Arquitetura e Organização de Computadores

Guilherme Henrique de Souza Nakahata

Universidade Estadual do Paraná - Unespar

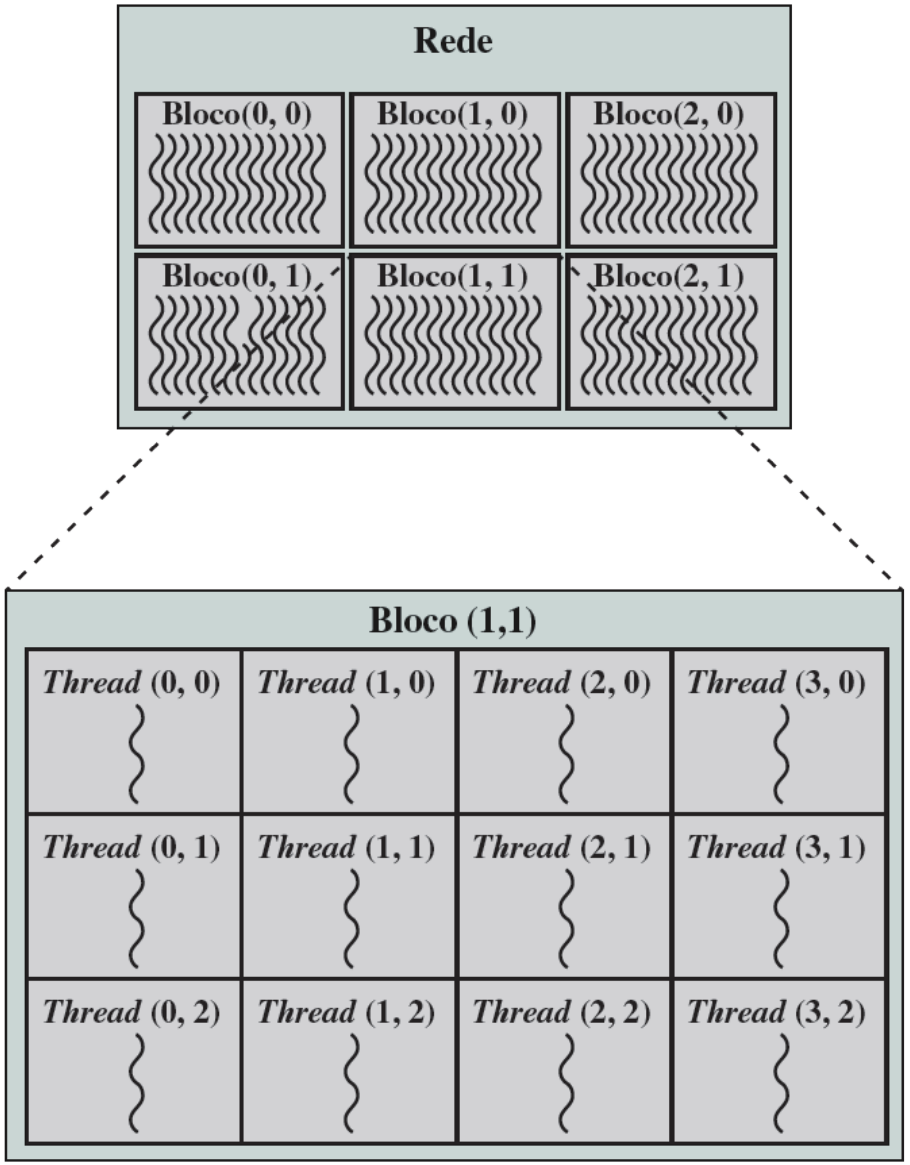
11 de Junho de 2024

Noções básicas sobre a CUDA

- A **CUDA** é uma plataforma de computação paralela e de modelo de programação criada pela NVIDIA e implementada pelas unidades de processamento gráfico (GPUs) que elas produzem.
- A CUDA C é uma linguagem baseada em C/C++.
- Um programa em CUDA pode ser dividido em:
  - O código a ser executado pelo host (CPU);
  - O código a ser executado pelo dispositivo (GPU);
  - O código relacionado à transferência de dados
  - Entre o host e o dispositivo.

Noções básicas sobre a CUDA

- Relação entre threads, blocos e uma rede:



Threads

```
void iniciaThreadsGVNS(FuncaoObjetivo funcaoObjetivo, bool debug, Solucao solucaoInicial, Parametro parametro, float porcentagem, int epocas,int
qtdCamadas){

        vector<thread> threads;

        // for(int i = 0; i < thread::hardware_concurrency(); i++){
        for(int i = 0; i < 10; i++){
                threads.push_back(thread(iniciaGVNS, funcaoObjetivo,debug,solucaoInicial,parametro, porcentagem, epocas, i,qtdCamadas));
        }

        for(int i = 0; i < threads.size(); i++){
                threads[i].join();
        }
}
```

