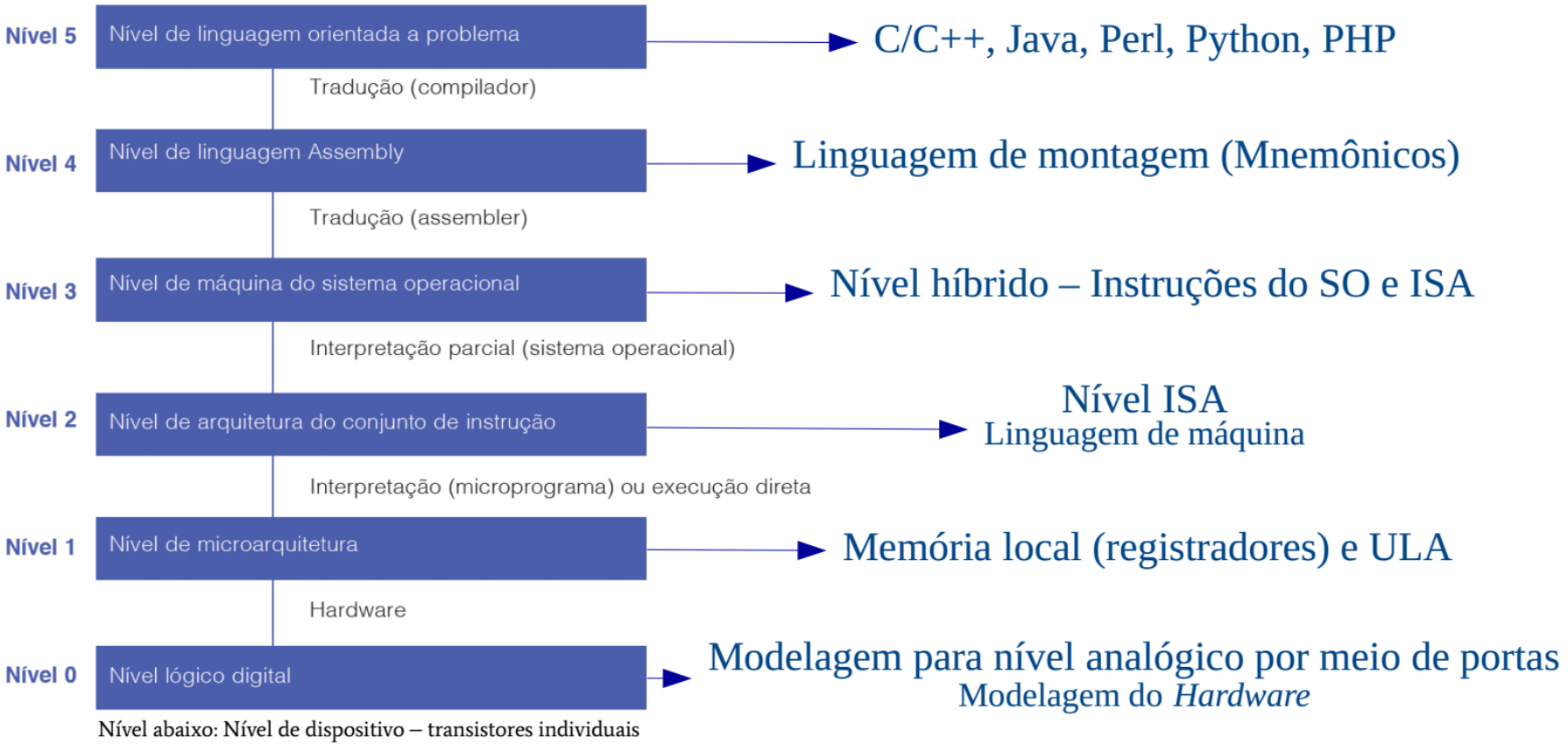
Arquitetura e Organização de computadores

ENGENHARIA DA COMPUTAÇÃO – UFC/SOBRAL

Prof. Danilo Alves

danilo.alves@alu.ufc.br

Máquina de Vários Níveis



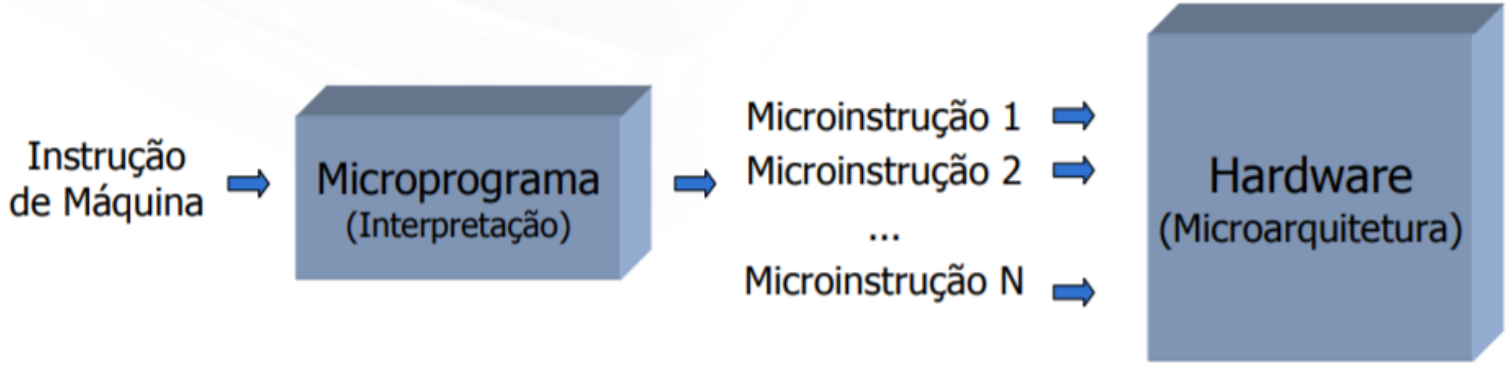
Nível abaixo: Nível de dispositivo – transistores individuais

Nível da Microarquitetura

- Função: implementar a camada ISA
  - O projeto do nível de microarquitetura depende do conjunto de instruções no nível ISA
- Inicia-se o conceito de programa como uma sequência de instruções a serem executadas diretamente pelos circuitos eletrônicos.
- A maior parte destas instruções envolve a movimentação de dados através do **caminho de dados**, ou alguns testes simples.
- Em alguns computadores (ex. CISC), a operação do caminho de dados é controlada por um programa conhecido como **microprograma**.

Nível da Microarquitetura

- O **microprograma** é um interpretador cuja função geral é buscar, decodificar e executar instrução por instrução do nível ISA (instrução de máquina)
- Cada instrução de máquina é interpretada e pode dar origem à execução de muitas **microinstruções**.



Nível da Microarquitetura

- Microprograma
- Função: **buscar**, **decodificar** e **executar** as instruções (de máquina), uma a uma, usando o caminho de dados para a realização de uma tarefa.
- Exemplo: Execução de uma instrução de SOMA (ADD)
  - A instrução deve ser buscada na memória, seus operandos devem ser localizados e trazidos para os registradores, a soma deve ser calculada na ULA, e o resultado deve ser encaminhado para o lugar apropriado
- O microprograma é um conjunto de microinstruções.
- Ele é armazenado numa memória ROM do processador, chamada *control store*, ou memória de controle.
- Cada microinstrução especifica os sinais de controle necessários para controlar a microarquitetura.

