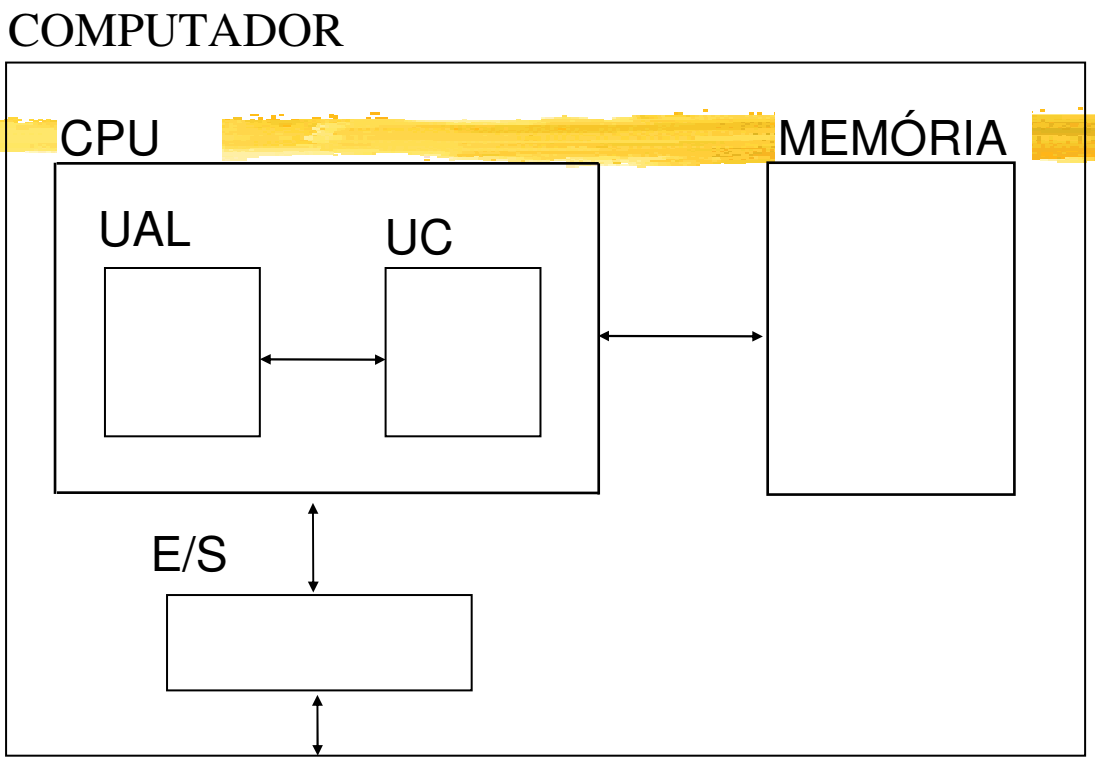
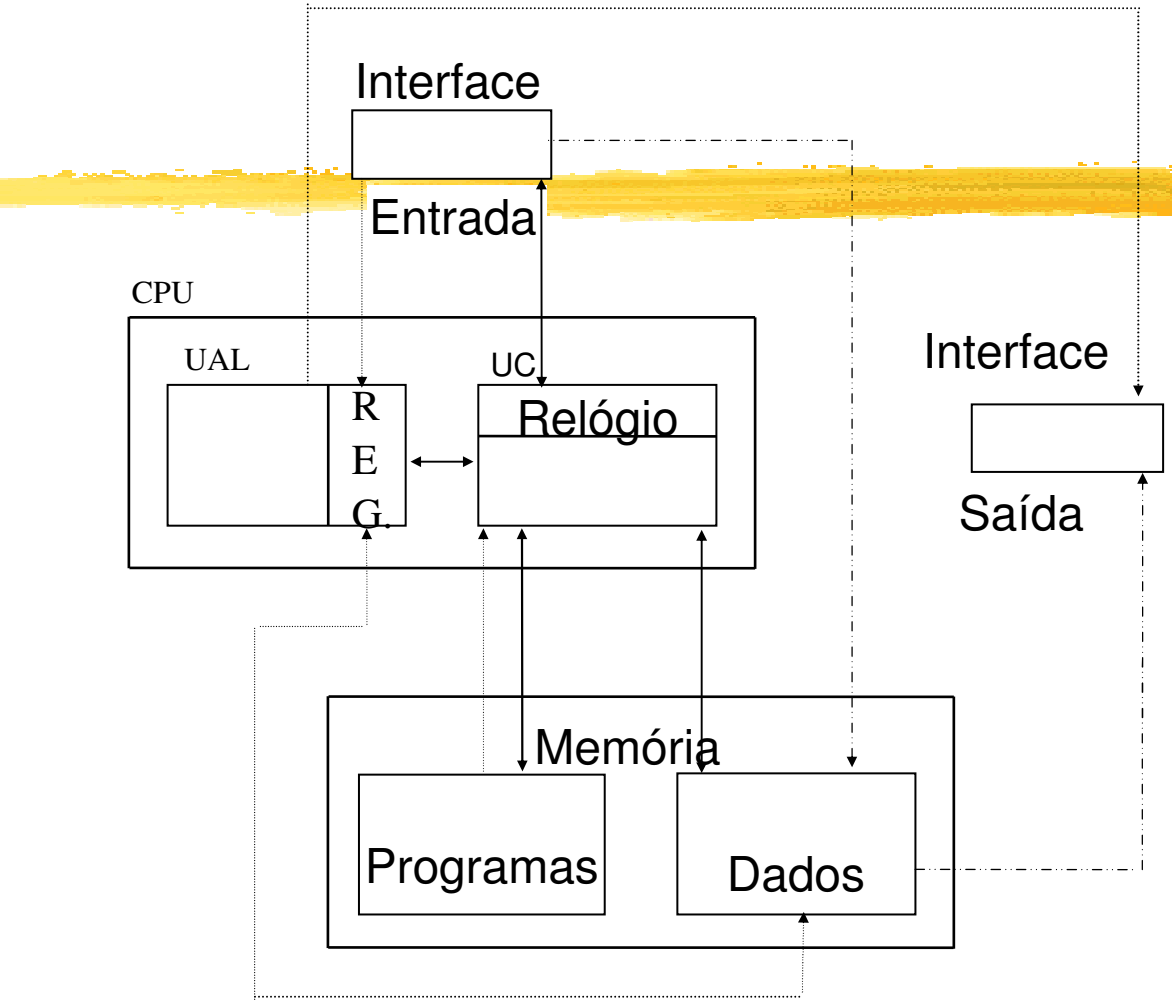
3.5 Organização de Computadores

