# INE5607 – Organização e Arquitetura de Computadores

## Hierarquia e Gerência de Memória

## Aula 30: Memórias modernas

Prof. Laércio Lima Pilla

laercio.pilla@ufsc.br




# Sumário

- Memória secundária
- Memória primária não volátil
- Memórias 3D
- Coerência de cache
- Exemplos de hierarquias de memória atuais
- Mecanismos de transferência de dados

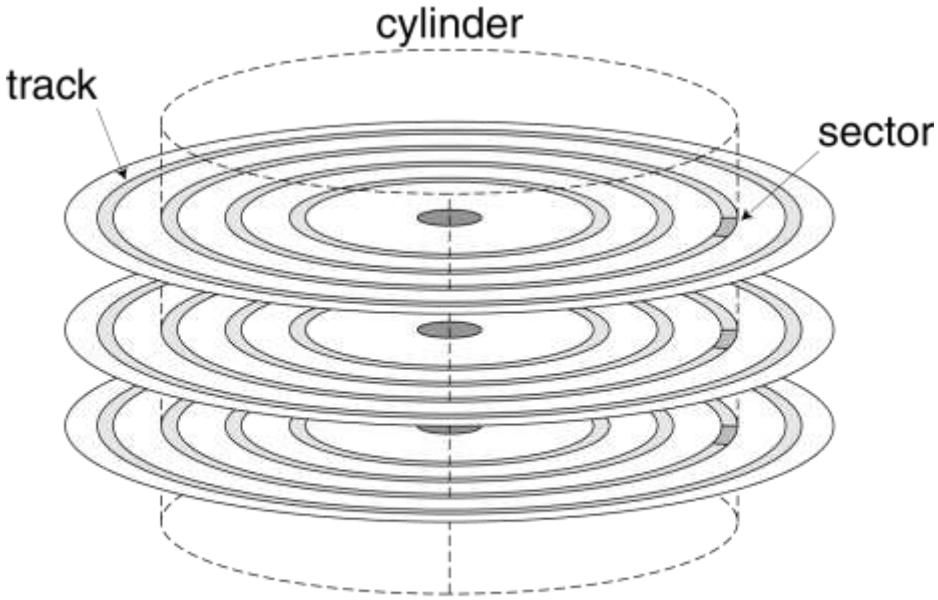
# MEMÓRIA SECUNDÁRIA

# Memória secundária

- **Memória secundária**
  - **Não volátil**
  - Armazena programas e dados enquanto não estão em execução
    - Ou páginas que não cabem na memória principal

# Memória secundária

- **Discos rígidos**
  - **HDD: *Hard Disk Drive***
  - Armazenamento magnético giratório




Figuras dos slides do capítulo 6 da 4ª ed. do livro-texto.

# Memória secundária

- **Discos rígidos**
  - Cada setor armazena
    - Identificador do setor
    - Dados (512 B, 4KB)
    - Código de correção de erros
      - Próxima aula
    - Campos de sincronização e espaçamentos

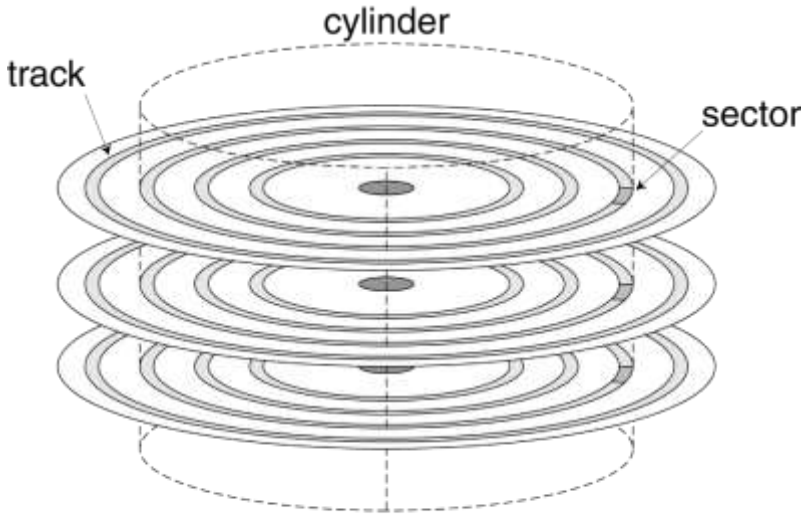



Figura dos slides do capítulo 6 da 4ª ed. do livro-texto.

# Memória secundária

- **Discos rígidos**
  - **Acesso a um setor** envolve
    - Atraso na **fila de acessos** se outros estão pendentes
    - **Tempo de busca** (*seek*): movimento das cabeças
    - **Latência rotacional**
    - **Transferência de dados**
    - Sobrecusto do controlador

# Memória secundária

- **Exemplo de acesso**
  - **Disco**
    - Setores de 512B, 15.000 RPM, tempo de busca médio de 4ms, taxa de transferência de 100 MB/s, atraso de controle de 0,2ms, disco ocioso
  - **Tempo de leitura médio**

    4ms tempo de busca

    +½ / (15000/60) = 2ms latência de rotação

    +512 / 100MB/s = 0,005ms tempo de transferência

    +0,2ms atraso de controle

    = 6,2ms

# Memória secundária

- **Exemplo de acesso**
  - **Disco**
    - Setores de 512B, 15.000 RPM, tempo de busca médio de **1ms**, taxa de transferência de 100 MB/s, atraso de controle de 0,2ms, disco ocioso
  - **Tempo de leitura médio**

    **1ms** tempo de busca

    +½ / (15000/60) = 2ms latência de rotação

    +512 / 100MB/s = 0,005ms tempo de transferência

    +0,2ms atraso de controle

    = **3,2ms** (6,2ms anteriormente)

# Memória secundária

- **Questões de desempenho de disco**
  - Tempo de busca médio dados por fabricantes
    - Baseados em todas possíveis buscas
    - **Escalonamento e informações de localidade levam a tempos de busca menores**
  - Controladores de disco alocam setores físicos e apresentam setores lógicos para o hospedeiro
  - **Discos incluem caches**
    - Fazem ***prefetch*** antecipando acessos
    - Evitam atraso de leitura e rotação