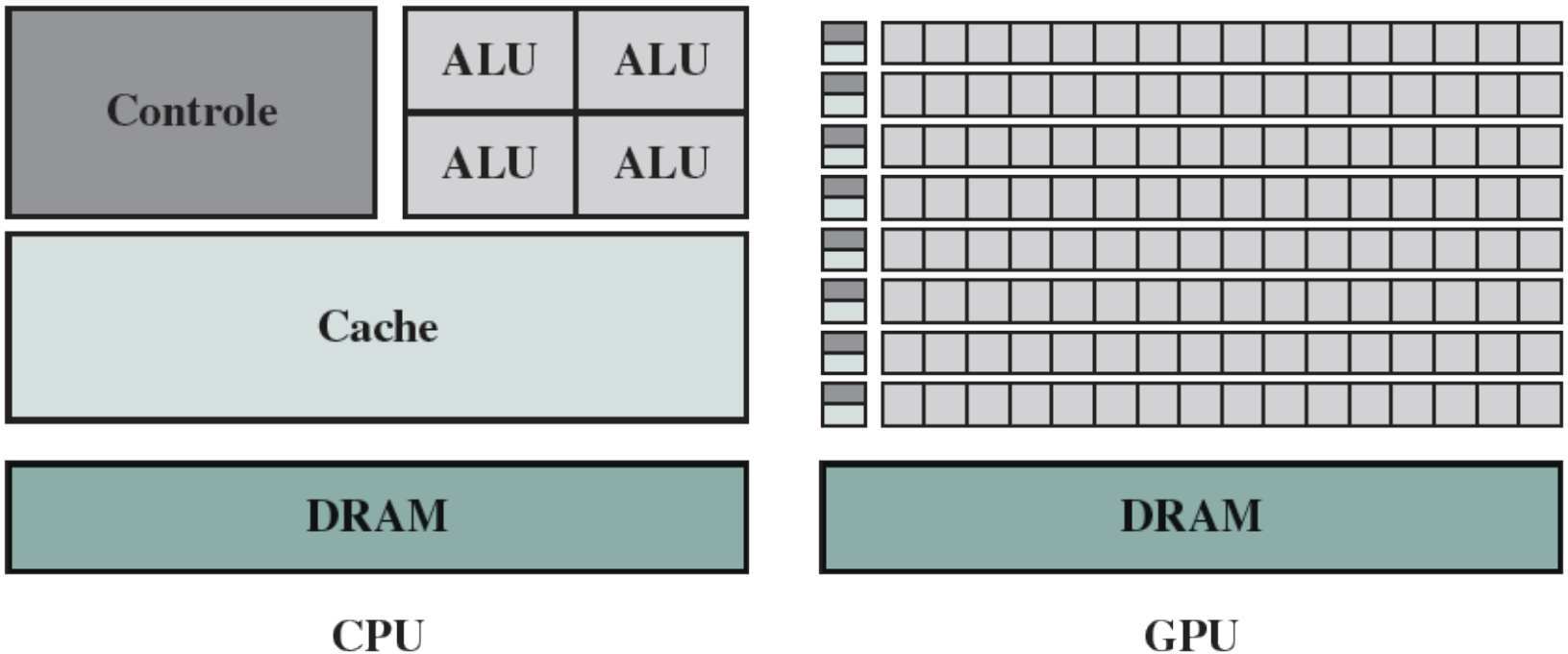
Noções básicas sobre a CUDA

- Termos da CUDA para mapeamento de equivalência de componentes de hardware de GPU:

Termo de CUDA	Definição	Componente equivalente de hardware de GPU
Kernel	Código paralelo na forma de uma função a ser executada na GPU	Não se aplica
Thread	Uma instância do kernel na GPU	*Core* processador de uma GPU/CUDA
Bloco	Um grupo de *threads* atribuído a um MS em particular	Multiprocessador de CUDA (MS)
Rede	A GPU	GPU