Estrutura Modular de um Computador



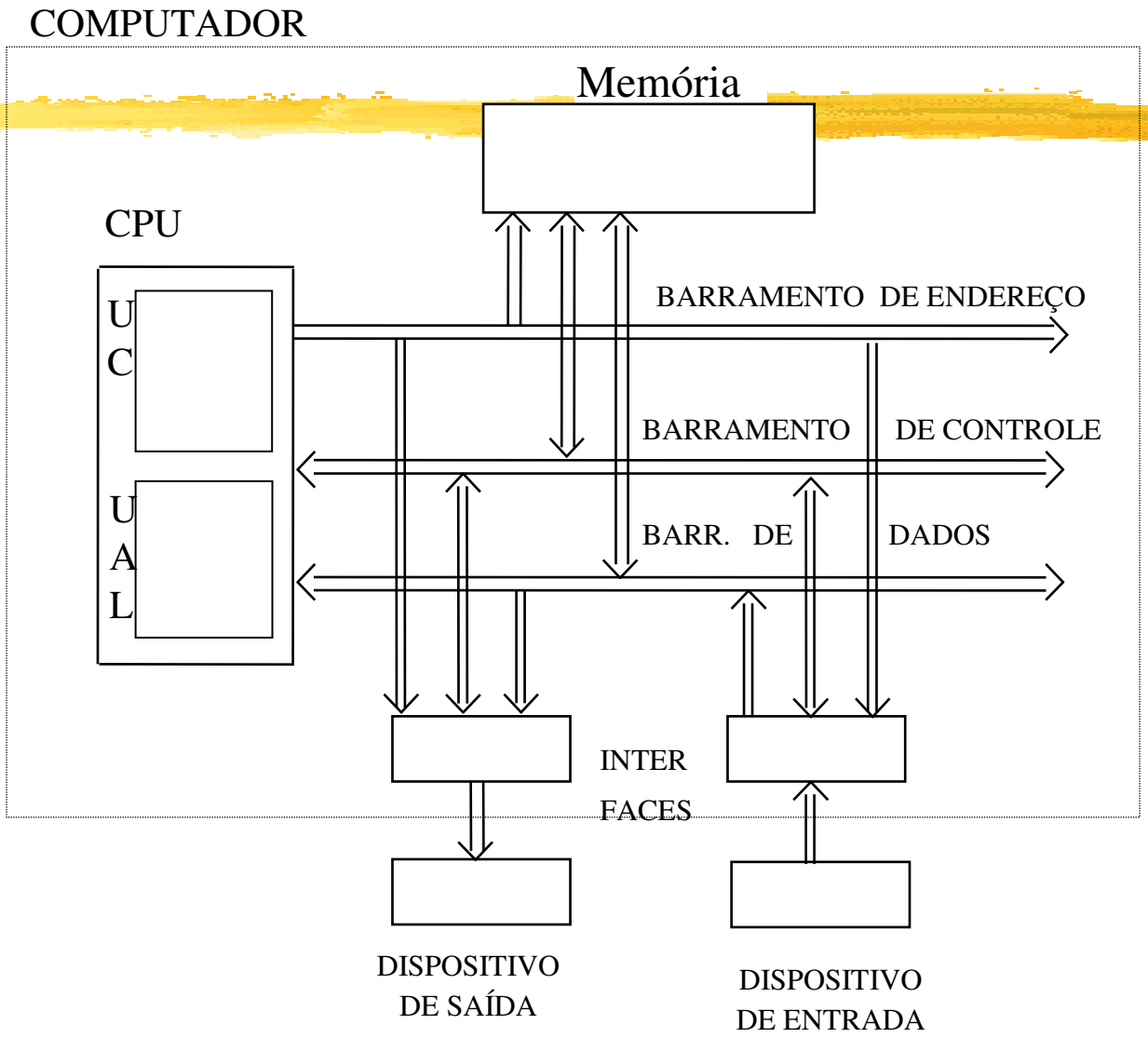
CPU - Unidade Central de Processamento
UAL - Unidade Aritmética e Lógica
UC - Unidade de Controle
E/S - Interface de Entrada/Saída