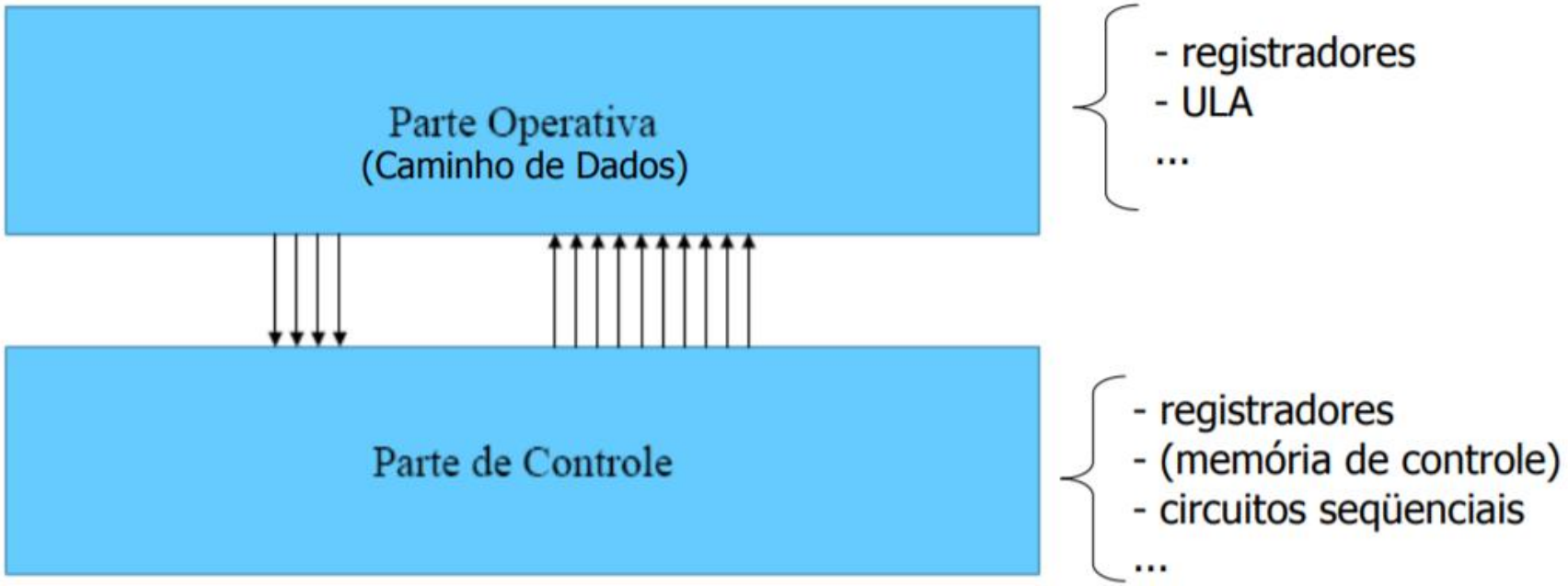
Nível da Microarquitetura

- Hardware (Microarquitetura)
- Em geral, enxerga-se:
  - Um conjunto de 8 a 32 registradores
  - O circuito da **ULA (Unidade Lógica e Aritmética)**
- Os registradores e a ULA são conectados para formar o **Caminho de Dados (Data Path)**, estrutura sobre a qual os dados fluem.
- A operação básica do caminho de dados
  - Seleção de um-dois registradores para que a ULA opere sobre eles

Nível da Microarquitetura

- Uma Microarquitetura é dividida em uma Parte Operativa e uma Parte de Controle
- Parte Operativa (ou Caminho de Dados)
  - Constituída de todos os componentes responsáveis pela execução das operações elementares sobre os dados (transformações nos dados)
- Parte de Controle
  - Constituída de circuitos sequenciais e/ou memória de microprograma que gera o controle ciclo-a-ciclo da parte operativa

Esquematizando a Microarquitetura



Um Exemplo de Microarquitetura

- Não existem os “princípios gerais de projeto de microarquitetura”.
  - Depende do conjunto de instruções do nível ISA
- Exemplo de microarquitetura adotado: **MIC1**
  - Um subconjunto da Máquina Virtual Java, em que há apenas instruções inteiras -> **IJVM**.
- Nossa microarquitetura conterá um microprograma
  - Considere que cada instrução no nível ISA corresponde a uma função a ser chamada pelo **programa principal** (microprograma)
  - Cada função, formada por uma sequência de microinstruções, define os passos a serem executados na microarquitetura

Um Exemplo de Microarquitetura

- Programa Principal:
- Composto por um *loop* que determina a função a ser chamada. E assim sucessivamente.

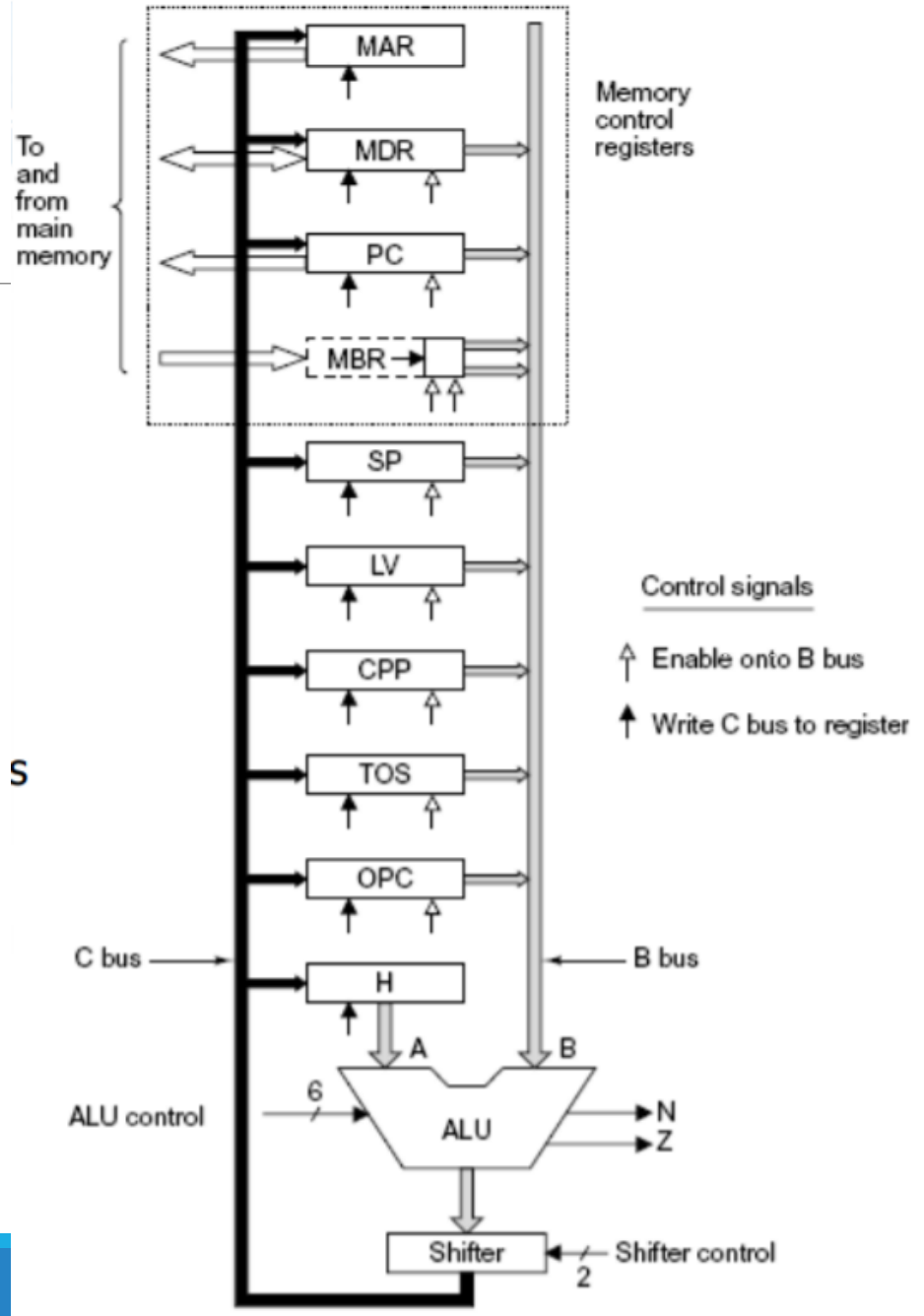
- O microprograma tem um conjunto de variáveis que representam o **estado** do computador
  - Cada função (que corresponde a uma instrução ISA) muda no mínimo uma das variáveis que formam o estado.
  - Na prática, essas variáveis correspondem aos registradores do sistema

Um Exemplo de Microarquitetura

- **Caminho de Dados** (Parte Operativa)
  - Contém a ULA e todas as suas entradas e saídas
    - 6 linhas de seleção (F0,F1, ENA, ENB, INVA, INC)
    - 2 entradas de dados (complemento de dois)
      - 1º : sempre do registrador H (Holding)
      - 2º : Qualquer um dos outros registradores, excluindo o H e MAR
  - Conjunto (array) de registradores
    - Esses registradores só podem ser acessados no nível da microarquitetura (ou seja, pelo microprograma).
- Em geral, esses registradores correspondem a variáveis (mesmo nome) utilizadas no nível ISA.