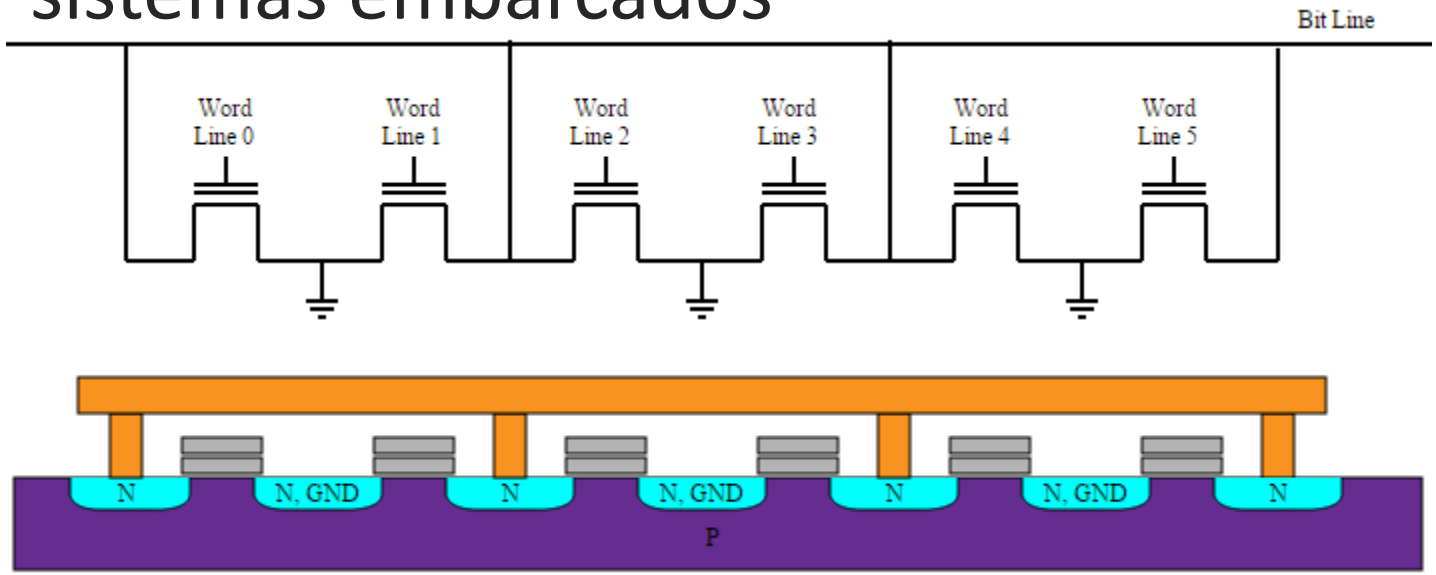
# Memória secundária

- **Memórias Flash**
  - **Armazenamento em semicondutores**
  - Não volátil
  - ~10 vezes mais rápidos do que discos
  - Menores, menor consumo
  - Mais $/GB


Figuras dos slides do capítulo 6 da 4ª ed. do livro-texto.

# Memória secundária

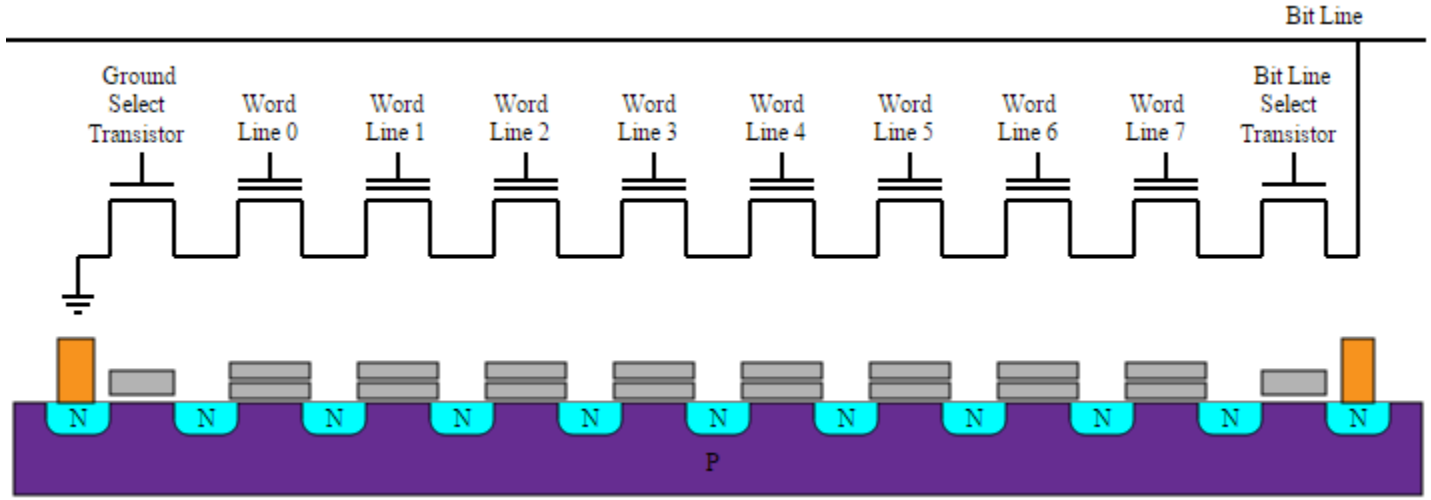
- **Flash NOR**
  - Acessos aleatórios em **palavras**
  - Pouco consumo estático
  - Usados para memórias de instruções em sistemas embarcados




"NOR flash layout" by wikipedia user Cyferz. Licensed under CC BY-SA 3.0 via Wikimedia Commons https://commons.wikimedia.org/wiki/File:NOR_flash_layout.svg#/media/File:NOR_flash_layout.svg

# Memória secundária

- **Flash NAND**
  - Mais denso, acesso a **blocos** ao invés de palavras
  - Mais baratos por GB
  - Usado em pendrives, ***Solid-State Drives***