Fluxo de Dados e de Controle



Fluxo de Controle
Fluxo de Dados
ADM - Acesso Direto à Memória

Barramentos



Barramentos - detalhes

Como já vimos são ligações (caminhos elétricos) entre os módulos / partes de um sistema computacional.

Devem ser analisados sob 3 aspectos:

Mecânico: tamanho, material, conectores. PLUG ⟷ SOQUETE

Elétrico: especificações elétricas em cada um de seus pinos / ou trilhas.

Protocolo (funcional): sinais – dados e controle – que fluem pelos seus pinos ou trilhas.

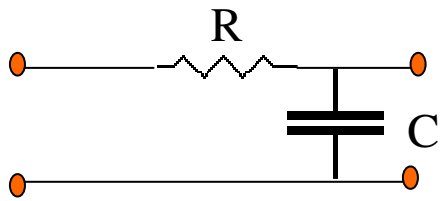
Barramento em uma placa mãe de PC

Barramento x Linha de Transmissão
(ver Alan Clements, Microprocessor System Design)

Por vezes não atentamos que ao transportar um sinal (em alguma frequência) temos de considerar o comportamento de um barramento como o de uma linha de transmissão.

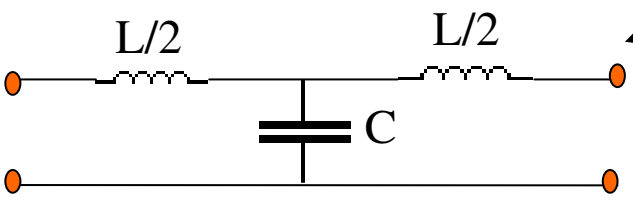
O primeiro cabo submarino entre USA-Europa (1855) mereceu um tratamento matemático feito por Lord Kelvin de forma a estudar o seu comportamento.

Modelo mais simples de um barramento:



aplica-se em trilhas curtas: < dezenas de cm

para trilhas mais longas



atraso de propagação de um pulso é dado por $\sqrt{L.C}$ por unidade de comprimento.

Alguns valores típicos:

meio	L (nH/cm)	C (pF/cm)	Z (ohm)	T (ns/m)
Fio	20	0,06	600	4
Trilha PCB	5 – 10	0,5 -1,5	70 -100	5
Backplane busline	5 – 10	1 – 3	100	7 - 17

Aproximação experimental para análise de atraso em trilhas:

seja - a = tempo de subida/descida do sinal
b = atraso de propagação / metro
c = comprimento da trilha

→ razão = a/(b.c)