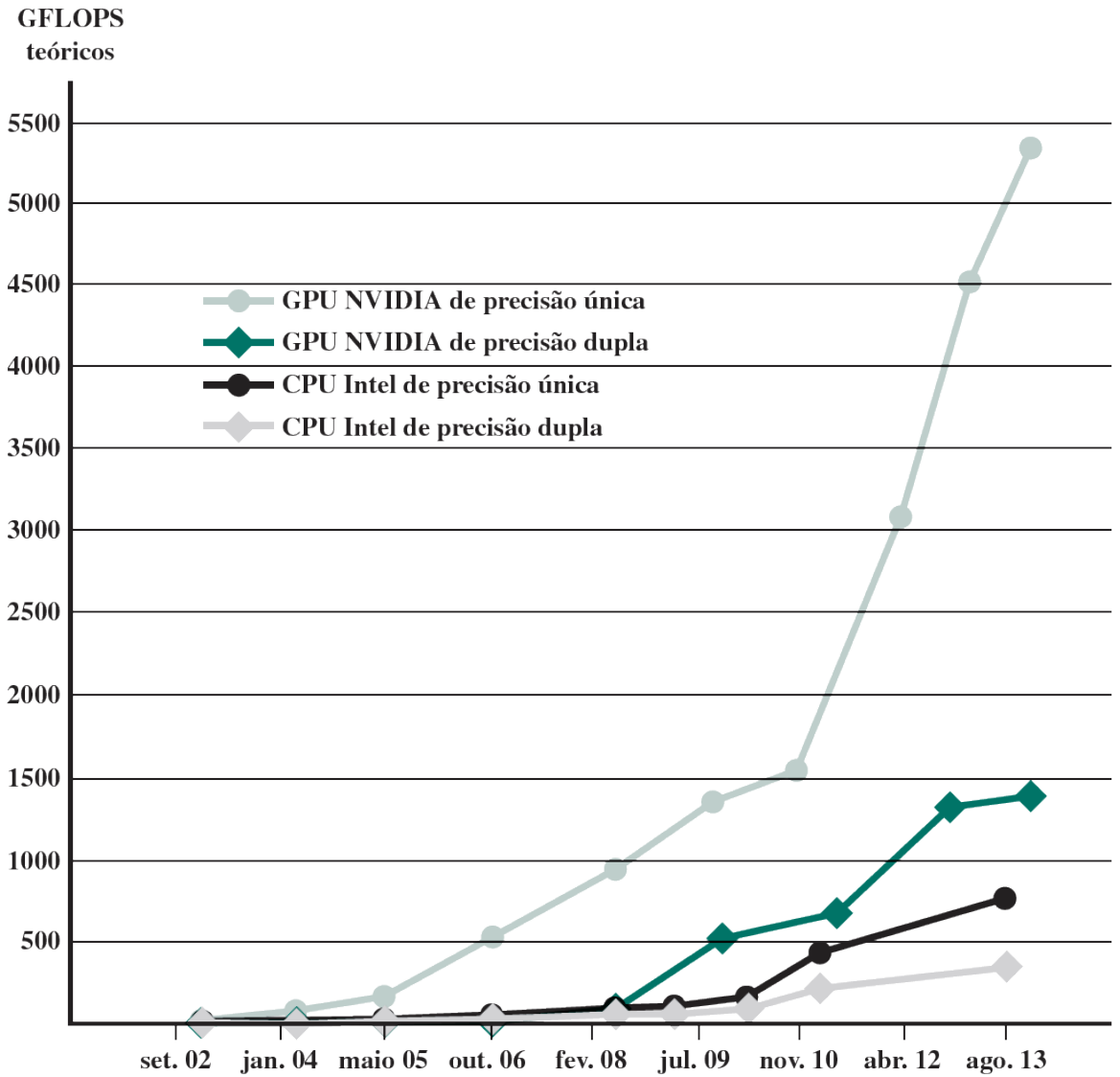
GPU versus CPU

- Dedicação de transistores/área de silício de uma CPU versus uma GPU:



GPU versus CPU

- Operações de ponto flutuante por segundo para CPU e GPU:

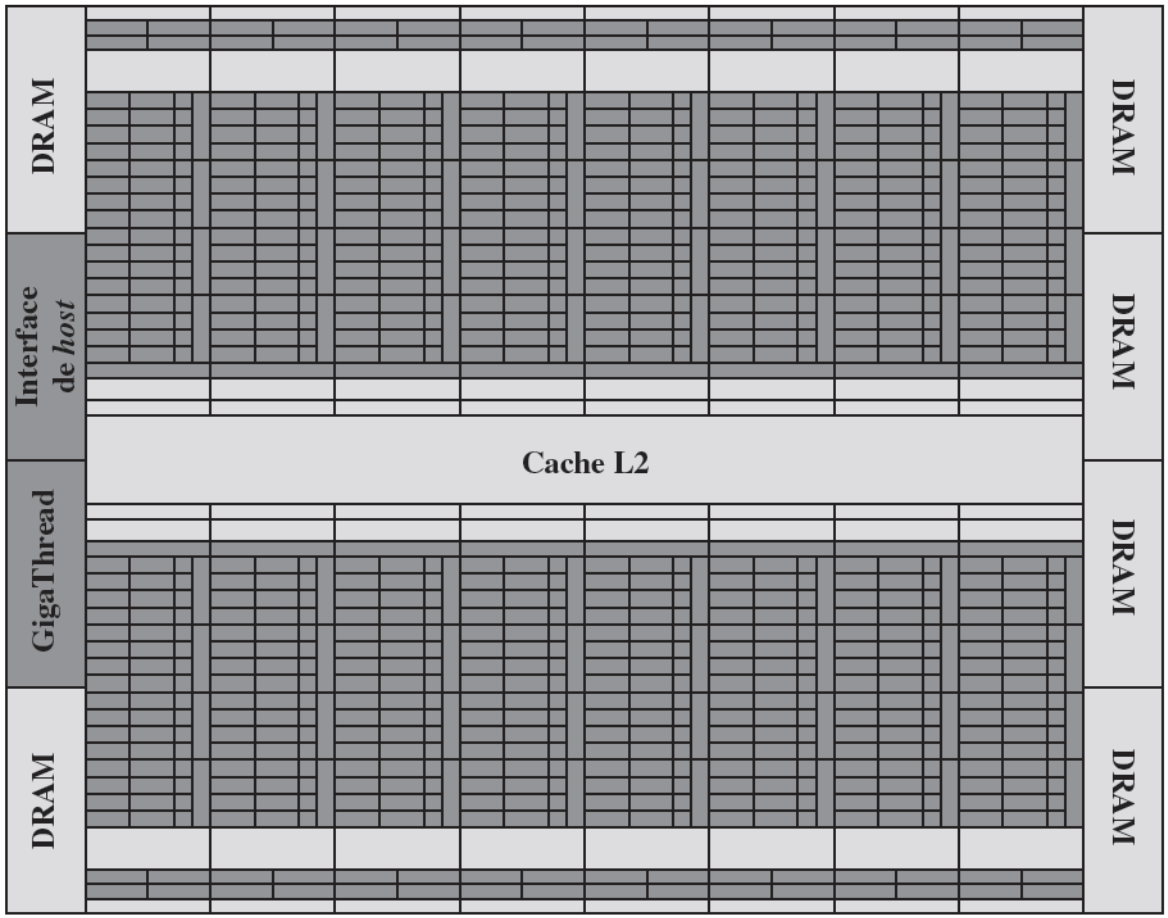


Visão geral da arquitetura de uma GPU

- A evolução histórica da arquitetura de uma GPU pode ser dividida em três fases ou eras principais:
  - A primeira fase cobriria os anos iniciais da arquitetura da GPU (do início da década de 1980 ao final da década de 1990).
  - A segunda fase abrangeria a modificação iterativa da arquitetura resultante da GPU, fase 1, de um pipeline de hardware especializado e fixo para um processador totalmente programável.
  - A terceira fase abrange como a arquitetura GPU/GPGPU torna um coprocessador SIMD altamente paralelizado e acessível.

Visão geral da arquitetura de uma GPU

- Arquitetura Fermi da NVIDIA:



Multiprocessador de Streaming (MS)

- É um componente importante da GPU que executa operações em paralelo;
- Permite que a GPU realize processamento paralelo para tarefas gráficas e computacionais;
- Cada multiprocessador de streaming é composto por múltiplos núcleos de processamento (CUDA cores);
- GPU modernas são compostas por vários multiprocessadores de streaming (MS);
- Cada MS pode executar várias tarefas em paralelo;
- Projetado para acelerar tarefas que podem ser paralelizadas;
  - Simulações;
  - Renderização 3D;
  - Processamento de imagens;
  - Aprendizado de máquina;
  - Deep Learning.

Detalhes da arquitetura do multiprocessador de streaming

- Os componentes básicos para um único MS são:
  - Cores de processadores de GPU (total de 32 cores).
  - Escalonador warp e porta de despacho.
    - Warp é um grupo de threads (32 threads - NVIDIA);
    - Gerencia a execução dos warps;
    - Quais warps serão executados;
    - Ciclo de clock;
    - Otimizar o processamento paralelo;
  - Dezesseis unidades de carga/armazenamento.
  - Quatro unidades de funções especiais (SFUs).
  - Registradores 32k $\times$ 32 bits.
  - Memória compartilhada e cache L1.

Detalhes da arquitetura do multiprocessador de streaming

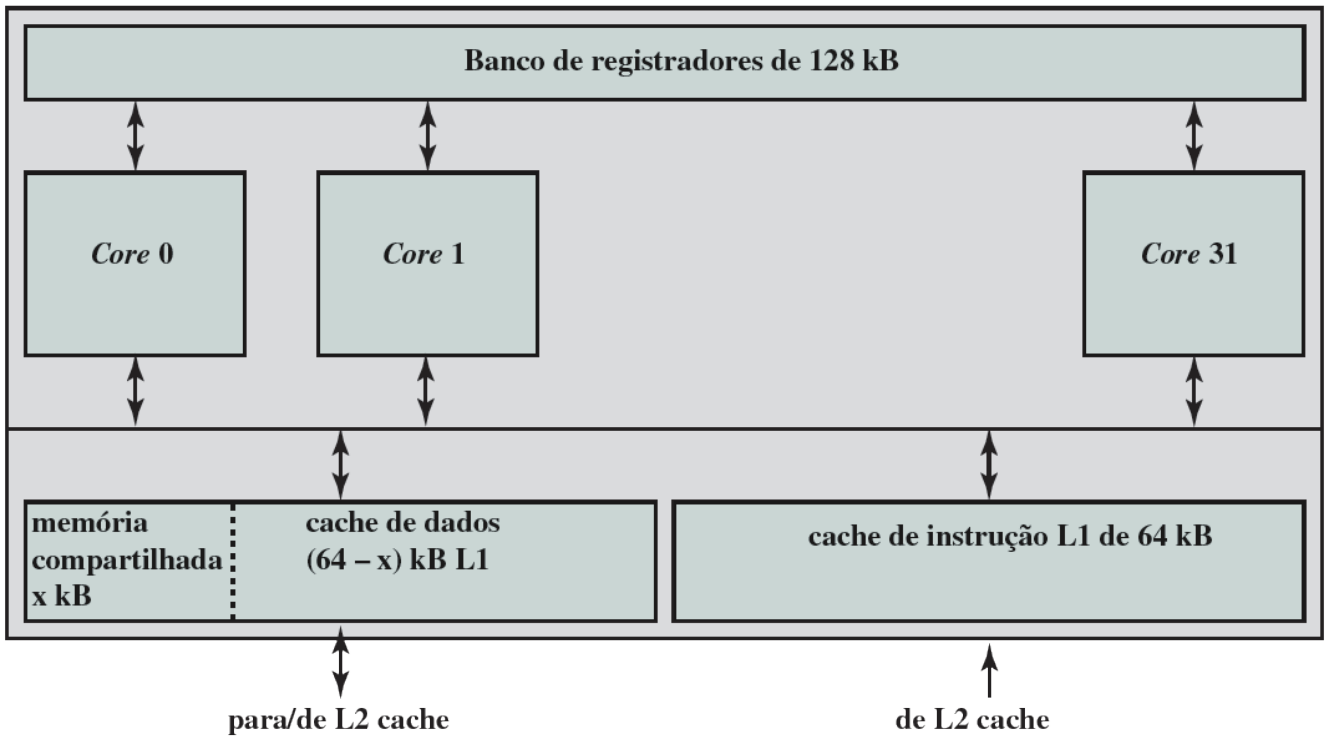
- **ESCALONADOR WARP DUPLO** Dois warps podem ser escalonados e executados em paralelo dentro de um único ciclo de clock.
- **CORES CUDA** Existe um total de 32 cores CUDA dedicados a cada MS na arquitetura Fermi. Cada core CUDA tem dois pipelines ou caminhos de dados separados: uma unidade pipeline de inteiro (INT) e uma unidade de pipeline de ponto flutuante (PF).