"Nand flash structure" by wikipedia user Cyferz. Licensed under CC BY-SA 3.0 via Wikimedia Commons https://commons.wikimedia.org/wiki/File:Nand_flash_structure.svg#/media/File:Nand_flash_structure.svg

# Memória secundária

- **Memórias Flash**
  - **Desgaste** devido a escritas
    - **Bloco precisa ser apagado antes da escrita**
    - ~100K a 1M antes de bits ficarem “presos”
    - Não serve para ser usado como RAM
  - **Remapeamento de blocos**
    - Blocos migrados entre células
    - Evita usar demais uma célula

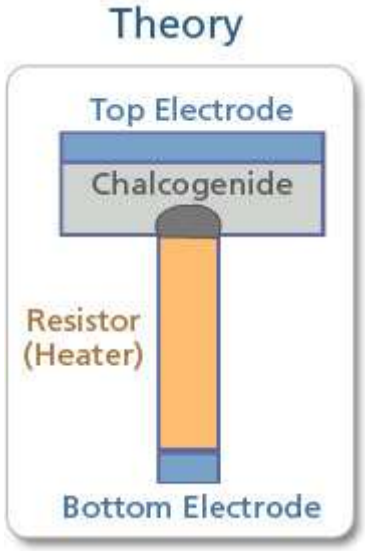
# MEMÓRIA PRIMÁRIA NÃO VOLÁTIL

# Memória primária não volátil

- **Memória primária não volátil**
  - RAM não volátil!!!
  - **Vantagens**
    - Não precisa ser realimentado periodicamente
      - Economiza energia
    - Dados se mantém em memória em caso de queda energética
  - **Desvantagens** (até o momento)
    - Mais cara
    - Escrita mais lenta
    - Limite de escritas (como Flash)

# Memória primária não volátil

- **PCM: *Phase Change Memory***
  - Altera as propriedades físicas e elétricas de materiais para armazenar informações
    - Como em CDs e DVDs
  - **Calcogenetos**
    - **Altera fase com calor**
      - Esfriamento rápido: amorfo
      - Esfriamento lento: cristalino




http://www.micron.com/about/innovations/pcm

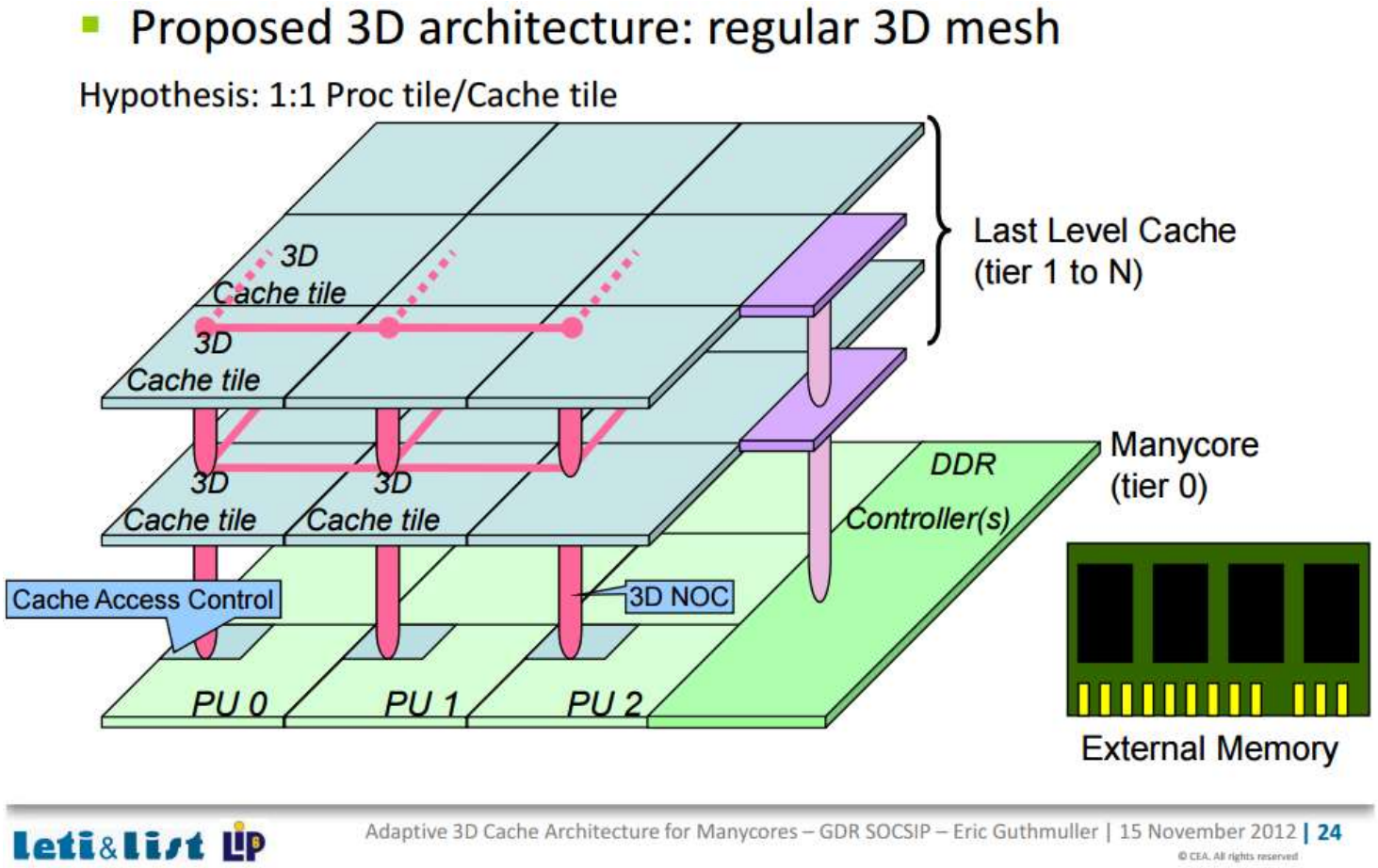
# MEMÓRIAS 3D

# Memórias 3D

- **Estrutura padrão de memórias**
  - Bidimensional
    - Área ocupada por cache no processador
    - Chips de memória RAM em uma placa
- **Circuitos integrados tridimensionais**
  - Empilhamento de camadas bidimensionais
    - **Menor área**
    - **Menos potência**
    - **Menores distâncias para transmitir dados**

