

Nível de Microprogramação

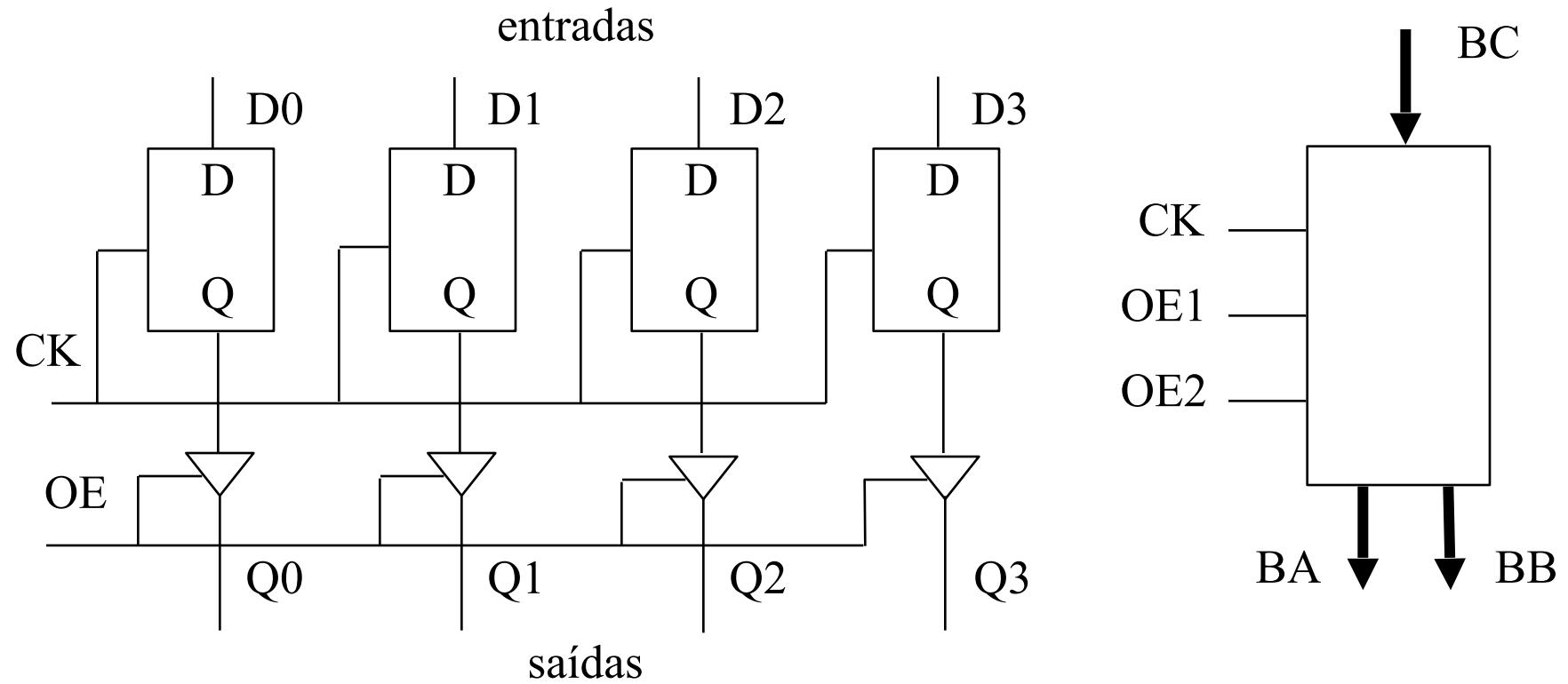
O nível de microprogramação tem uma função específica: *executar interpretadores para outras máquinas virtuais*.

Um *microprograma* comprehende um programa que controla os registradores, os barramentos, a ULA, as memórias e outros componentes do *hardware*.