Um Exemplo de Microarquitetura

- **MAR:** Memory Address Register
- **MDR:** Memory Data Register
- **PC:** Program Counter
- **MBR:** Memory Buffer Register
- **SP:** Stack pointer (aponta para o topo da pilha)
- **LV:** Local Variables (Quadro de variáveis Locais)
- **CPP:** Constants Pool Pointer (Ponteiro para o Pool de Constantes)
- **TOS:** Top Of Stack (Guarda o conteúdo de memória apontado por SP)
- **OPC:** OPeration Code (Registrador temporário em algumas instruções)
- **H (holder)**



Um Exemplo de Microarquitetura

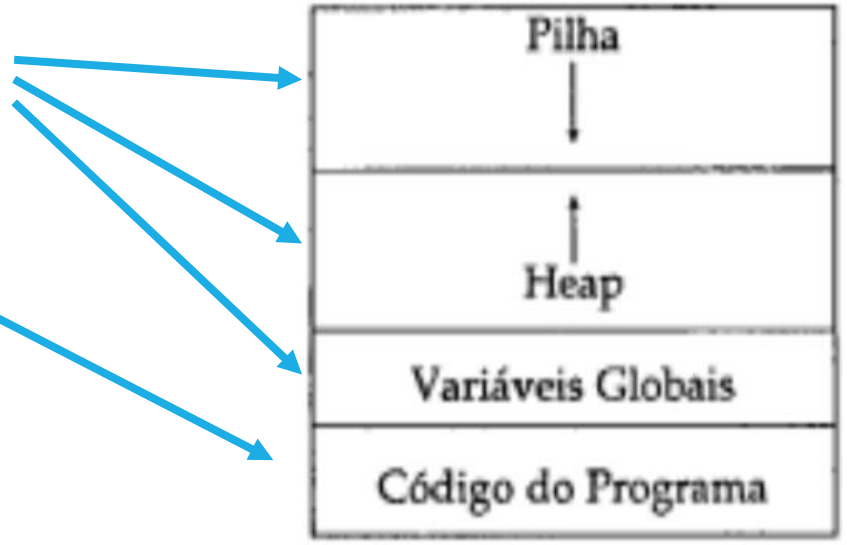
- **Operação da Memória**
- Na execução de um programa há, em geral, quatro regiões logicamente distintas na memória, que possuem funções específicas.
- Região de **Código do Programa**
- Região de **DADOS (Variáveis Globais)**
- **Pilha (Stack):** dentre os diversos usos se destacam o endereço de retorno das chamadas de função, argumentos para funções e variáveis locais, além de guardar o estado atual da CPU.
- **Heap:** geralmente uma região de memória livre que um programa pode usar para alocação dinâmica de memória (por exemplo)

Um Exemplo de Microarquitetura

- **Operação da Memória (cont.)**
- Uma microarquitetura executa instruções que são armazenadas na memória
- Instruções de um programa são executadas segundo uma ordem precisa
- Uma microarquitetura necessita saber, sempre, qual a próxima instrução a ser executada
  - **Necessidade de um registrador apontador de programa (PC)**
  - Necessidade de realização de operações sobre o conteúdo do PC, para atualização a cada instrução executada
- Existem **duas "portas de memória"** usadas pela máquina para se comunicar com a memória
  - Uma porta de 32 bits, que endereça palavras: controlada pelos registradores MAR e MDR
  - Uma porta de 8 bits, que endereça bytes: controlada pelo PC e MBR
    - Apenas de leitura

Um Exemplo de Microarquitetura

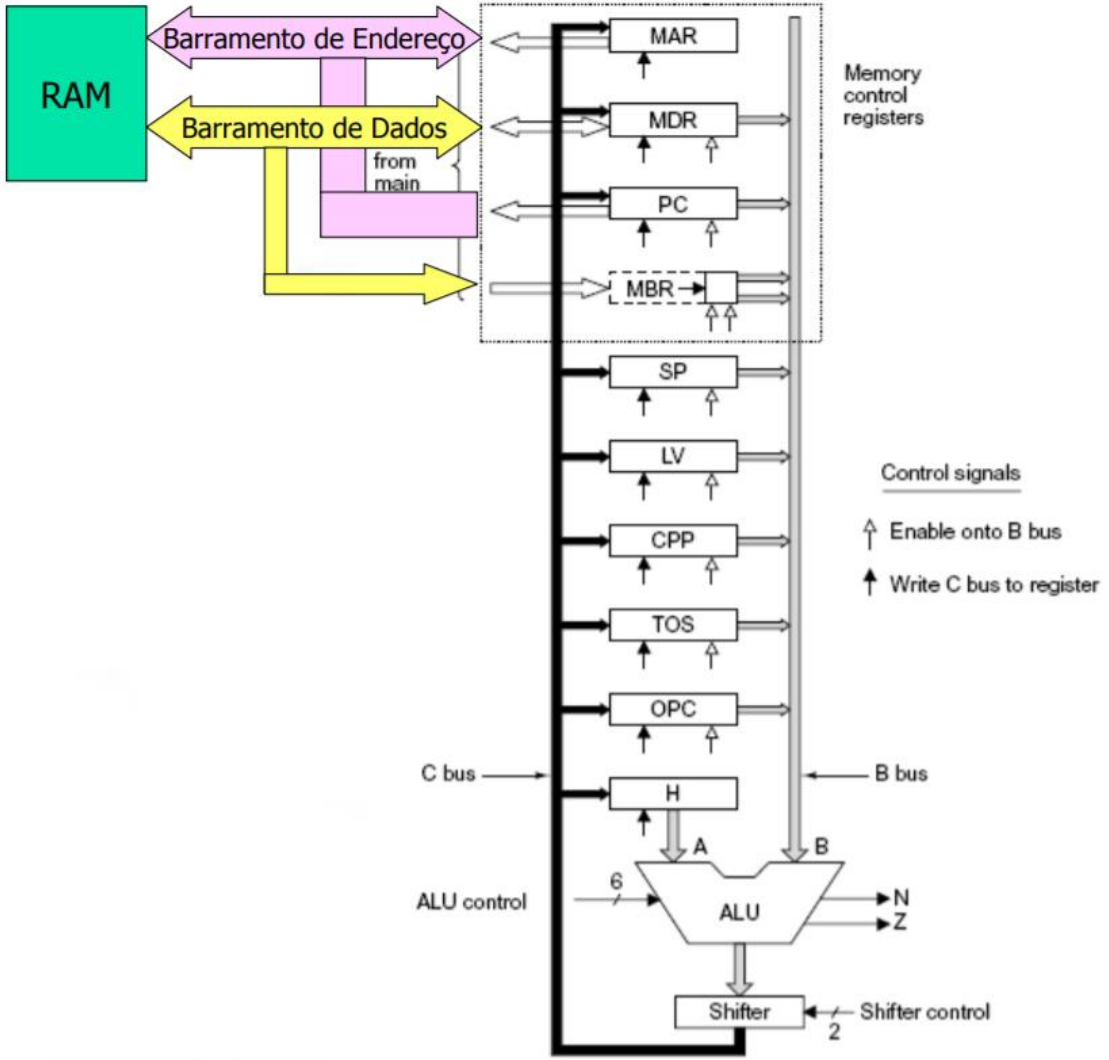
- O PC lê um byte da memória e o coloca nos 8 bits menos significativos do registrador MBR
- **MBR – Memory Buffer Register**
  - Uma porta para leitura de dados usada quando se deseja ler da memória dados de apenas 8 bits.
- Diferença na funcionalidade de MAR e PC
  - Eles são usados para referenciar duas partes diferentes da memória
- **MAR/MDR** é usada para ler/escrever palavras de dados pertencentes ao nível ISA
- **PC/MBR** é usada para ler o programa executável (programa constituído por um grupo de bytes).