# Memórias 3D

- **Exemplos:** Cache 3D




Adaptive 3D Cache Architecture for Manycores. Eric Guthmuller
http://www.lip6.fr/public/2012-11-15_Guttmuller_Memoire_3D.pdf

# Memórias 3D

- **Exemplos:** Hierarquia de memória 3D

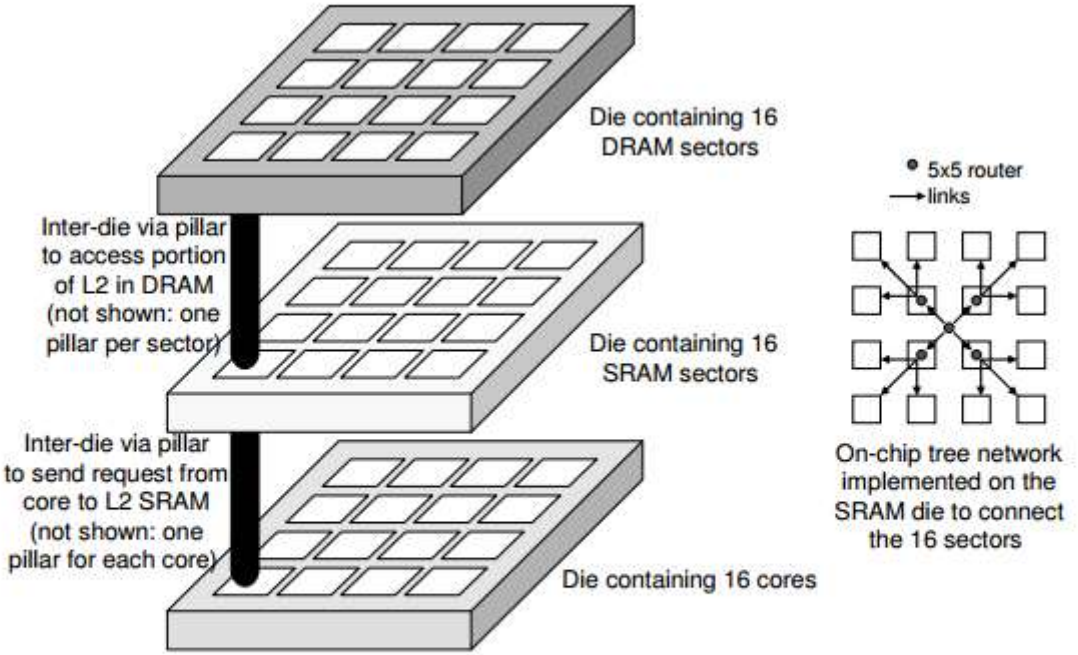
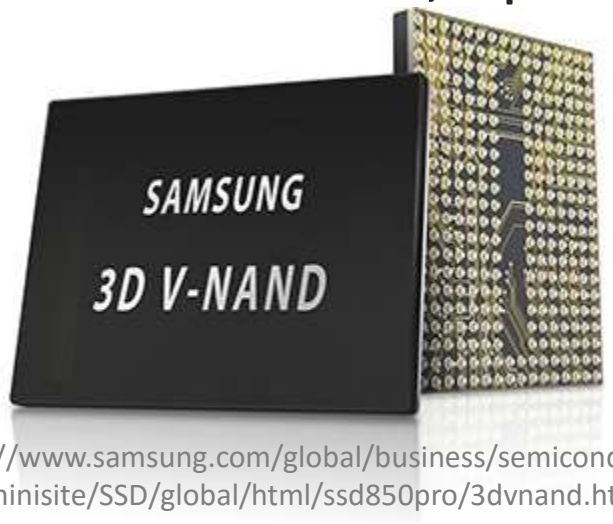



**Figure 1. Stacked 3D processor layout. The bottom die contains the 16 cores, the second die contains 16 SRAM banks, and the top die contains 16 DRAM banks. Inter-die via pillars (one for each bank) are used to implement vertical connections between cores and DRAM/SRAM banks. Horizontal communication happens on the on-chip tree network (shown on the right) implemented on the second die. There are no horizontal connections between cores (banks) on the bottom (top) die.**

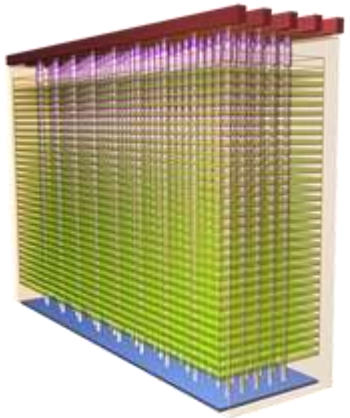
Niti Madan et al. Optimizing Communication and Capacity in a 3D Stacked Reconfigurable Cache Hierarchy. http://www.hpl.hp.com/people/naveen_muralimanohar/hpca09.pdf

# Memórias 3D

- **Exemplos:** Flash NAMD 3D
  - Mecanismo para aumentar a capacidade de SSDs
  - Feito por empresas como Samsung e Micron
    - 32 andares de NAMD
    - 10TB em 2,5 polegadas




http://www.samsung.com/global/business/semiconductor/minisite/SSD/global/html/ssd850pro/3dvnand.html



http://www.micron.com/about/innovations/3d-nand

# COERÊNCIA DE CACHE

# Coerência de cache

- **Problema de coerência de cache**
  - Suponha dois núcleos (A e B) compartilhando um espaço de endereçamento

| Etapa | Evento | Cache núcleo A | Cache núcleo B | Memória principal |
|---|---|---|---|---|
| 0 | | | | 0 |
| 1 | A lê end. X | 0 | | 0 |
| 2 | B lê end. X | 0 | 0 | 0 |
| 3 | A escreve 1 no end. X | 1 | **0** | 1 |