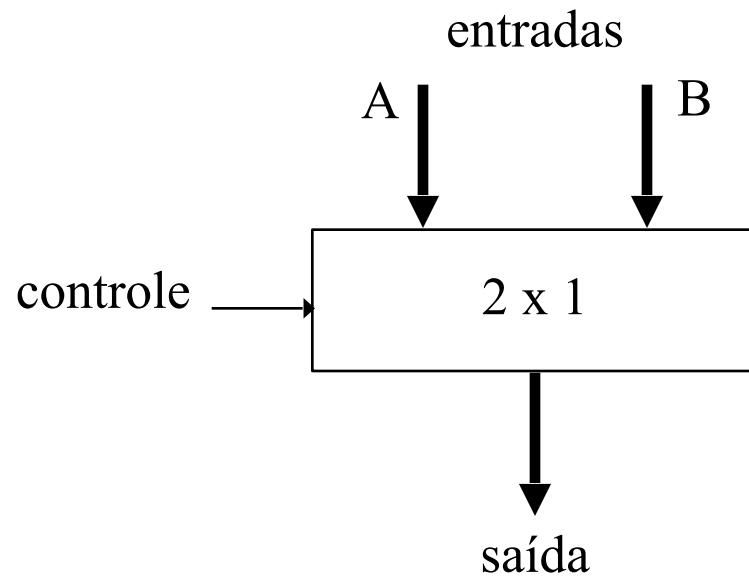
Os registradores estão localizados fisicamente dentro do processador.

Um *barramento* é uma coleção de fios usados para transmitir sinais em paralelo. Pode ser *unidirecional* ou *bidirecional*.

Um barramento *tri-state* possui dispositivos capazes de apresentarem na saída 0, 1 ou alta impedância. Utilizados quando há muitos dispositivos ligados a um mesmo barramento.

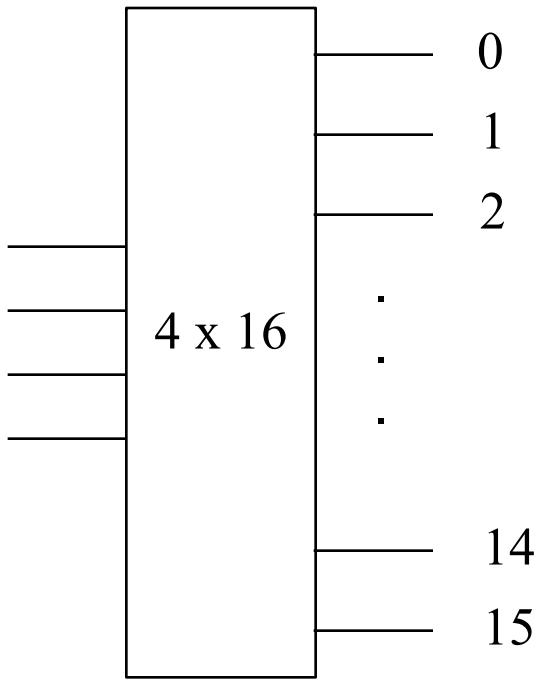


Um multiplexador tem 2^n entradas, uma saída da mesma largura da entrada e uma entrada de controle de n bits, que seleciona uma das entradas e a direciona para a saída.



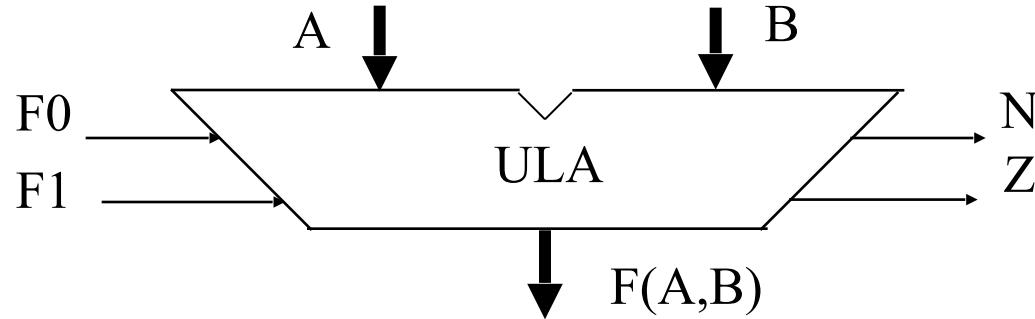
Um *demultiplexador* é o inverso de um multiplexador, direcionando a entrada para uma dentre 2^n saídas, de acordo com as n linhas de controle.

Um decodificador tem n linhas de entrada e 2^n linhas de saída. De acordo com o código binário da entrada, uma das saídas é ativada.