Um Exemplo de Microarquitetura

- Operação da Memória (cont.)
- Transferência de Dados entre a MP e a CPU
  - É realizada com o uso de dois registradores especiais (para dados de 32 bits).
- **MAR – Memory Address Register**
  - **REM – Registrador de Endereços de Memória**
  - Armazena o endereço da memória, onde será lida ou gravada uma palavra.
  - Tamanho deve permitir acesso a todos os N endereços da memória
- **MDR – Memory Data Register**
  - **RDM – Registrador de Dados da Memória**
  - Armazena a representação da informação (palavra) a ser transferida.
  - Tamanho é, em geral, igual ao da palavra.

Um Exemplo de Microarquitetura



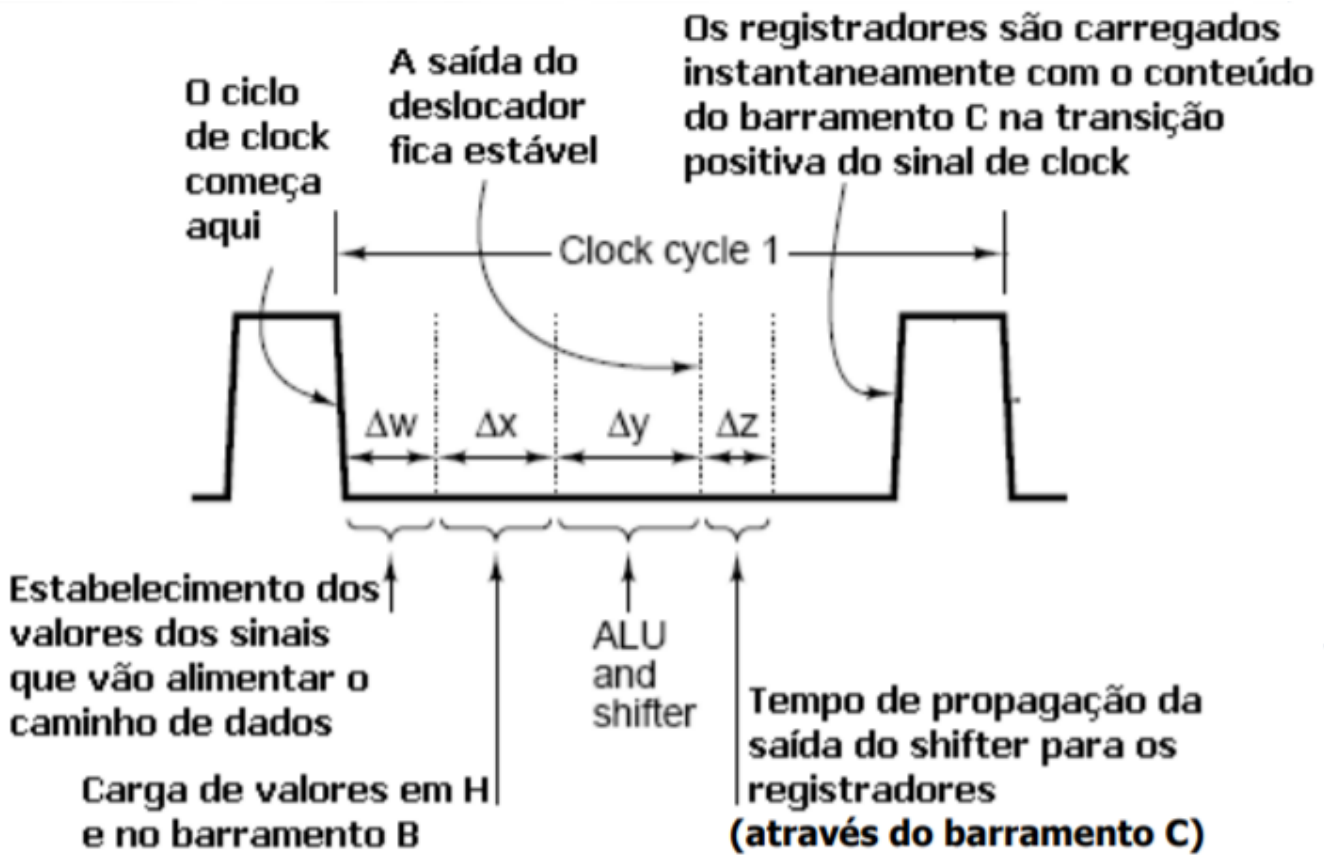
Um Exemplo de Microarquitetura

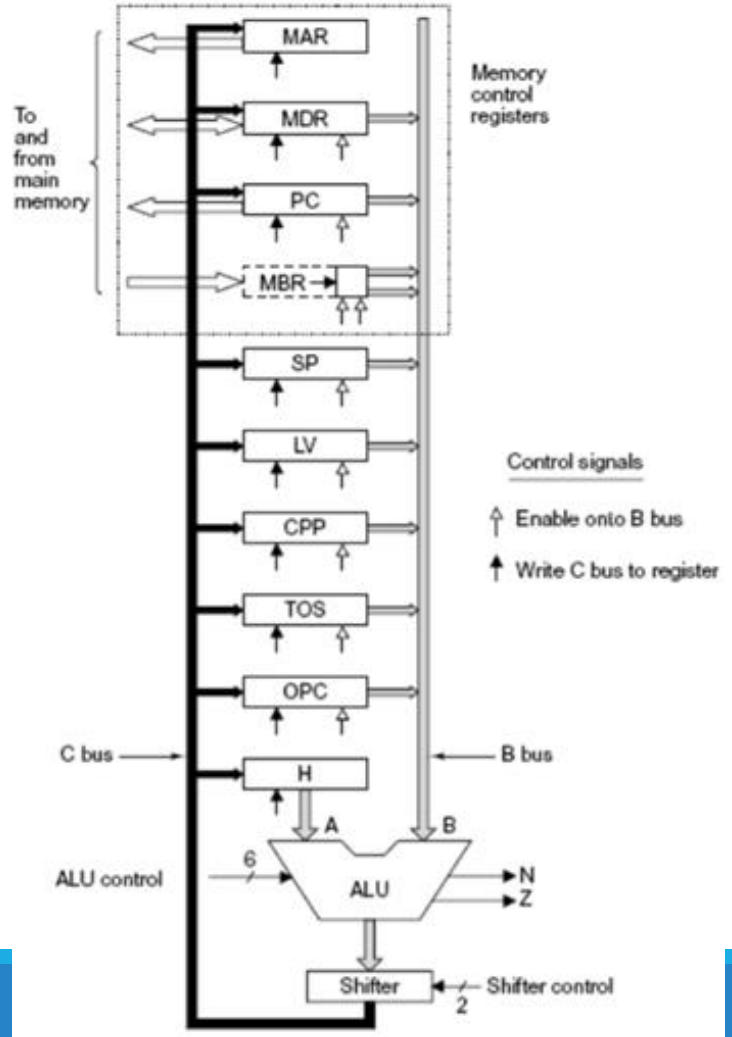
- **Leitura de um dado**
  - A CPU coloca em MAR, o endereço da posição cujo conteúdo deve ser lido
  - A CPU comanda uma leitura (sinal de controle para a memória - READ)
    - MAR -> barramento de endereço
  - O conteúdo (palavra) da posição do endereço contido em MAR, é então transferido para o MDR
    - barramento de dados -> MDR
- **Escrita de um dado**
  - A CPU coloca em MAR o endereço de memória onde a palavra será gravada, e em MDR a palavra a ser gravada.
  - A CPU comanda uma gravação (sinal WRITE)
    - MAR -> barramento de endereço
    - MDR -> barramento de dados
  - A palavra armazenada em MDR é então transferida para a posição de memória cujo endereço está em MAR.

