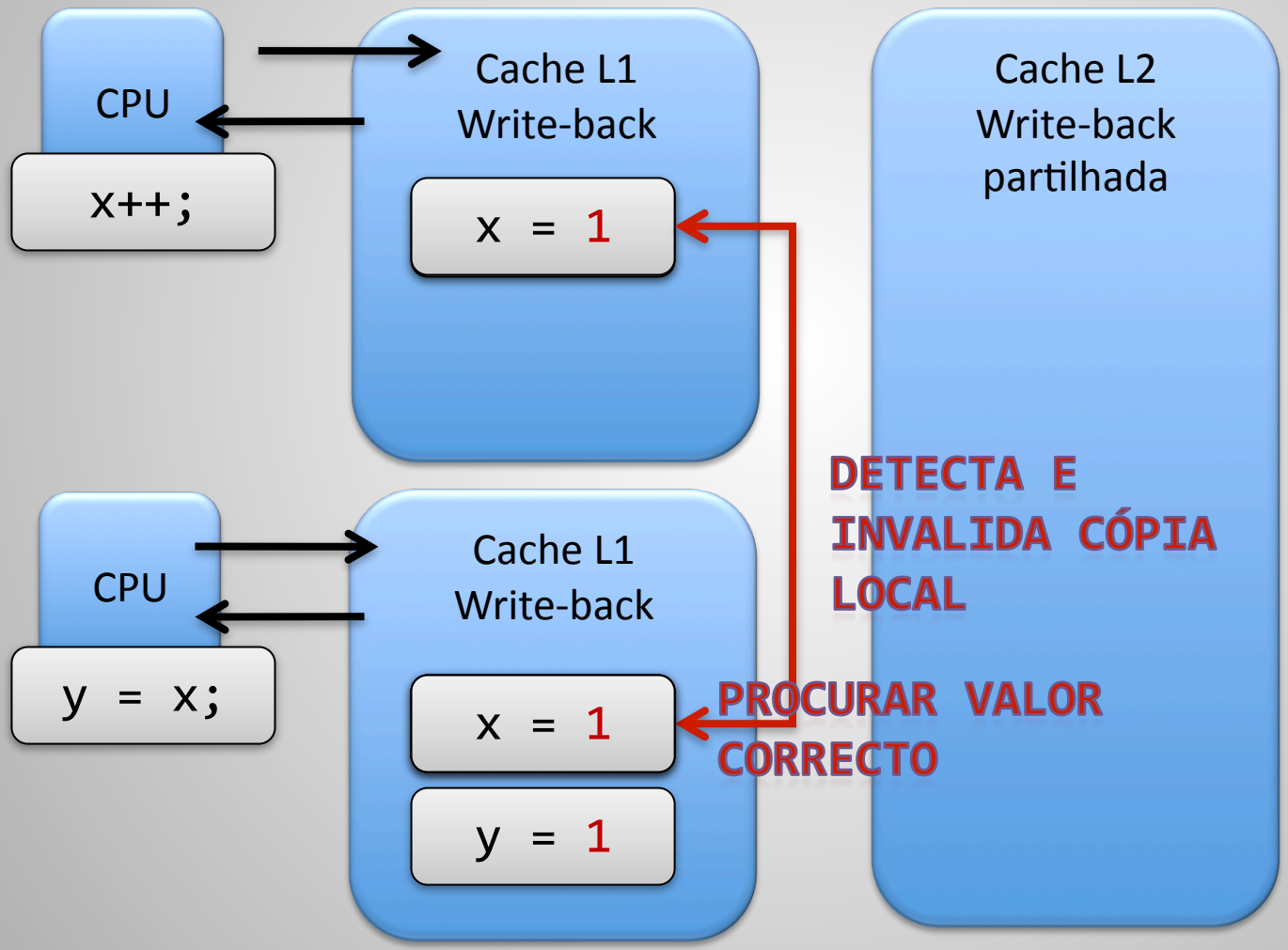
Coêrencia de caches



Coêrencia de caches

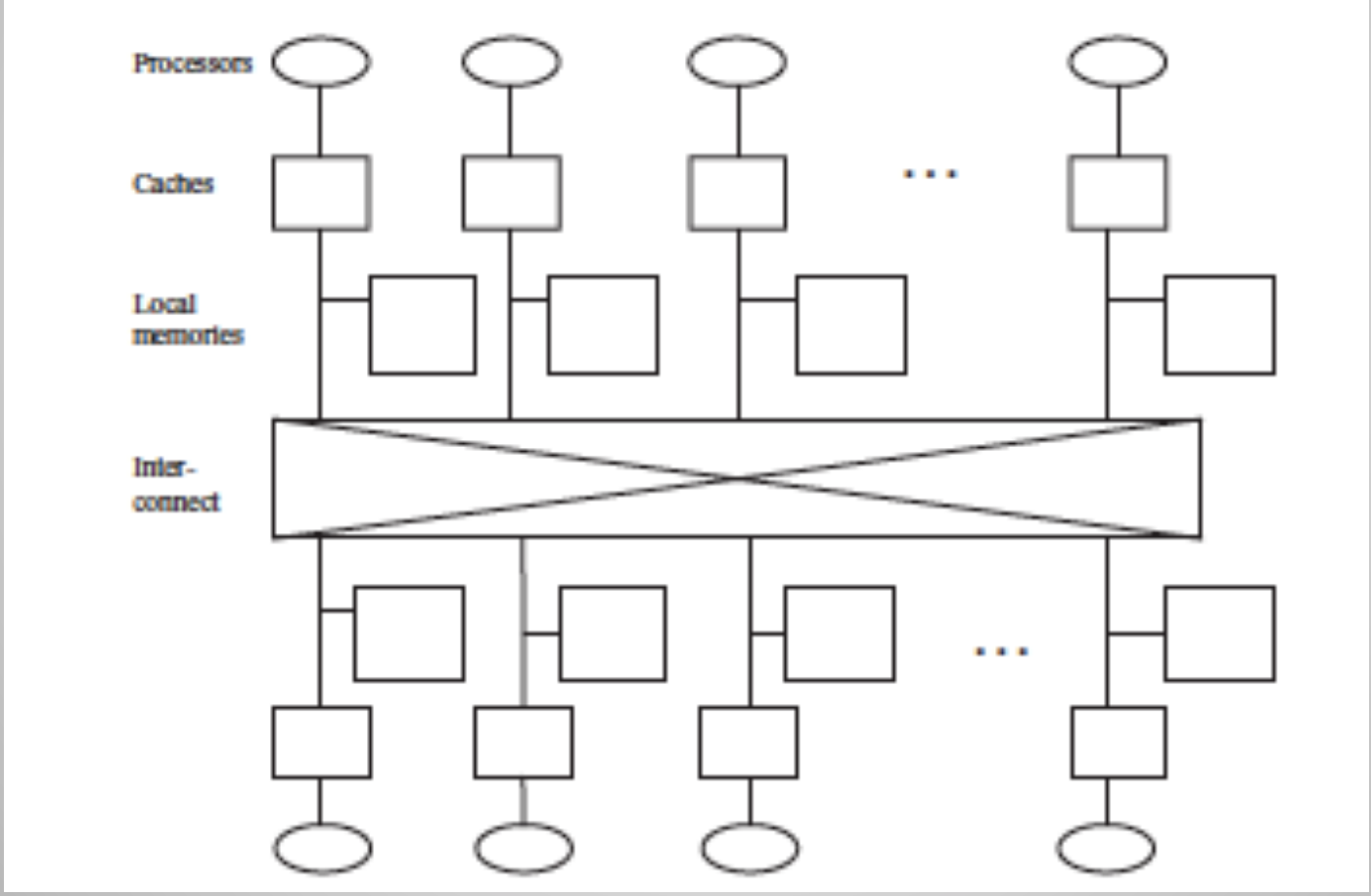
- É necessário ter um protocolo para garantir a coerência das caches
- Estes protocolos são normalmente protocolos de invalidação que invalidam as cópias dos blocos alterados
- É implementado em hardware