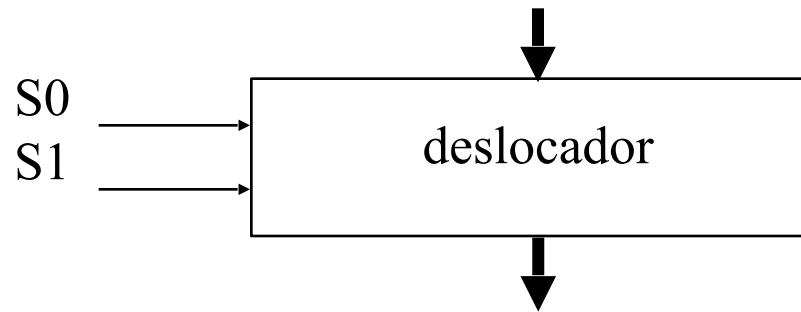


Um codificador é o inverso de um decodificador, possuindo 2^n entradas e n saídas. Somente uma das entradas estará ativa.

A Unidade Lógica e Aritmética possui duas entradas e uma saída para dados, havendo outras entradas e saídas de controle.



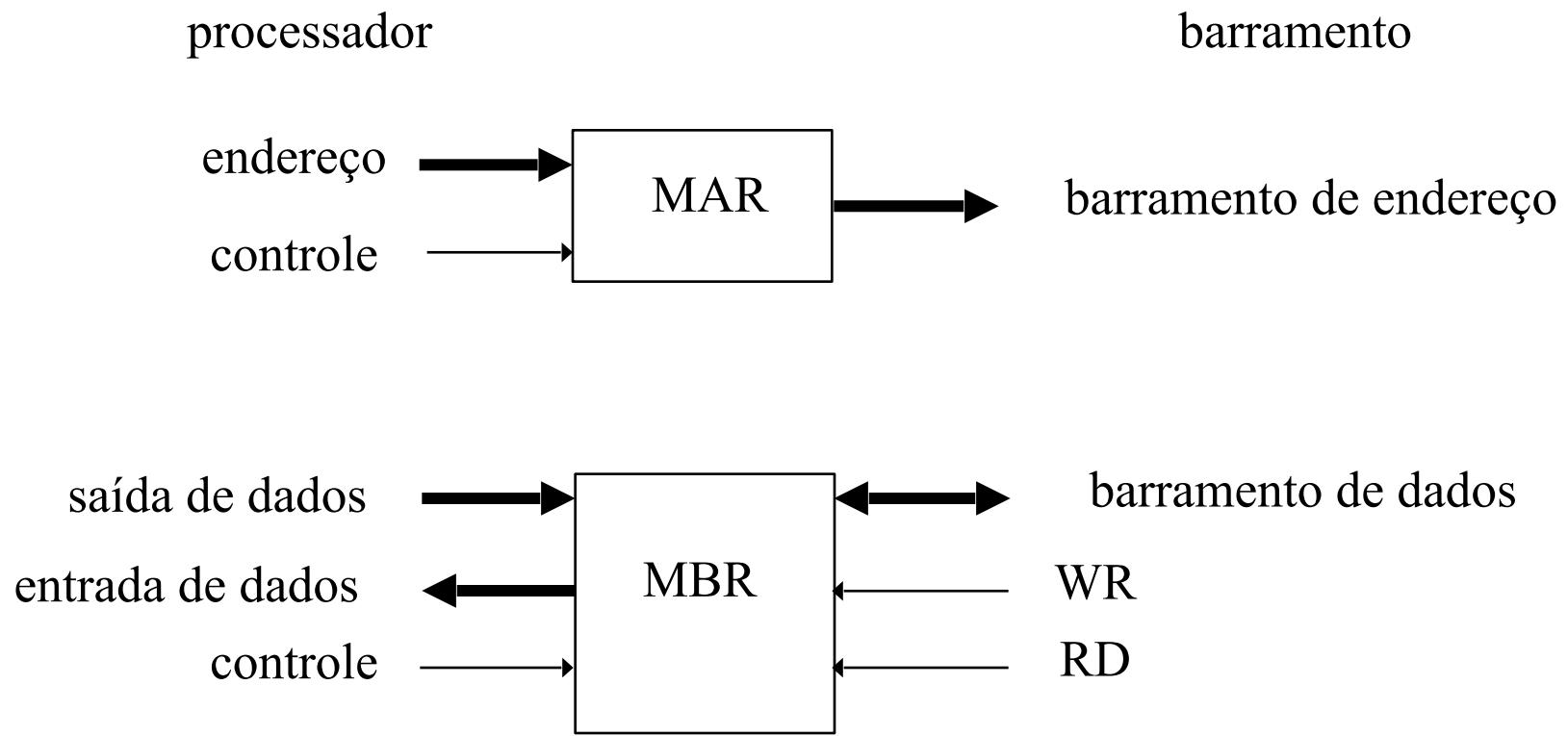
Um deslocador é um circuito com capacidade para deslocar à direita ou à esquerda, ou mesmo não deslocar.