Um Exemplo de Microarquitetura

- O MAR guarda o endereço de uma palavra de 32 bits (devido às necessidades da JVM... nível ISA)
  - MAR = 0 => palavra 0 => bytes 0-3
  - MAR = 1 => palavra 1 => bytes 4-7
  - MAR = 2 => palavra 2 => bytes 8-11
- Na implementação física, a memória é orientada a byte (cada byte é uma célula, cada célula tem um endereço único)
- Quando o MAR é colocado no barramento de endereço, seus 32 bits NÃO são mapeados diretamente nas linhas 0 a 31 do barramento

Temporização do Caminho de Dados





Temporização do Caminho de Dados

- O ciclo do caminho de dados pode ser dividido em subciclos
- O início do primeiro subciclo é marcado pela transição negativa do clock
  1. Os sinais de controle são ativados (Δw)
  2. O barramento B é carregado a partir dos registradores (Δx)
  3. A ALU e o shifter (deslocador) operam (Δy)
  4. O resultado da operação da ALU e do deslocador se propaga através do barramento C em direção aos registradores (Δz)

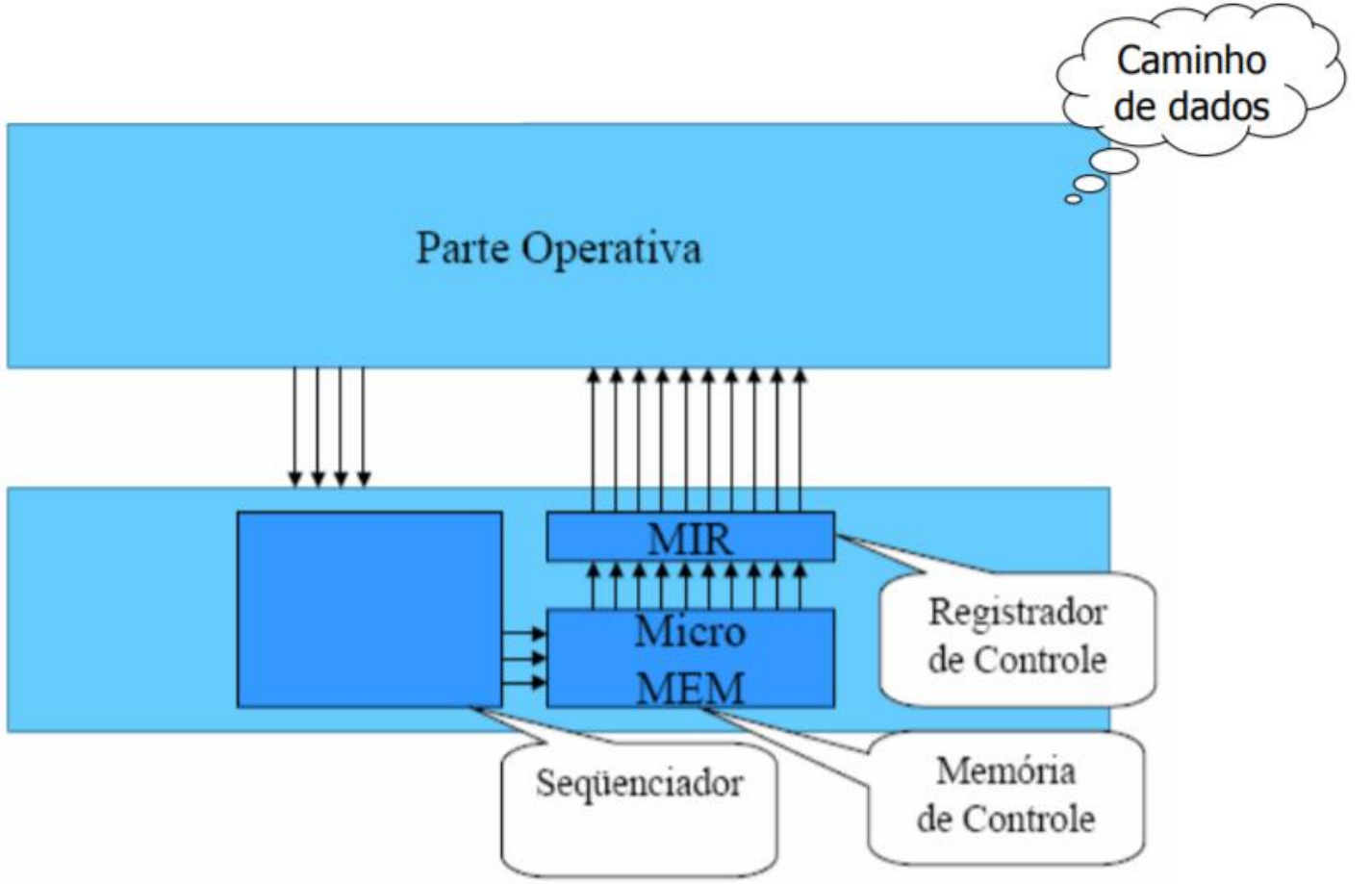
Temporização do Caminho de Dados

- O resultado é armazenado nos registradores na próxima transição positiva do clock
- Os subciclos são **implícitos**
  - As “fronteiras” entre os subciclos são determinadas pelos tempos de propagação inerentes aos circuitos envolvidos
- Os engenheiros do projeto devem assegurar que a transição positiva do clock, que marca a carga dos registradores, ocorra depois de $\Delta w+\Delta x+\Delta y+\Delta z$

Microarquitetura: Parte de Controle

- Microprogramação envolve:
- Memória para armazenar o microprograma
  - Memória de controle
- Código do microprograma
  - Microisntruções
- Sequenciamento de microinstruções
  - Corresponde à ordem de execução do microprograma
- Registrador de controle, contendo a microinstrução corrente, que deve ser executada
  - MIR: MicroInstruction Register
  - Seus bits alimentam os sinais de controle que operam o caminho de dados

Microarquitetura: Parte de Controle



Exemplo de Microarquitetura: Microinstruções

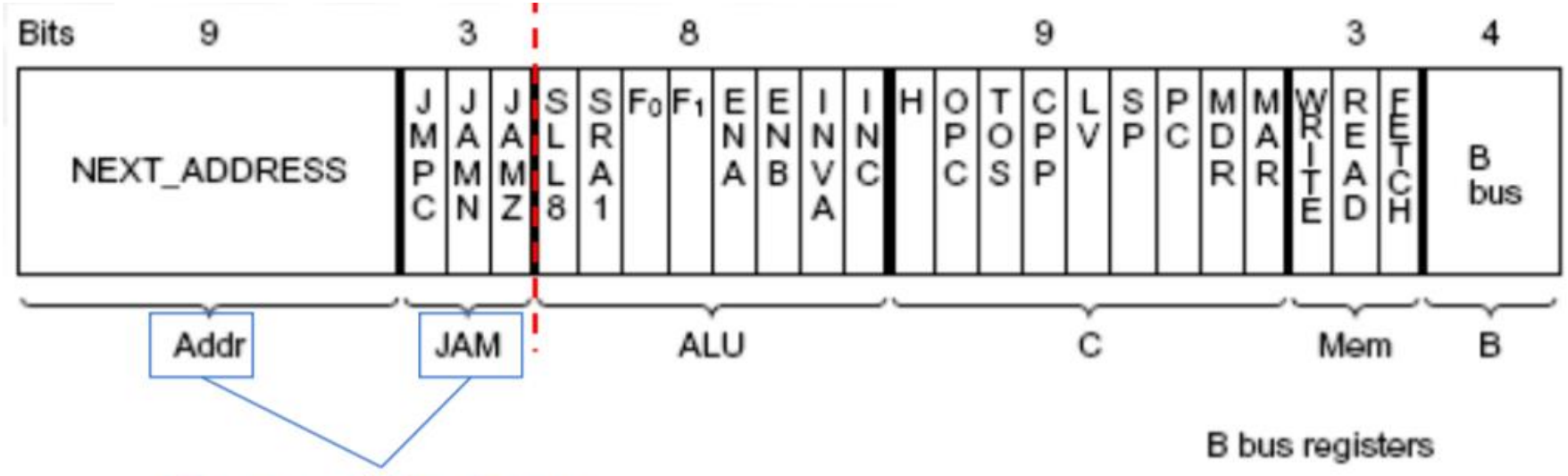
- Para controlar o caminho de dados anterior, precisa-se de 29 sinais, que podem ser divididos em grupos funcionais.
  - 9 sinais para controlar a escrita de dados do barramento C nos registradores
  - 9 sinais para controlar a habilitação da saída dos registradores para o barramento B, para compor as entradas da ULA.
  - 8 sinais para controlar as funções da ULA e do deslocador.
  - 2 sinais para indicar leitura/escrita da memória por intermédio de MAR/MDR
  - 1 sinal para indicar a busca na memória por intermédio de PC/MBR.
- Os valores desses 29 sinais de controle especificam as **operações** a serem executadas em um **ciclo do caminho de dados**.