# Coerência de cache

- **Compreensão informal**
  - P escreve em X; P lê de X (sem escritas no meio tempo) → leitura retorna valor escrito
    - **Ordem do programa**
  - $P_1$ escreve em X; $P_2$ lê de X (mais tarde) → leitura retorna valor escrito
    - **Visão coerente da memória**
  - $P_1$ escreve em X; $P_2$ escreve em X → todos núcleos enxergam escritas na mesma ordem
    - **Serialização das escritas**
    - Terminam com o mesmo valor para o endereço X

# Coerência de cache

- **Operações realizadas para garantir coerência em multiprocessadores**
  - **Migração** de dados para caches locais
    - Redução largura de banda para memória compartilhada
  - **Replicação** de dados
    - Bom para dados somente de leitura
    - Reduz contenção de acesso

# Coerência de cache

- **Protocolos de coerência de cache**
  - Suporte a migração e replicação
    - **Invalidam valores desatualizados**
  - **Protocolos baseados em *snooping***
    - Cada cache monitora leituras e escrita em um barramento compartilhado
      - Menos escalável
  - **Protocolos baseados em diretórios**
    - Caches e memória mantêm registro do estado de blocos
      - Divididos em diretórios
      - Uma cache é responsável por um bloco

# Coerência de cache

- **Exemplo de protocolo de *snooping***
  - Mensagem de invalidação para todas caches
    - *Broadcast*
  - Leituras levam a falhas nas outras caches

| Evento | Atividade no barramento | Cache núcleo A | Cache núcleo B | Memória principal |
|---|---|---|---|---|
| | | | | 0 |
| A lê end. X | Falha p/ X | 0 | | 0 |
| B lê end. X | Falha p/ X | 0 | 0 | 0 |
| A escreve 1 no end. X | Invalida X | 1 | - | 0 |
| B lê end. X | Falha p/ X | 1 | 1 | 1 |

# Coerência de cache

- **Quando escritas são vistas por outros núcleos?**
  - "Enxergar": uma leitura retorna o valor escrito
  - Não pode ser instantâneo
  - **Suposições**
    - Escrita completa apenas quando todos núcleos a viram
    - Núcleo não reordena escritas
  - **Consequência**
    - P escreve em X; P escreve em Y → todos núcleos que veem o novo Y também veem o novo X

# Coerência de cache

- **Consequências**
  - Compartilhamento de variáveis exige **sincronização**
    - Evitar **condições de corrida**
    - Garantir que escrita será vista por todos
  - Escritas em um mesmo bloco por diferentes núcleos geram invalidações
    - Podem não ser na mesma palavra
    - ***False sharing***
      - Compartilhamento falso

# EXEMPLOS DE HIERARQUIAS DE MEMÓRIA ATUAIS

# Exemplos atuais

- **ARM Cortex-A8 e Intel Core i7**

| Característica | ARM Cortex-A8 | Intel Core i7 |
|---|---|---|
| Endereço virtual | 32 bits | 48 bits |
| Endereço físico | 32 bits | 44 bits |
| Tamanho de página | Variável: 4, 16, 64 KB, 1, 16 MB | Variável 4 KB, 2/4 MB |
| Organização de TLB | *não foi tratado em aula | |
| **Cache L1** | **ARM Cortex-A8** | **Intel Core i7** |
| Organização | Dividido entre instruções e dados | |
| Tamanho | 32KB cada | 32KB cada |
| Associatividade | 4-way | 4-way (I), 8-way (D) |
| Substituição | Aleatória | LRU aproximado |
| Tamanho de bloco | 64 bytes | 64 bytes |
| Política de escrita | *não foi tratado em aula | |
| Tempo de acerto | 1 ciclo | 4 ciclos, pipelined |

# Exemplos atuais

| Cache L2 | ARM Cortex-A8 | Intel Core i7 |
|---|---|---|
| Organização | Unificado | Unificado por núcleo |
| Tamanho | 128 KB a 1 MB | 256 KB |
| Associatividade | 8-way | 8-way |
| Substituição | Aleatório | LRU aproximado |
| Tamanho de bloco | 64 bytes | 64 bytes |
| Política de escrita | *não foi tratado em aula | |
| Tempo de acerto | 11 ciclos | 10 ciclos |
| **Cache L3** | **ARM Cortex-A8** | **Intel Core i7** |
| Organização | - | Unificado |
| Tamanho | - | 8 MB |
| Associatividade | - | 16-way |
| Substituição | - | LRU aproximado |
| Tamanho de bloco | - | 64 bytes |
| Política de escrita | *não foi tratado em aula | |
| Tempo de acerto | - | 35 ciclos |

# Exemplos atuais

- **Nvidia Maxwell**
  - 2048 núcleos
    - 2 MB L2 cache
    - 4 GB GDDR5
  - SMM
    - Grupos de 128 núcleos
    - Cache de instr.
    - Cache L1
    - Cache texturas
    - Memória compartilhada


http://devblogs.nvidia.com/parallelforall/maxwell-most-advanced-cuda-gpu-ever-made/

# MECANISMOS DE TRANSFERÊNCIA DE DADOS

# Mecanismos de transferência de dados

- **Comunicação necessária entre componentes em um computador**
  - Memória e memória secundária
  - Memória e memória da GPU
  - Memória e dispositivo de saída
  - Dispositivo de entrada e processador
  - Disco e impressora
  - etc.

# Mecanismos de transferência de dados

- **O que dificulta a transferência de dados (E/S)?**
  - **Diversos programas** usando o processador **compartilham o sistema de E/S**
  - E/S normalmente envolve interrupções, o que causa uma transferência ao modo *kernel*
  - O **controle** de baixo nível de dispositivos é **complexo e variado**
    - Uso de drivers

# Mecanismos de transferência de dados

- **Papéis do sistema operacional**
  - Garantir que um programa acessa apenas partes do dispositivo que o usuário tem acesso
    - Não pode escrever em um arquivo sem direito
  - Abstrair acesso a dispositivos através de rotinas
  - Tratar interrupções vindo dos dispositivos
  - Controlar o quanto cada programa usa a E/S
    - Justiça (*fairness*): um processo não pode monopolizar o sistema
    - Desempenho: aumentar a vazão de dados

# Mecanismos de transferência de dados

- **Mecanismos de monitoramento de transferência de dados**
  - **Polling**: consulta periódica ao dispositivo
  - **Interrupção**: dispositivo solicita a interrupção do fluxo de execução
  - **DMA**: acesso direto à memória