A maioria dos computadores tem um barramento de endereço, um barramento de dados e sinais de controle para a comunicação entre a UCP e os demais componentes do sistema.

Um acesso à memória é quase sempre consideravelmente mais demorado que o tempo necessário para executar uma única microinstrução.

O registrador MAR é responsável pelo armazenamento do endereço da memória. O registrador MBR é responsável pelo armazenamento do dado.



A linha de controle de MBR permite carregar o registro com dado da UCP. O sinal RD carrega o registro com dado do barramento. O sinal WR libera o conteúdo do registro no barramento.

[Fluxo de Dados]

Um formato de microinstrução, contendo alguns campos codificados, pode ser:

1	2	2	2	1	1	1	1	4	4	4	8	
A M U X	C O N D	A L U	S H	M B R	M A R	R D	W R	E N C	C	B	A	ADDR

AMUX : 0 = latch A; 1 = MBR

COND: 0 = não salta; 1 = salta se N=1;
2 = salta se Z=1; 3 = salta sempre

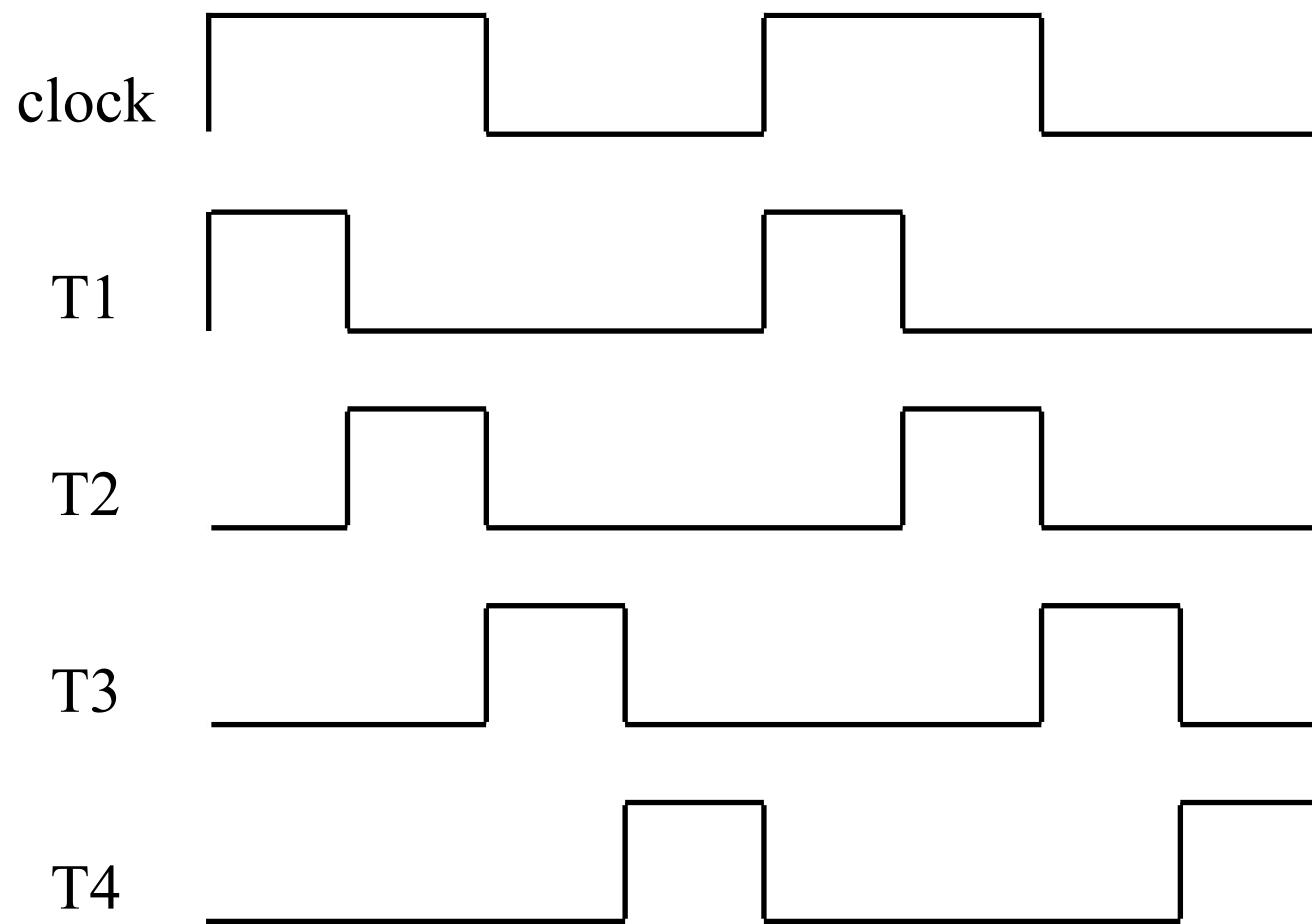
ALU: 0 = A+B; 1 = A.B; 2 = A; 3 = NOT A

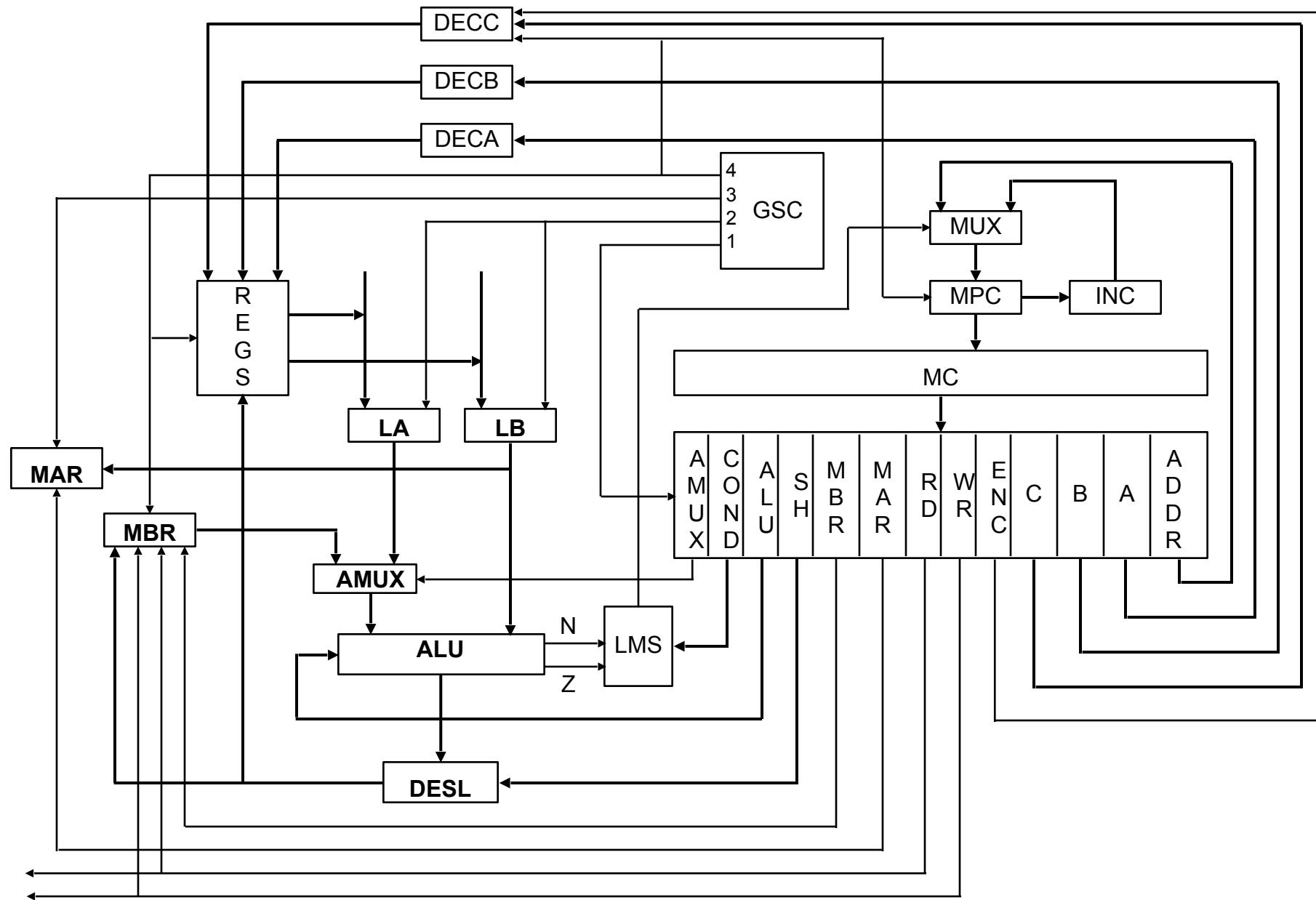
SH: 0 = não desloca; 1 = desloca 1 bit à direita;
2 = desloca 1 bit à esquerda; 3 = x