Exemplo 1: a = 2 ns, b = 5 ns/m , c= 4 cm → razão = 10

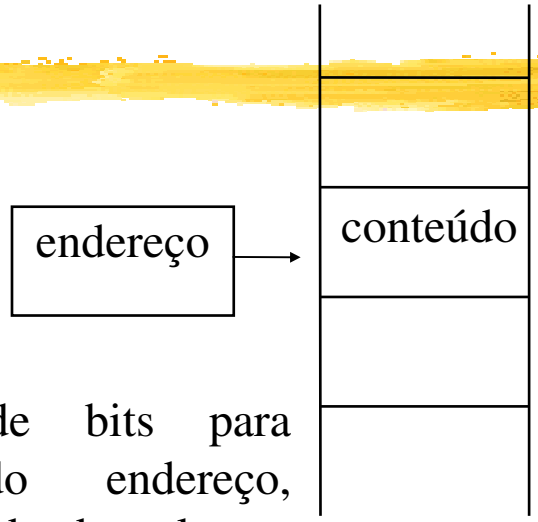
Exemplo 2: a = 2 ns, b = 20 ns/m , c= 50 cm → razão = 0,25

Analisando esta razão vemos que no primeiro caso o tempo de subida domina o comportamento enquanto no segundo não e assim neste **segundo caso deve-se considerar o barramento como uma linha de transmissão**.

3.6 Descrição de um computador

- Memórias
  - principal; associativa; cache
- UAL
  - máquinas de 0 a 3 endereços; flags
- UC
  - registradores
- Instrução de máquina e sua execução

Memória (principal)



a quantidade de bits da palavra da memória, determina a quantidade de informação que pode ser representada

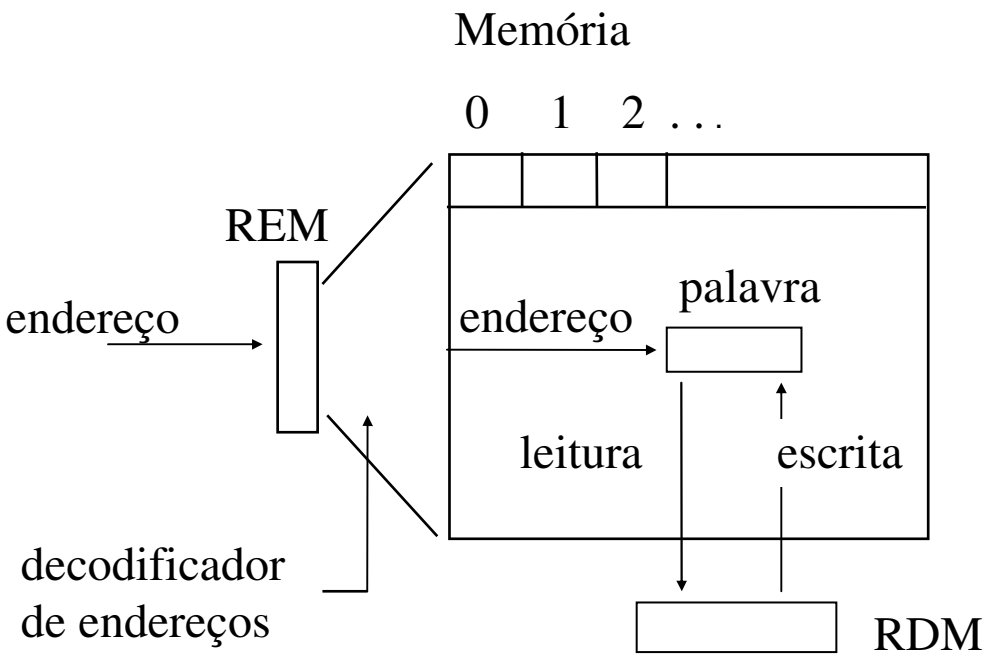
m bits $\rightarrow 2^{m}$ dados distintos

a quantidade de bits para especificação do endereço, define a quantidade de palavras que podem ser acessadas