# Mecanismos de transferência de dados

- **Polling**
  - **Funcionamento**
    - Dispositivo escreve informações da transferência de dados em um registrador de controle
    - CPU lê registrador periodicamente
  - **Vantagem**
    - **Método mais simples** de comunicação
      Implementação totalmente em software!
  - **Desvantagem**
    - **Pode desperdiçar muito tempo da CPU**
      - CPU é em geral mais rápida que dispositivo de E/S
      - Muitas leituras sem novidades

# Mecanismos de transferência de dados

- **Interrupção**
  - **Funcionamento**
    - Interrompe a CPU apenas quando transferências precisam de atenção
  - **Vantagens**
    - **Mecanismo de interrupção libera a CPU**
      - Não exige consulta periódica
  - **Desvantagens**
    - **Ainda sobrecarrega a CPU com transferência e gerência**
      - Melhor do que polling mas não é mágico

# Mecanismos de transferência de dados

- **DMA: *Direct Memory Access***
  - **Funcionamento**
    - **Acesso direto à memória**
    - Implementado usando um **controlador especial e dedicado** configurado quando necessário
  - **Vantagens**
    - **Possibilita altas taxas de transferências**
      - Sobrecusto de configuração atenuado pela quantidade de dados transferidos
  - **Desvantagens**
    - **Requer controlador e interface mais complexa**
      - Mais hardware, mais potência, mais área...

# Mecanismos de transferência de dados

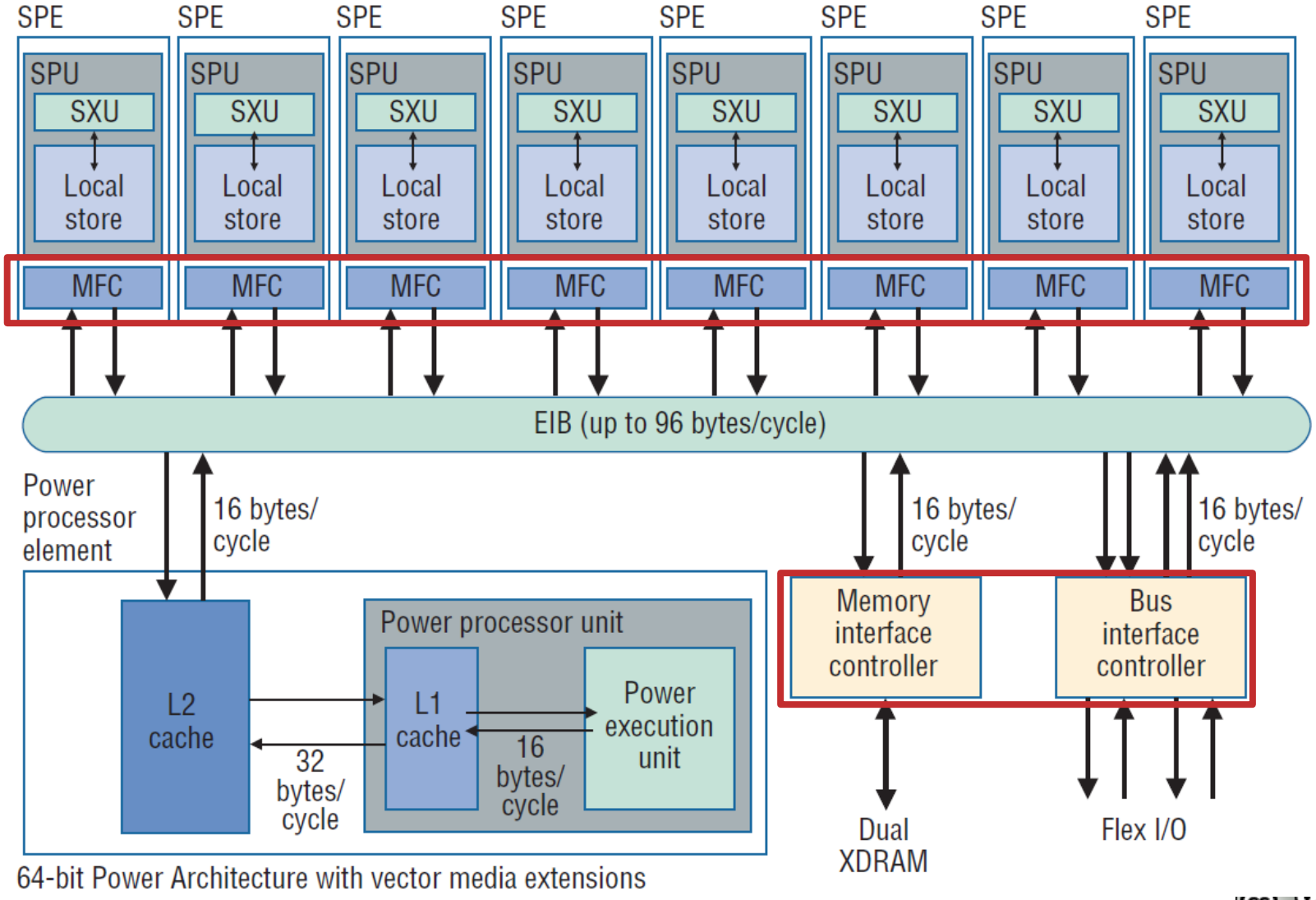
- **DMA: *Direct Memory Access***
  - **Funcionamento**
    - **Acesso direto à memória**
    - Implementado usando um **controlador especial e dedicado** configurado quando necessário
  - **Vantagens**
    - **Possibilita altas taxas de transferências**
      - Sobrecusto de configuração atenuado pela quantidade de dados transferidos
  - **Desvantagens**
    - **Requer controlador e interface mais complexa**
      - Mais hardware, mais potência, mais área...