Coêrencia de caches



NUMA

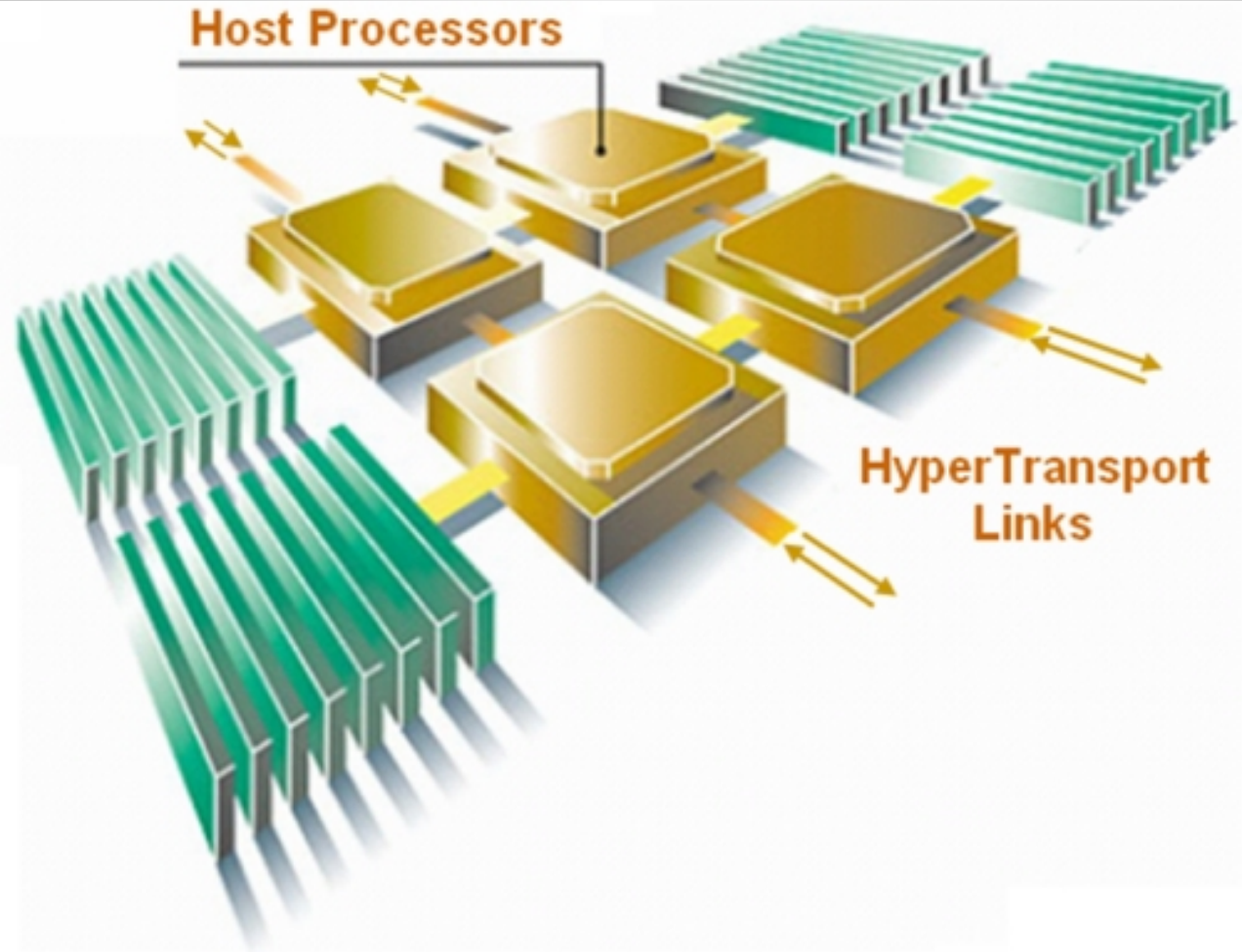
- O tempo de acesso à memória local de cada CPU é menor do que o tempo de acesso ás memórias locais de outros nós