Detalhes da arquitetura do multiprocessador de streaming

- **UNIDADES DE FUNÇÃO ESPECIAIS** Cada MS tem quatro SFUs. A SFU apresenta operações transcendentais, como cosseno, seno, recíproca e raiz quadrada, em apenas um ciclo de clock.
- **UNIDADES DE CARGA E ARMAZENAMENTO** Cada uma das 16 unidades de carga e armazenamento do MS calcula a fonte e o endereço de destino para um único thread por ciclo de clock.
- **REGISTRADORES , MEMÓRIA COMPARTILHADA E CACHE L1** Cada MS tem seu próprio conjunto (on-chip) dedicado de registradores e de memória compartilhada/bloco de cache L1.

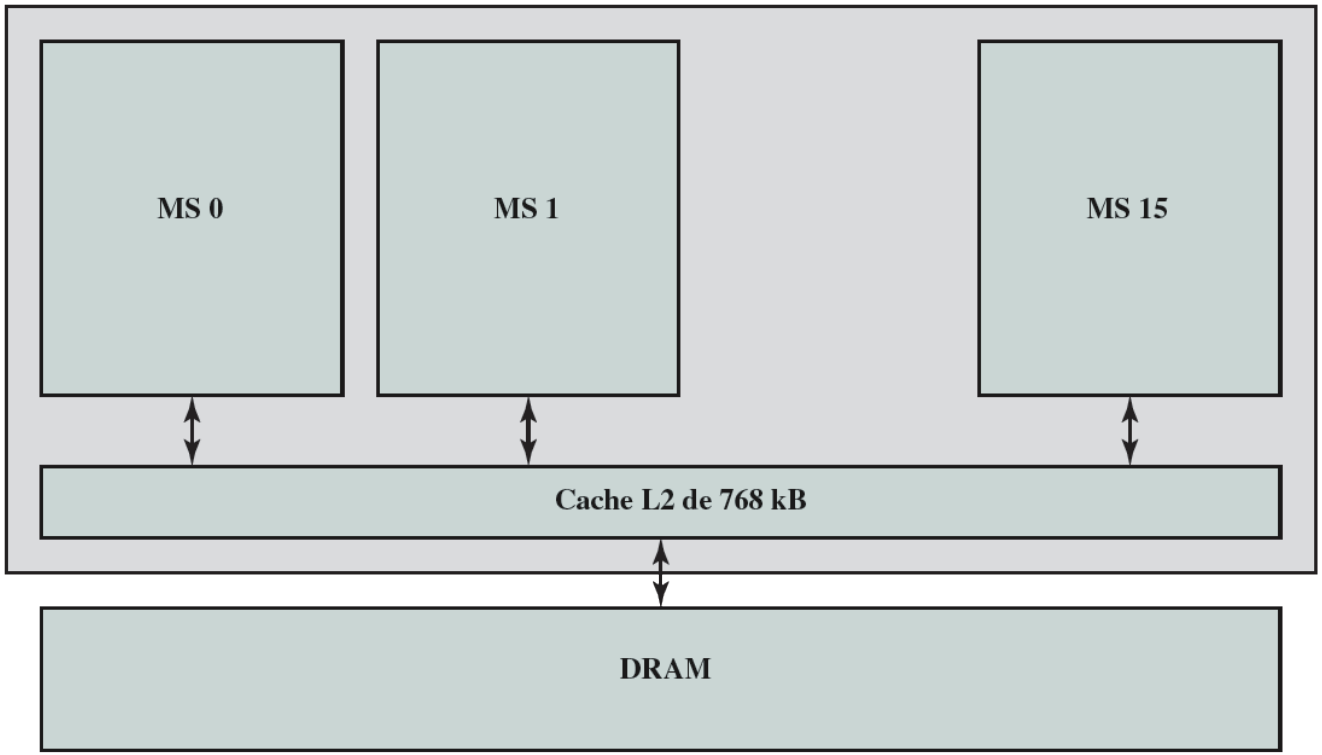
Detalhes da arquitetura do multiprocessador de streaming

- Arquitetura da memória do MS:



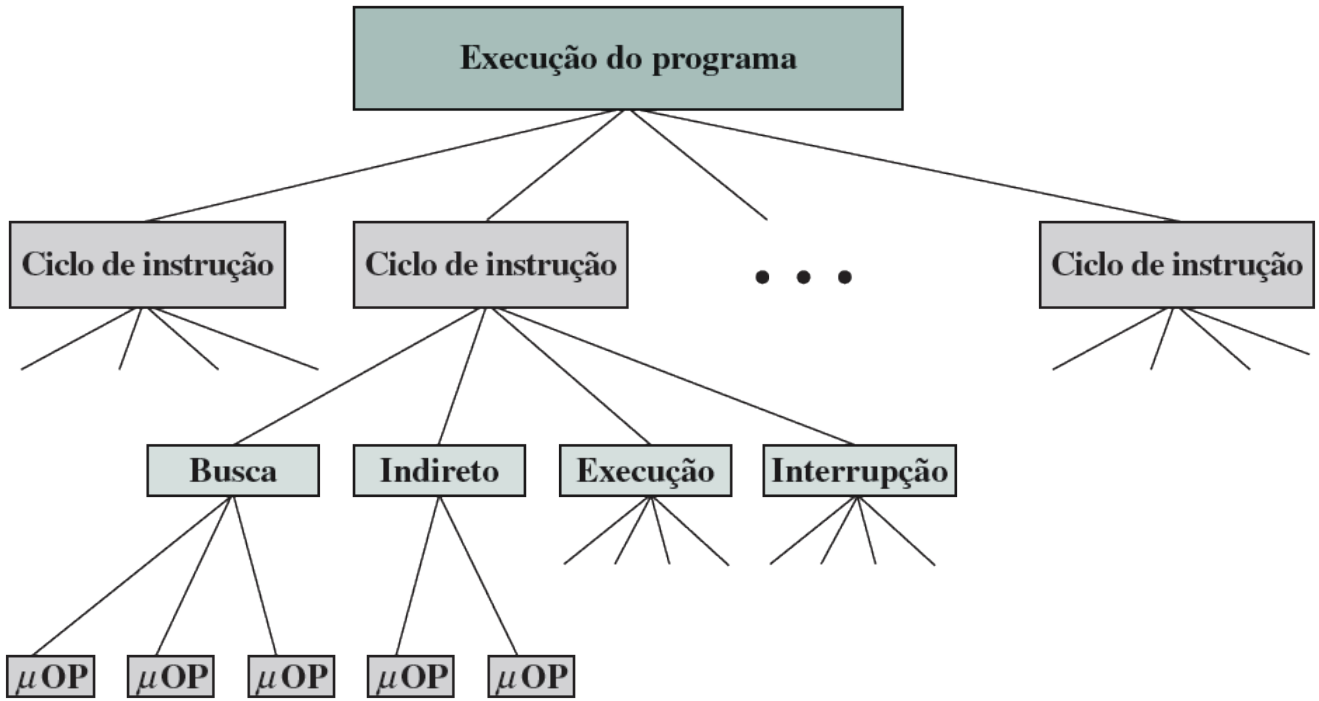
Detalhes da arquitetura do multiprocessador de streaming

- Arquitetura completa da memória:



Micro-operações

- Elementos que constituem uma execução de programa:

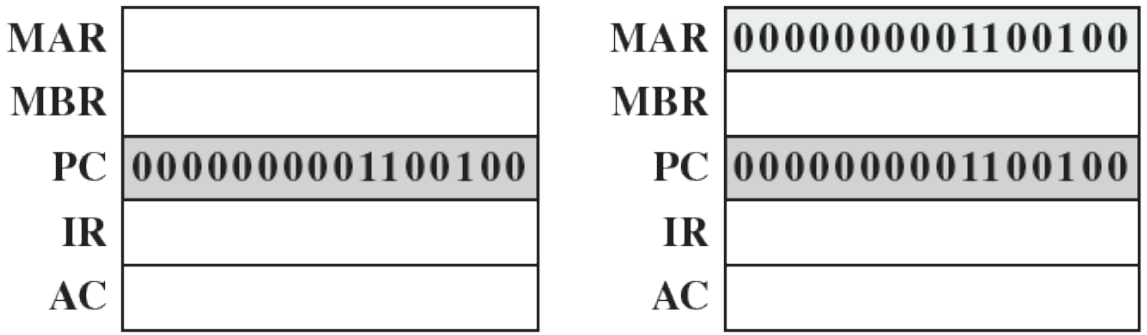


Ciclo de busca

- Quatro registradores estão envolvidos:
  - **Registrador de endereço de memória** (MAR): especifica o endereço na memória para uma operação de leitura ou escrita.
  - **Registrador de buffer de memória** (MBR): contém o valor a ser guardado na memória ou o último valor lido da memória.
  - **Contador de programa** (PC): guarda o endereço da próxima instrução a ser lida.
  - **Registrador de instrução** (IR): guarda a última instrução lida.