# Mecanismos de transferência de dados

- **Avanços em DMA**
  - **Em desktops, não há mais um controlador centralizado de DMA**
    - Cada mestre de barramento age como *DMA engine*
  - **Processadores de E/S**
    - Como DMA, mas mais avançado
    - Recebe instruções para executar ao invés de comando da CPU
  - **RDMA: DMA remoto**

# Mecanismos de transferência de dados

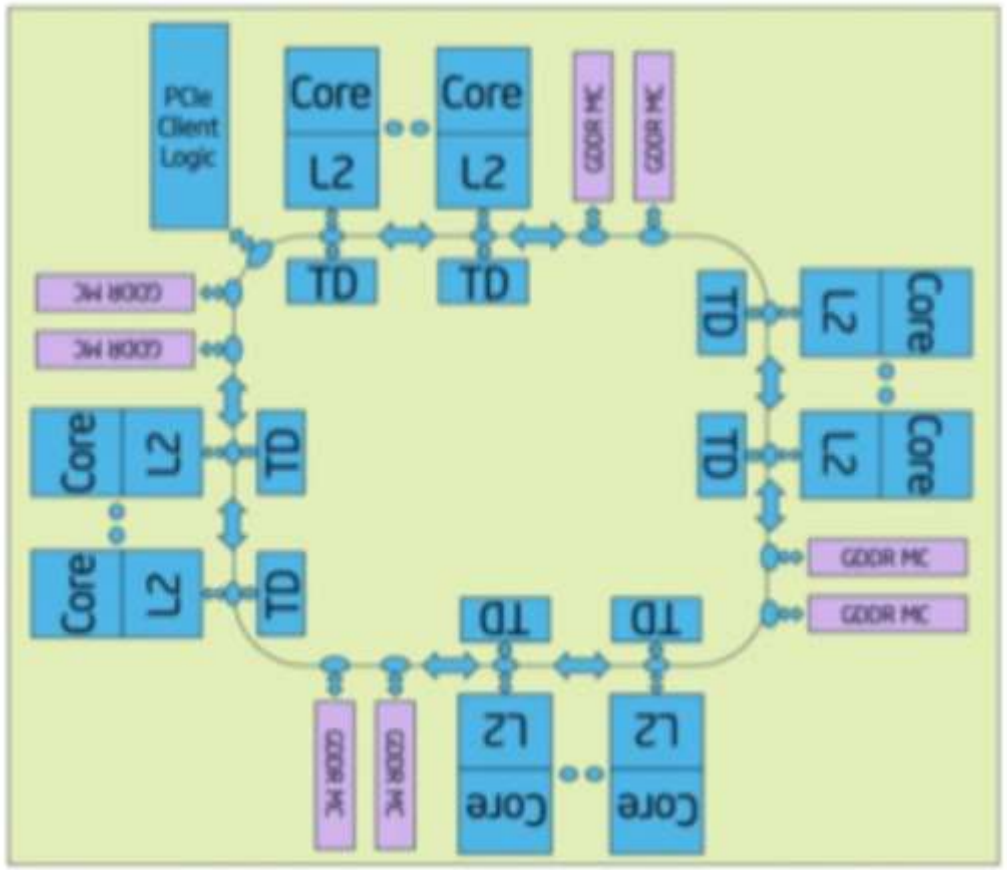
- Exemplos: Processador Cell




64-bit Power Architecture with vector media extensions

# Mecanismos de transferência de dados

- Exemplo: Acelerador Xeon Phi

# Mecanismos de transferência de dados

- Exemplo: Acelerador Xeon Phi

*2.1.8.2.1 DMA Capabilities*
*Direct Memory Access (DMA) is a common hardware function within a computer system that is used to relieve the CPU from the burden of copying large blocks of data. To move a block of data, the CPU constructs and fills a buffer, if one doesn't already exist, and then writes a descriptor into the DMA Channel's Descriptor Ring. A descriptor describes details such as the source and target memory addresses and the length of data in cache lines. The following data transfers are supported:*

- *Intel® Xeon Phi™ coprocessor to Intel® Xeon Phi™ coprocessor GDDR5 space (aperture)*
- *Intel® Xeon Phi™ coprocessor GDDR5 to host System Memory*
- *Host System Memory to Intel® Xeon Phi™ coprocessor GDDR5 (aperture or non-aperture)*
- *Intra-GDDR5 Block Transfers within Intel® Xeon Phi™ coprocessor*