Exemplo de Microarquitetura: Microinstruções

- Como reduzir o número de sinais?
  - Se por um lado, o valor do barramento C pode ser escrito em mais de um registrador, **apenas um** registrador poderá ter a saída habilitada para o barramento B!
  - Usando um decodificador, é possível codificar em apenas 4 bits o registrador que vai colocar seus dados no barramento B
    - Apenas 9 das 16 saídas do decodificador são usadas
- Agora só precisamos de 9 + **4** + 8 + 2 + 1 = 24 sinais

Exemplo de Microarquitetura: Microinstruções

- Formato possível de microinstrução no exemplo Mic-1

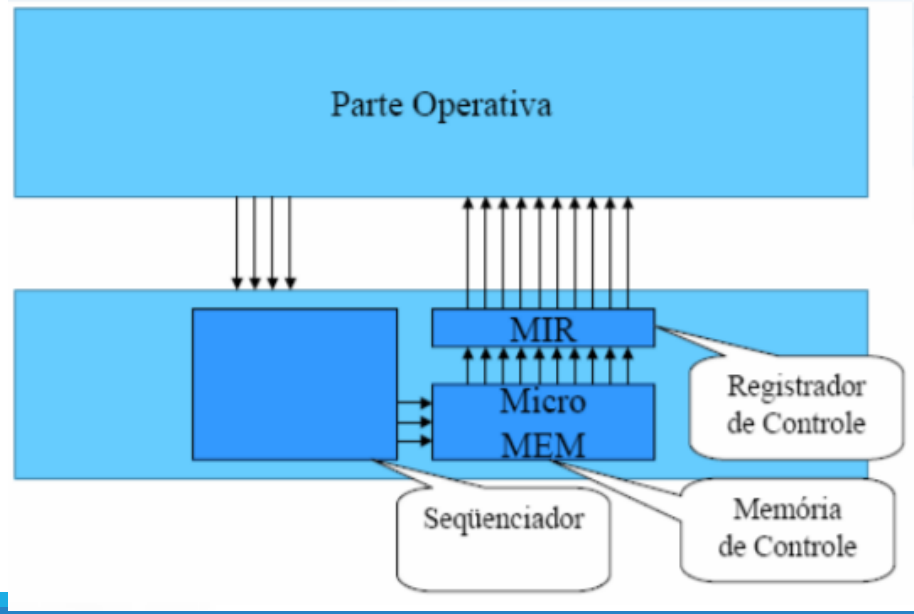


Exemplo de Microarquitetura: Microinstruções

- Em suma, os sinais da microinstrução são divididos:
  - Endereço (Addr): contém o endereço da próxima microinstrução a ser potencialmente ativada.
  - Desvio (JAM): determina como a próxima microinstrução será selecionada (usando ou não Addr).
  - ALU : especifica as funções da ULA e do deslocador.
  - C: seleciona qual ou quais registradores serão carregados com o valor que estiver no barramento C.
  - Memória (Mem): especifica as funções da memória.
  - B: seleciona a entrada do barramento B.

Controle Microprogramado – Mic-1

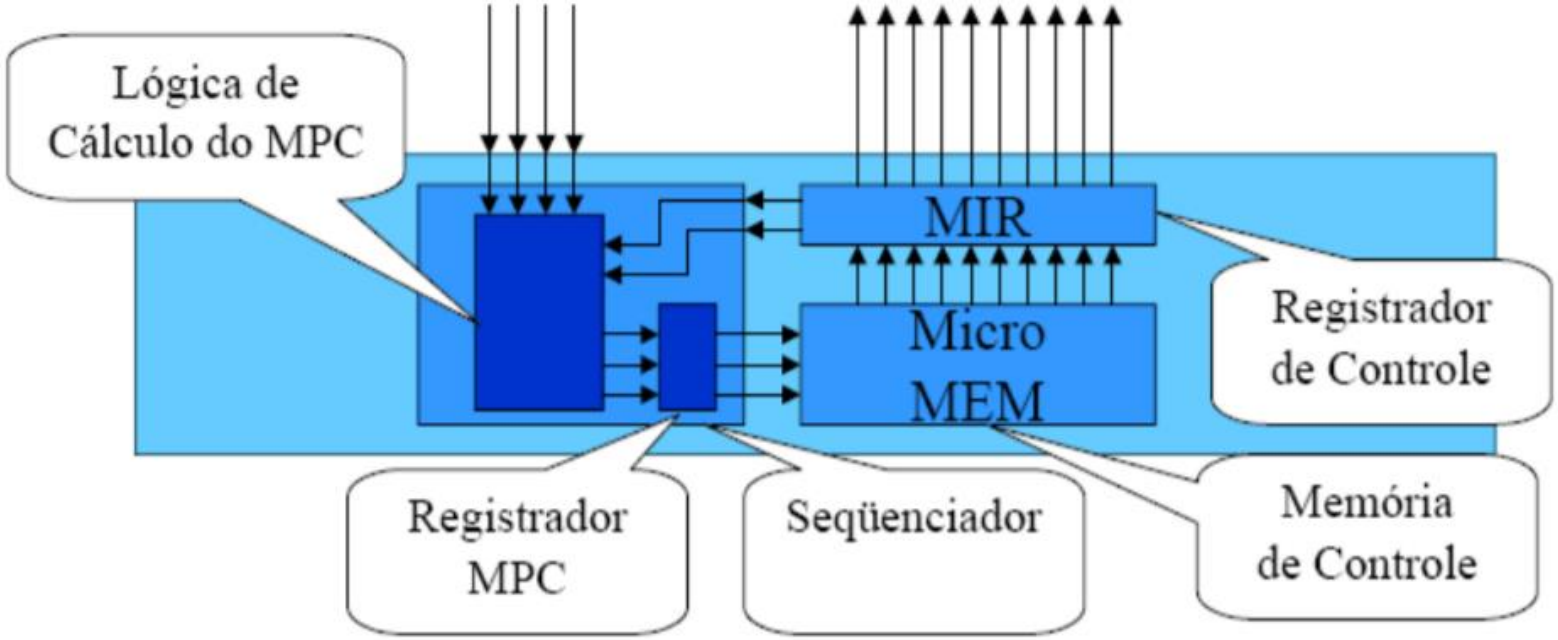
- A decisão sobre quais sinais de controle devem ser habilitados em cada ciclo é determinada por um **sequenciador**.
- **Sequenciador:** responsável pela execução de todos os passos necessários à execução de uma única instrução no nível ISA
  - Controla a sequência de microinstruções necessárias p/ executar cada instrução de máquina (ISA)