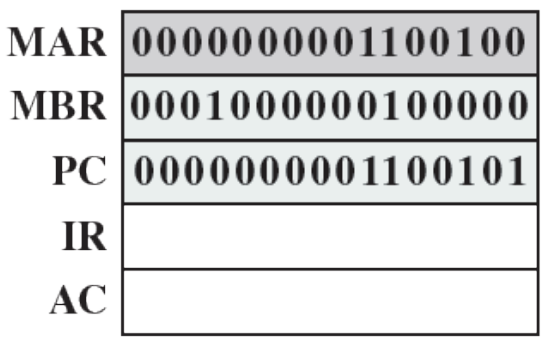
Ciclo de busca

- Sequência de eventos, ciclo de busca:

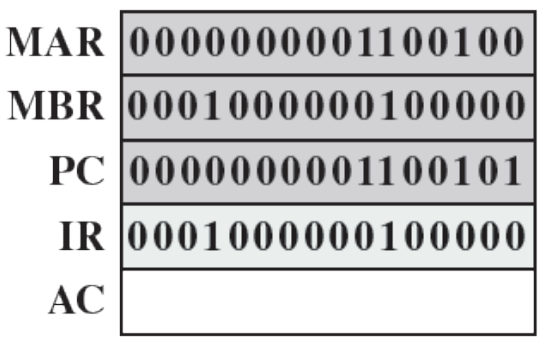


(a) Início (antes de t_1)

(b) Depois do primeiro passo



(c) Depois do segundo passo



(d) Depois do terceiro passo

Ciclo de busca

- A notação (t1, t2, t3) representa unidades de tempo sucessivas.
- Dessa forma temos:
  - Primeira unidade de tempo: move conteúdo de PC para MAR.
  - Segunda unidade de tempo: move conteúdo da posição de memória especificada por MAR para MBR. Incrementado por I o conteúdo de PC.
  - Terceira unidade de tempo: move o conteúdo de MBR para IR.

Ciclo indireto

- Uma vez lida a instrução, o próximo passo é buscar os operandos fontes.
- Se a instrução especifica um endereço indireto, então um ciclo indireto deve preceder o ciclo de execução.
- O fluxo de dados inclui as seguintes micro-operações:

```
t1: MAR   ←  (IR(Endereço))
t2: MBR   ←  Memória
t3: IR(Endereço)  ←  (MBR(Endereço))
```

Ciclo de interrupção

- Ao completar o ciclo de execução, um teste é feito para determinar se houve qualquer interrupção habilitada.
- Se sim, ocorre o ciclo de interrupção.
- A natureza desse ciclo varia muito de uma máquina para outra.
- Temos:

```
t1: MBR   ←  (PC)
t2: MAR   ←  Endereço_Salvar
    PC    ←  Endereço_Rotina
t3: Memória  ←  (MBR)
```

Ciclo de execução

- Por causa da variedade de opcodes, existe uma série de diferentes sequências de micro-operações que podem ocorrer.
- A unidade de controle examina o opcode e gera uma sequência de micro-operações com base no valor do opcode.
- Isso é chamado de decodificação da instrução.
- Primeiro, considere uma instrução de soma:
  ADD R1, X
- A qual adiciona o conteúdo da posição X ao registrador R1.

Ciclo de execução

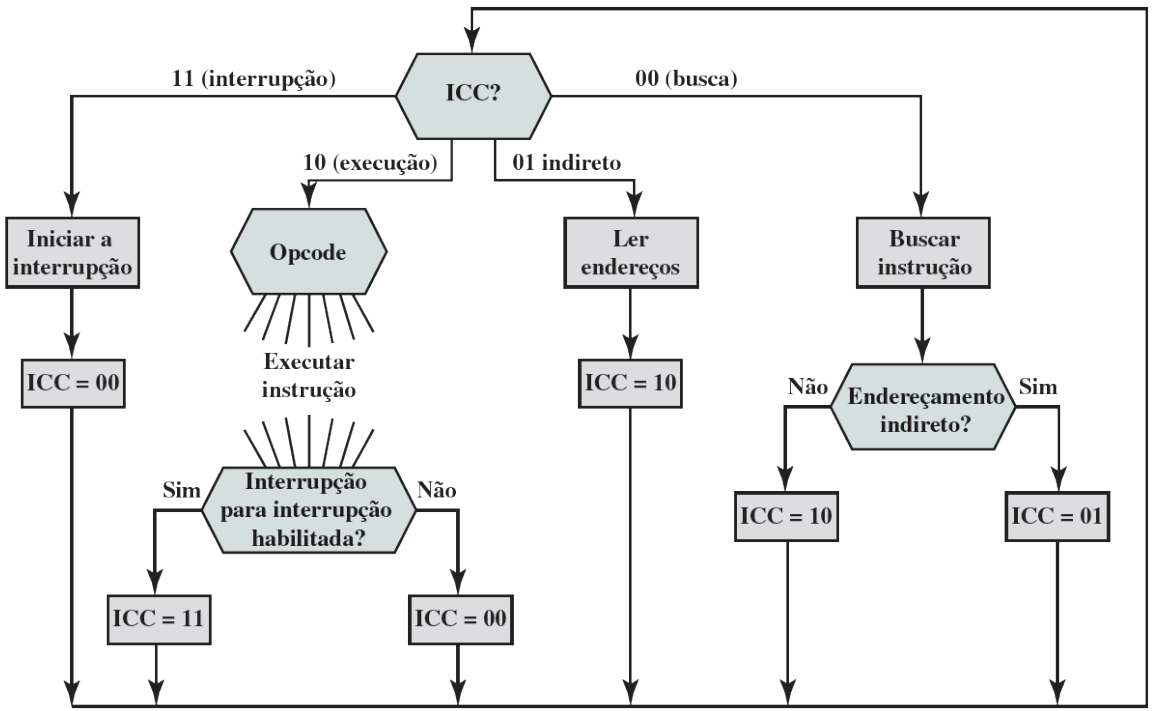
- A seguinte sequência de micro-operações pode ocorrer:

$$t_1: \text{MAR} \leftarrow (\text{IR(endereço)})$$
$$t_2: \text{MBR} \leftarrow \text{Memória}$$
$$t_3: \text{R1} \leftarrow (\text{R1}) + (\text{MBR})$$