# Mecanismos de transferência de dados

- Exemplo: Acelerador Xeon Phi

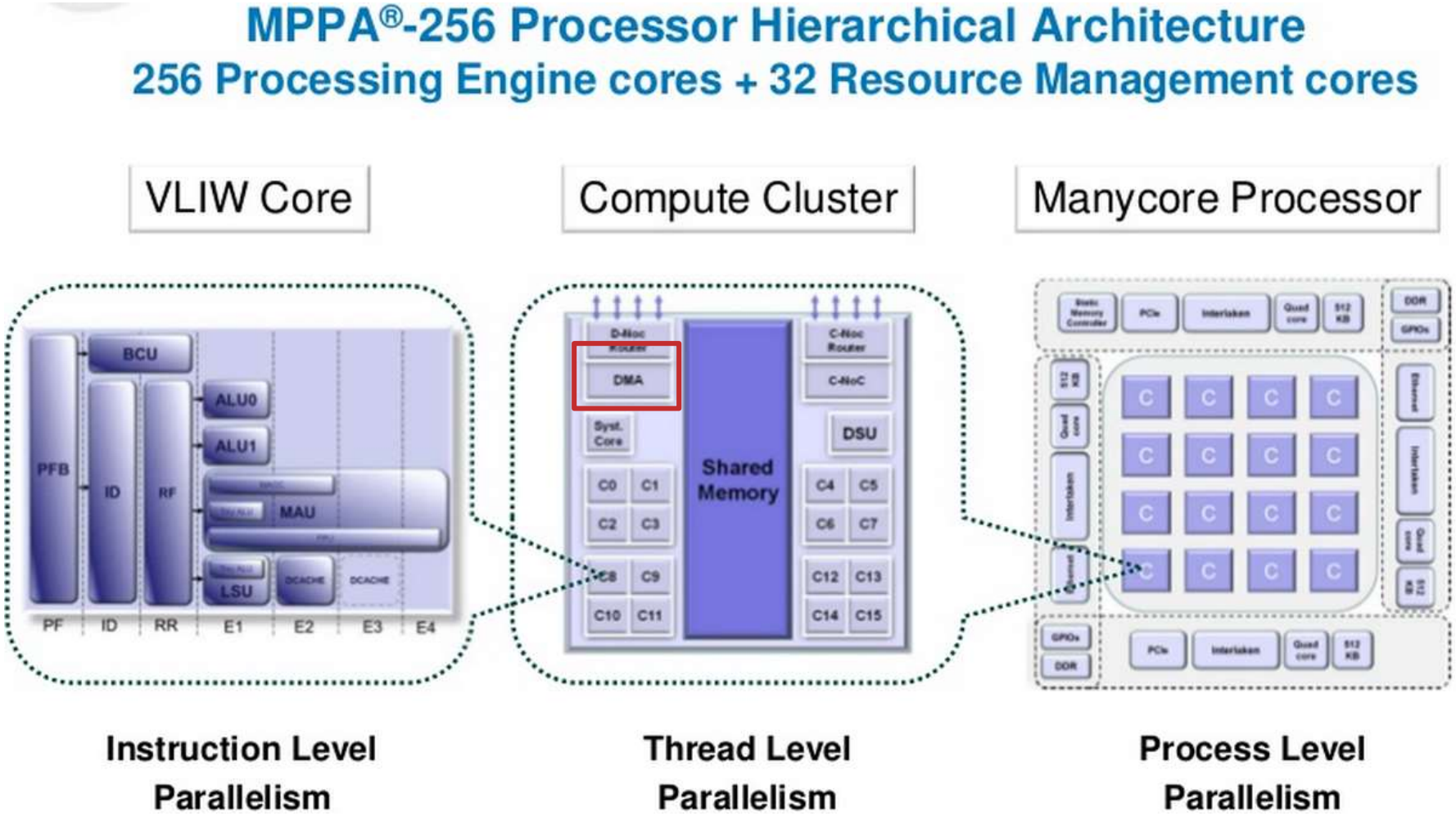
*In summary, the DMA controller has the following capabilities:*

- *8 DMA channels operating simultaneously, each with its own independent hardware ring buffer that can live in either local or system memory*
- *Supports transfers in either direction (host / Intel® Xeon Phi™ coprocessor devices)*
- *Supports transfers initiated by either side*
- *Always transfers using physical addresses*
- *Interrupt generation upon completion*
- *64-byte granularity for alignment and size*
- *Writing completion tags to either local or system memory*

*Intel® Xeon Phi™ Coprocessor System Software Developers Guide*
http://www.intel.com.br/content/dam/www/public/us/en/documents/product-briefs/xeon-phi-coprocessor-system-software-developers-guide.pdf

# Mecanismos de transferência de dados

- Exemplo: Kalray MPPA-256



# Mecanismos de transferência de dados

- Exemplo: Kalray MPPA-256

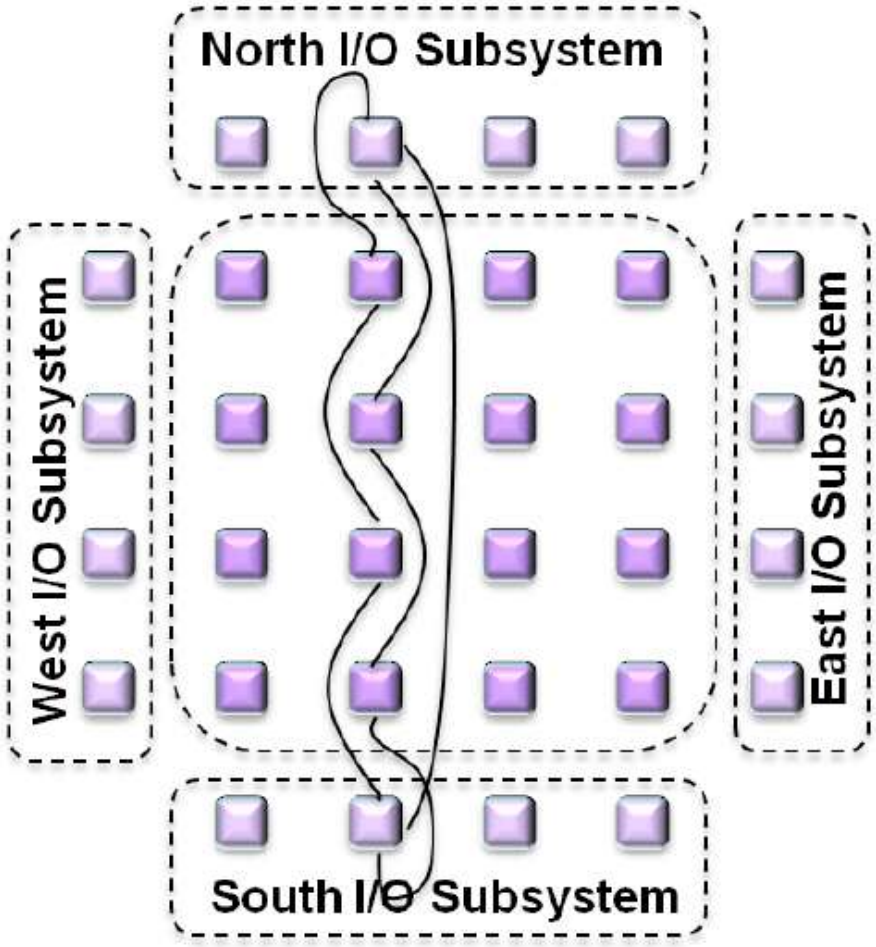



Figure 3.1: MPPA® MANYCORE NoC nodes and sample links.

# CONSIDERAÇÕES FINAIS

# Considerações finais

- **Estruturas de memória em constante evolução**
  - Memória secundária mais rápida
  - Memória primária não volátil
- **Compartilhamento de memória em processadores modernos**
  - Níveis de cache compartilhados
    - Precisam manter a **coerência** dos dados

# INE5607 – Organização e Arquitetura de Computadores

## Hierarquia e Gerência de Memória

## Aula 30: Memórias modernas

Prof. Laércio Lima Pilla

laercio.pilla@ufsc.br