Um ciclo básico consiste em colocar os valores nos barramentos A e B, armazená-los nos dois *latches*, passá-los pela ALU e pelo deslocador, e armazená-los na memória local ou no MBR. O seqüenciamento dos eventos compreende:

- 1 - carregar a próxima microinstrução no registrador de microinstrução (MIR);
- 2 - colocar o conteúdo dos registros nos barramentos A e B, e guardá-los nos *latches* A e B;
- 3 - dar tempo à ALU e ao deslocador para produzirem um resultado e carregar o MAR, se necessário;
- 4 - armazenar o valor existente no barramento C, na memória local ou no MBR.

Subciclos





A escolha da próxima microinstrução é determinada pela *lógica de microsequenciamento*, durante T4, quando N e Z são válidos.

As condições de desvio são:

- 0 = não salte (a próxima microinstrução está em MPC + 1);
- 1 = desvie para ADDR, se N = 1;
- 2 = desvie para ADDR, se Z = 1;
- 3 = desvie indondicionalmente para ADDR.

O microprograma, para a arquitetura proposta, deve realizar a busca, decodificação e execução da instrução do programa de nível convencional de máquina.

O registrador AMASK é a máscara de endereço (0x007777) usada para separar os bits de endereço do restante da instrução.

O registrador SMASK é a máscara de pilha (0x000377) usada para separar a constante, associada às instruções INSP e DESP, do restante da instrução.

Para realizar uma subtração utiliza-se complemento a dois:

$$x - y = x + \bar{y} + 1$$

[Instruções]

A microlinguagem de montagem consiste dos seguintes comandos:

- 1 - atribuição: $AC := A;$
- 2 - aritmética: $PC := PC + 1;$
- 3 - lógica: $A := \text{band}(IR, SMASK);$
 $B := \text{inv}(C);$
- 4 - deslocamento: $TIR := \text{lshift}(TIR);$
 $D := \text{rshift}(A + B);$
- 5 - desvios: $\text{goto } 0;$
 $\text{if } N \text{ then goto } 50;$

Em muitos computadores, a microarquitetura tem suporte de hardware para extrair código de operação da macroinstrução e colocá-lo diretamente no MPC.

Não há instruções de E/S, utilizando E/S mapeada em memória.

(4092) - dado a ser lido

(4093) - o bit de sinal indica dado para leitura (=1)

(4094) - dado a ser escrito

(4095) - o bit de sinal indica dispositivo pronto (=1)

Microprogramação horizontal: microinstrução com campos muito pouco codificados.

Microprogramação vertical: microinstrução com campos mais codificados.

Um microprograma é mais vertical quanto maior for o grau de codificação da microinstrução.

Uma microinstrução extremamente vertical poderia ter um código de operação e operandos.

[Microinstruções Verticais]