- Começamos com IR contendo a instrução ADD.
- No primeiro passo, a parte do endereço de IR é carregada em MAR. Então, a posição de memória referenciada é lida.
- Finalmente, os conteúdos de R1 e MBR são adicionados pela ALU.

Ciclo de instrução

- Fluxograma do ciclo de instrução:



Controle do processador

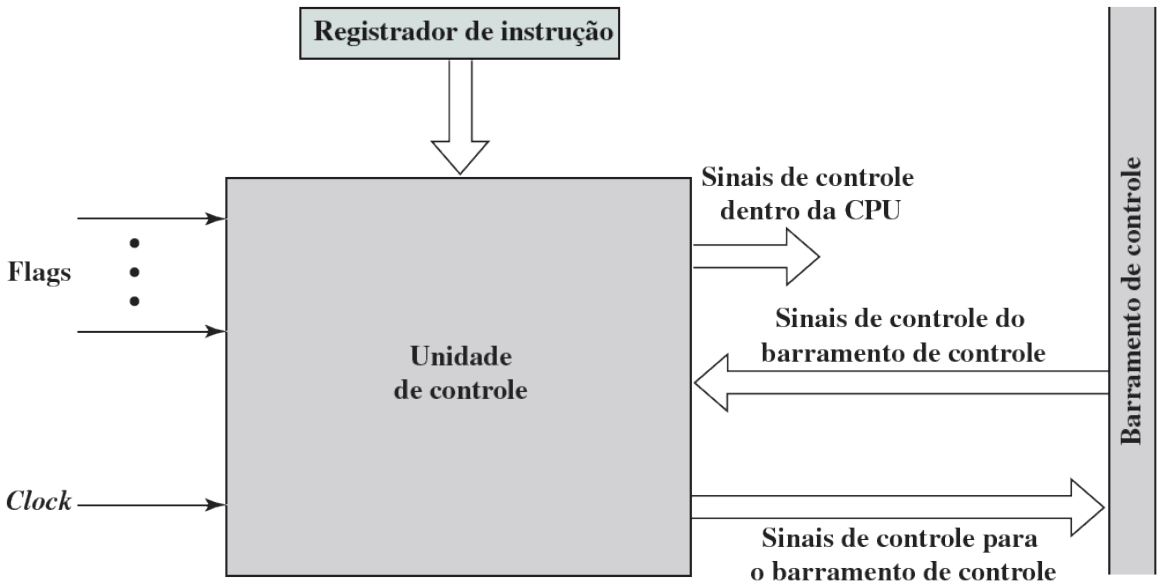
- Uma definição dos requisitos funcionais é a base para o projeto e a implementação da unidade de controle.
- Com essa informação em mãos, o próximo processo de três passos leva a uma caracterização da unidade de controle:
  - Definir elementos básicos do processador.
  - Descrever as micro-operações que o processador executa.
  - Determinar as funções que a unidade de controle deve realizar para fazer com que as micro-operações sejam executadas.

Controle do processador

- A unidade de controle desempenha duas tarefas básicas:
  - **Sequenciamento**: a unidade de controle faz com que o processador siga uma série de micro-operações na sequência correta, com base no programa que está sendo executado.
  - **Execução**: faz cada micro-operação ser executada.
- A chave para o funcionamento da unidade de controle é o uso de sinais de controle.

Sinais de controle

- Diagrama de bloco da unidade de controle:



Exemplo de sinais de controle

- A cada ciclo de clock, a unidade de controle lê todas as suas entradas e emite um conjunto de sinais de controle.
- Os sinais de controle vão para três destinos diferentes:
  - **Caminhos de dados:** a unidade de controle controla o fluxo interno de dados.
  - **ALU:** a unidade de controle controla a operação da ALU por meio de um conjunto de sinais de controle.
  - **Barramento de sistema:** a unidade de controle envia sinais de controle para filas de controle do barramento de sistema.

Bibliografia Básica

- STALLINGS, W. **Arquitetura e Organização de Computadores**. 10 ed. São Paulo: Pearson, 2017;
- TANENBAUM, A. S. **Organização Estruturada de Computadores**. 5 ed. Pearson 2007;
- HENNESY, J. PATTERSON, D. **Organização e Projeto de Computadores**. 3 ed. Editora Campus, 2005.

Obrigado!
Dúvidas?

Guilherme Henrique de Souza Nakahata

guilhermenakahata@gmail.com

https://github.com/GuilhermeNakahata/UNESPAR-2024