AMD NUMA Hypertransport

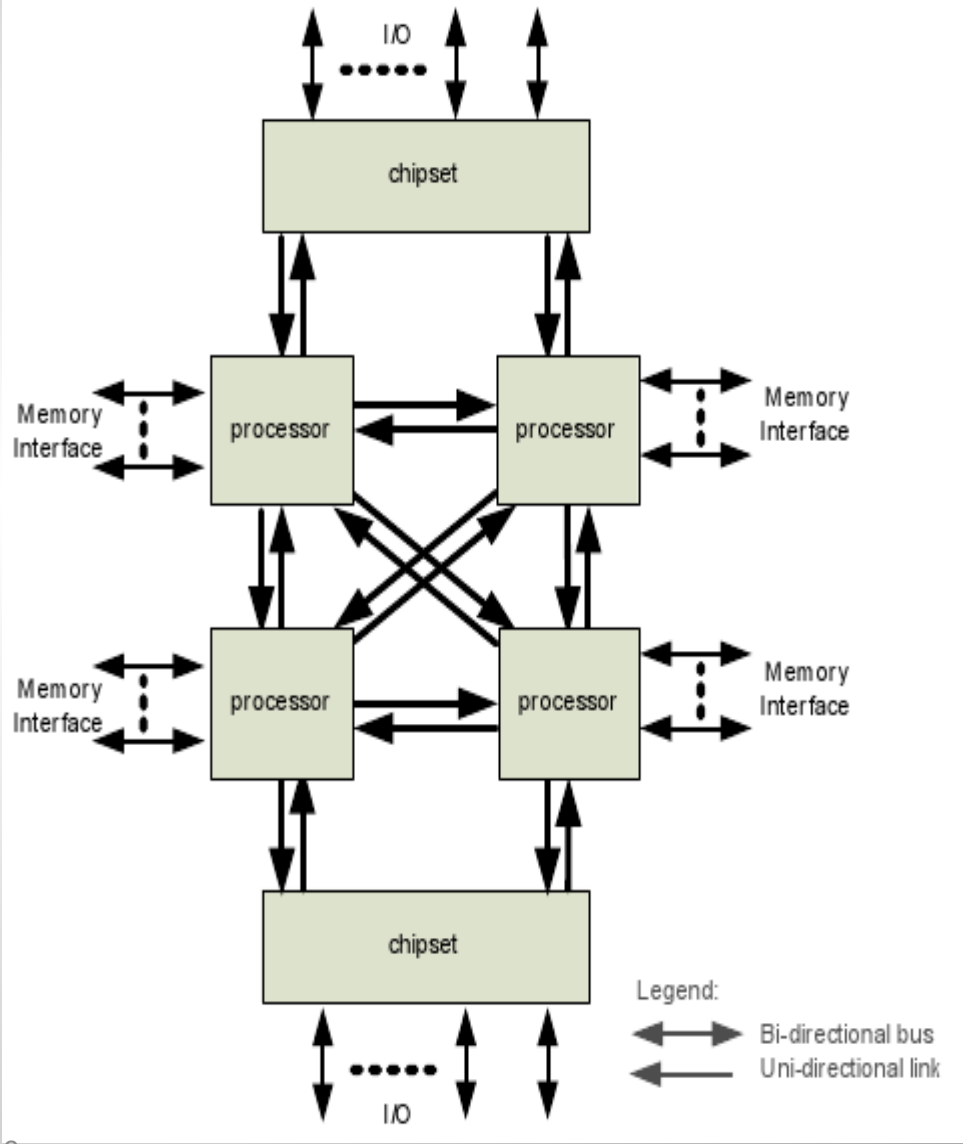
As máquinas com múltiplos CPUs AMD são NUMAS:

- Memória local a cada CPU tem acesso directo
- Memória de outros CPUs é via “HyperTransport Link”



Intel NUMA

- Quick-path
  - Proposta da Intel onde cada processador tem a sua memória dedicada
- Não é uma arquitectura SMP, é sim NUMA (Non-Uniform Memory Access)
  - O tempo de acesso a memória não é uniforme
  - Aceder a dados na memória de outro processador é mais lento do que aceder a dados na própria memória



MIMD memória distribuída

- Cada unidade processadora tem a sua própria memória
- Os vários nós estão próximos e ligados por redes muito rápidas
  - Gigabit
  - Infiniband
  - Myrinet