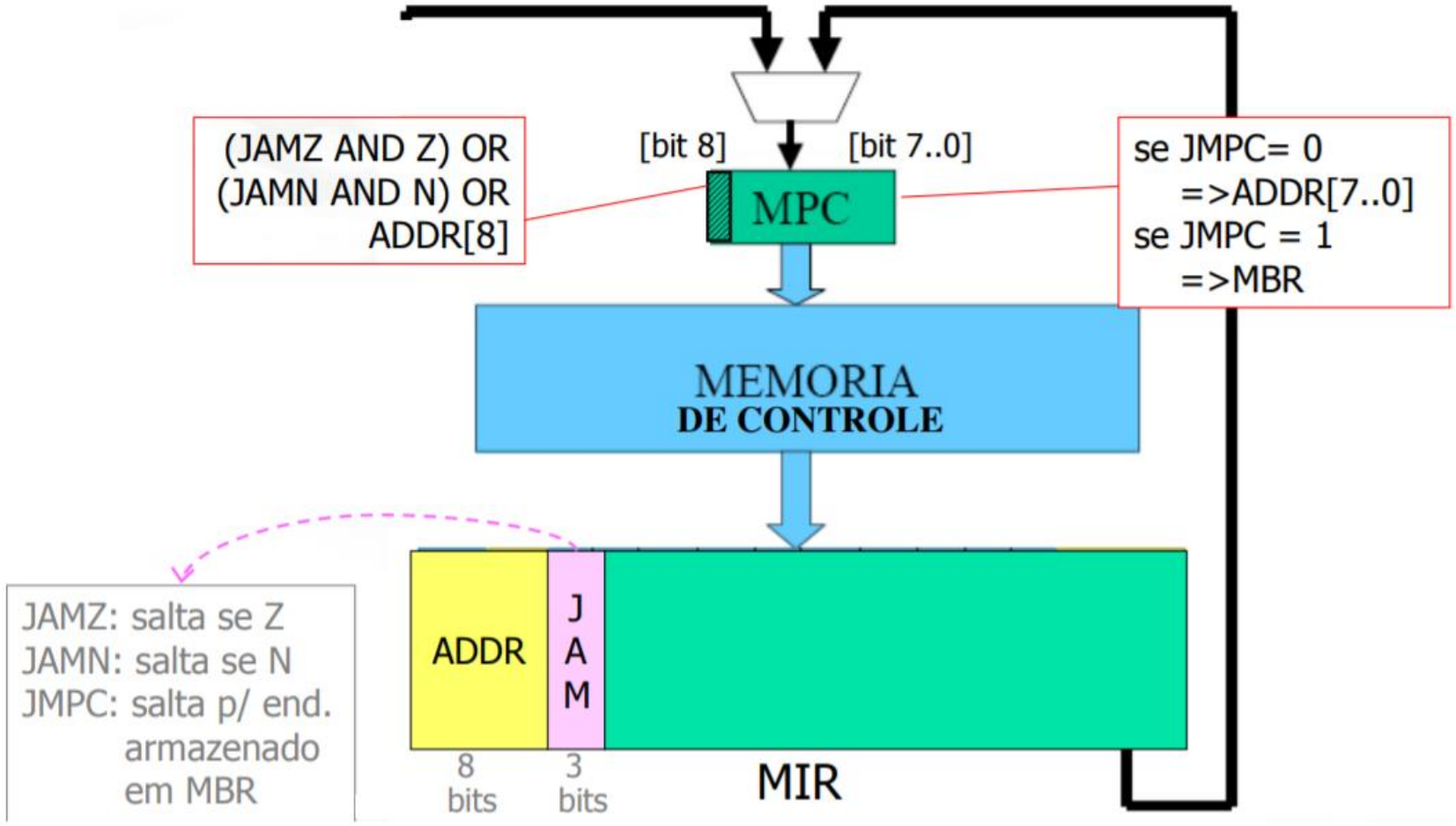
Controle Microprogramado – Mic-1

- Produz dois tipos de informação a cada ciclo:
- 1 – O estado de cada sinal de controle do sistema.
  - Na prática o sequenciador define a microinstrução que é colocada no MIR
- 2 – O endereço da microinstrução que será executada em seguida
  - O sequenciador determina o endereço da próxima microinstrução em função da microinstrução corrente (que encontra-se no MIR)
  - Esse endereço uma vez determinadado, é armazenado em um registrador, o **MPC** (MicroProgram Counter)
  - Desta forma, o MPC contém o endereço da próxima **microinstrução** a ser executada (localizada na **memória de controle**)

Controle Microprogramado – Mic-1

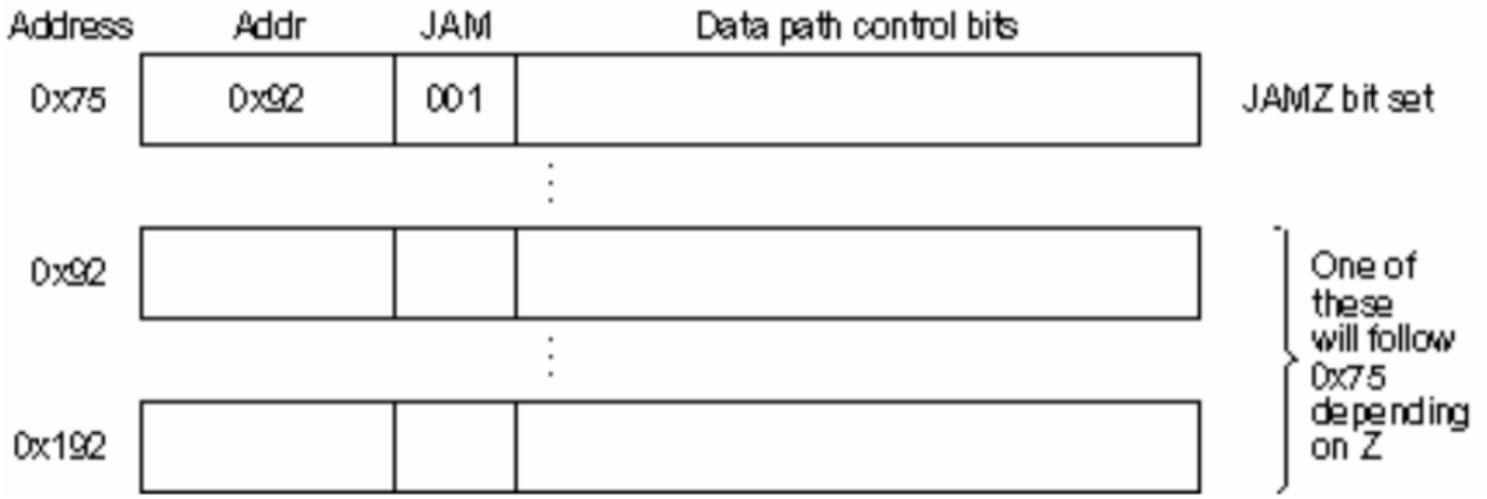


Controle Microprogramado – Mic-1

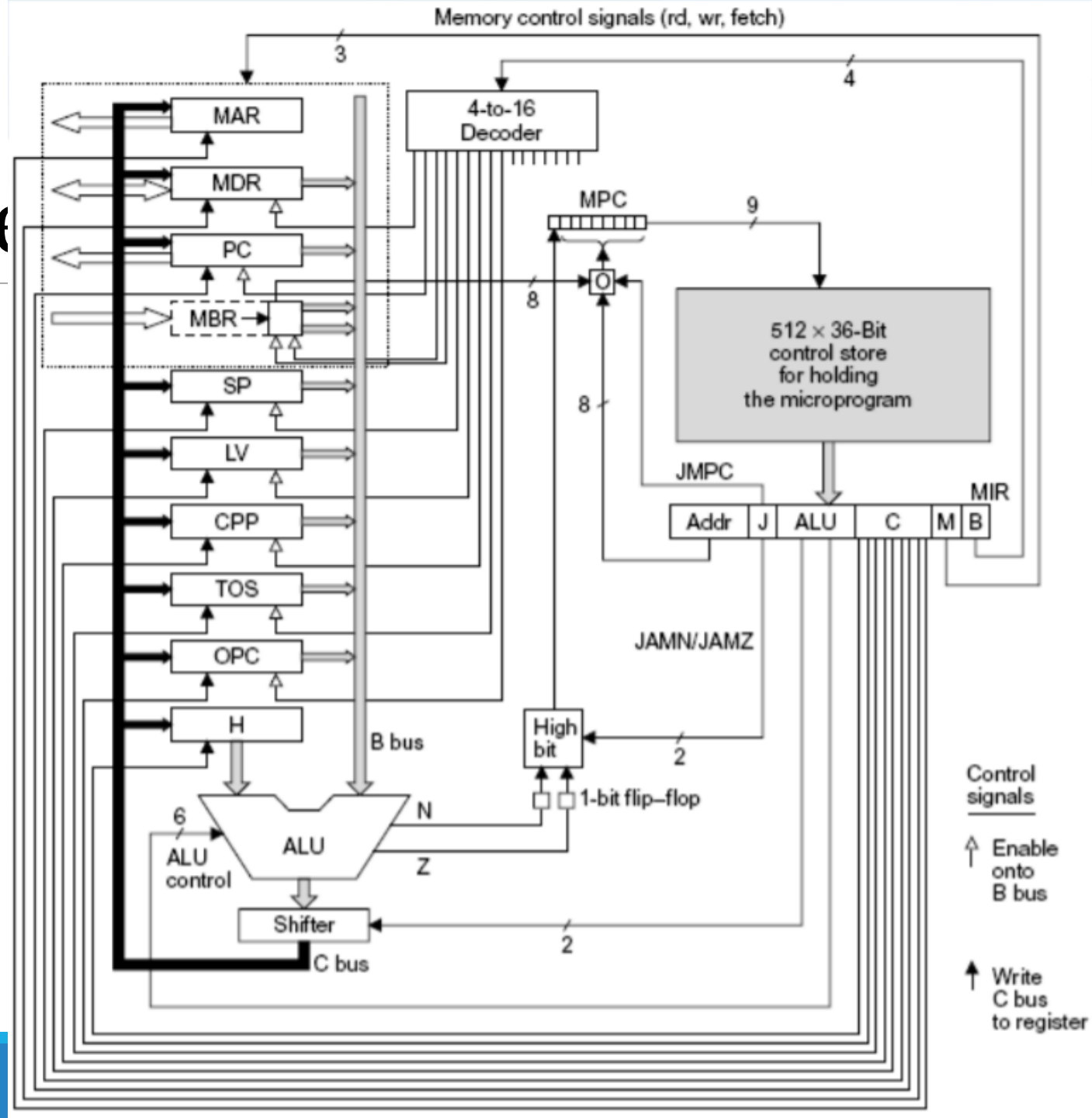


Controle Microprogramado – Mic-1

- As microinstruções são lidas em sequência e esta sequência pode ser quebrada através de instruções de desvio.
- Para se fazer um desvio, seleciona-se a próxima instrução através de microinstruções do tipo jump.
  - Desvios podem ser condicionais ou incondicionais.
  - Nos desvios condicionais, flags são testados para verificar condição de desvio.



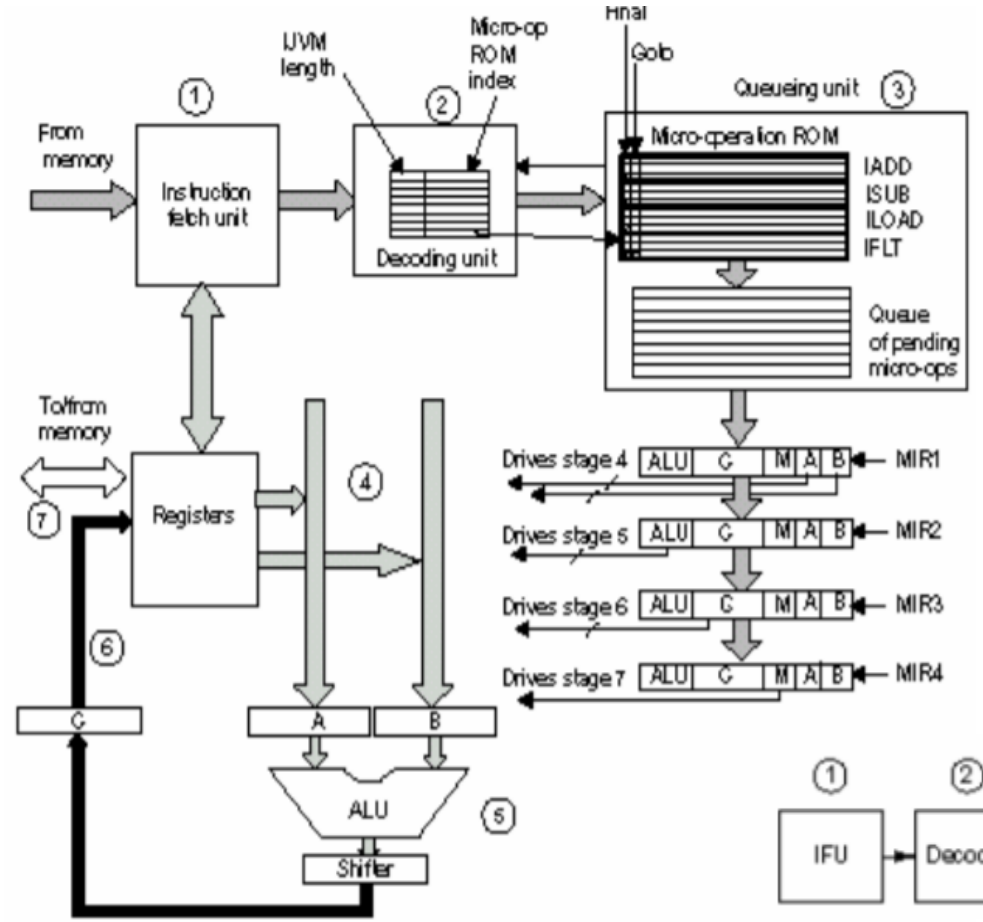
Controle


Memory control signals (rd, wr, fetch)
3
4
MAR
MDR
PC
MBR
SP
LV
CPP
TOS
OPC
H
4-to-16
Decoder
MPC
9
8
O
512 × 36-Bit
control store
for holding
the microprogram
8
JMPC
MIR
Addr
J
ALU
C
M
B
JAMN/JAMZ
B bus
High
bit
2
1-bit flip–flop
N
Z
6
ALU
control
ALU
Shifter
2
C bus
Control
signals
Enable
onto
B bus
Write
C bus
to register

Resumo

- O caminho de dados é o coração de um processador. Nele estão alguns registradores, dois barramentos e uma ou mais unidades funcionais (ULAs e deslocadores).
- O principal *loop* de execução é composto pela busca de alguns operandos dos registradores e pelo envio desses operandos para a ULA, com o intuito de executar uma operação sobre eles. Os resultados dessa operação são armazenados de volta nos registradores.
- O sequenciador exerce o controle sobre o caminho de dados, buscando as microinstruções em uma memória de controle.
- Cada microinstrução tem um conjunto de bits que controla o caminho de dados durante um ciclo. Esses bits especificam os operandos a serem selecionados, as operações a serem executadas e o que fazer com os resultados dessas operações.
- Além disso, cada microinstrução especifica a sua sucessora potencial (em geral em um campo da própria microinstrução).

Microarquitetura MIC4



Implementação de um Pipeline subdividindo o estágio de execução das microinstruções.

Observe a existência de 4 MIR, cada um controlando um estágio do pipeline

Estágios do Pipeline do MIC4

Referências

- Andrew S. Tanenbaum, Organização Estruturada de Computadores, 5ª edição, Prentice-Hall do Brasil, 2007.
- Roberta L. Gomes, Nível da Microarquitetura, Universidade Federal do Espirito Santo, 2007.