n bits $\rightarrow 2^{n}$ palavras distintas

X - endereço
(X) - conteúdo do endereço X,

(nossa) Representação da memória principal

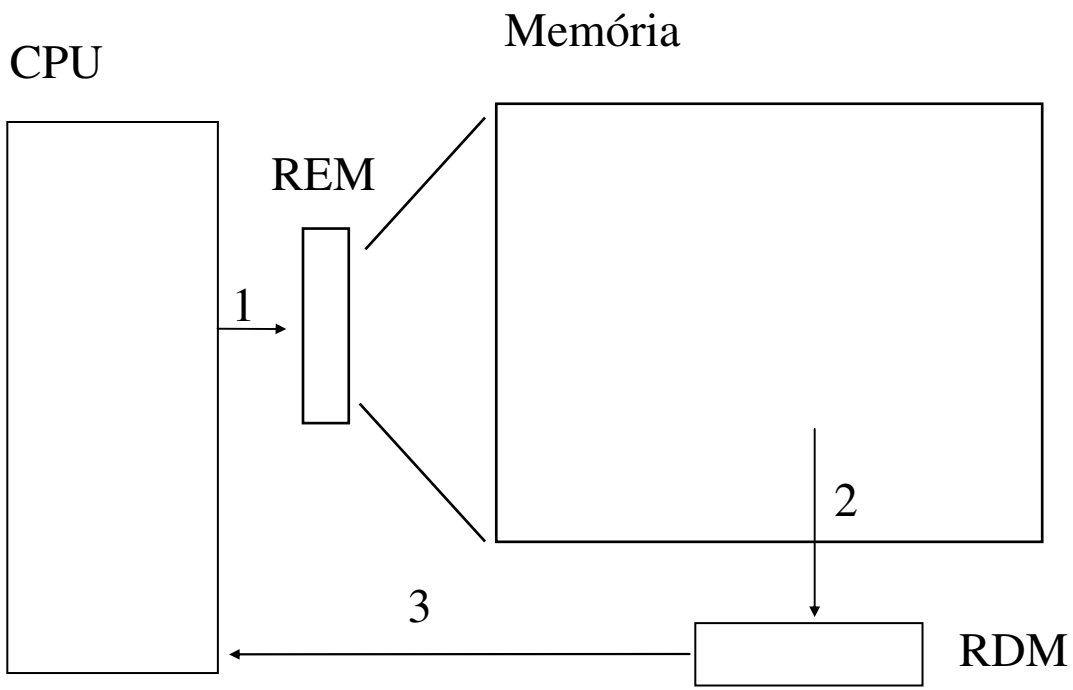


REM - registrador de endereços da memória

RDM - registrador de dados da memória

Ciclos de memória
Ciclo de Leitura

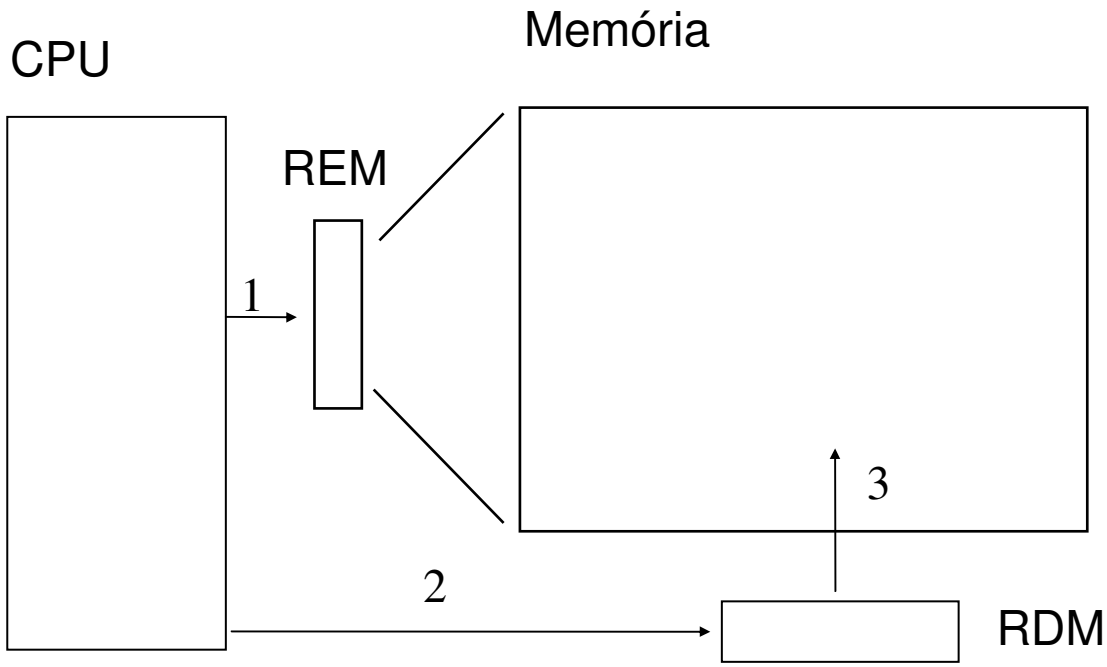
Ciclo de Memória corresponde a um acesso à memória, seja de leitura ou de escrita.



Pulso	Microoperação
1	REM ← endereço
2	leitura :
	RDM ← ((REM)) ou
	RDM ← (m)
3	CPU ← (RDM)

onde m é o endereço de uma posição de memória

Ciclos de memória
Ciclo de Escrita

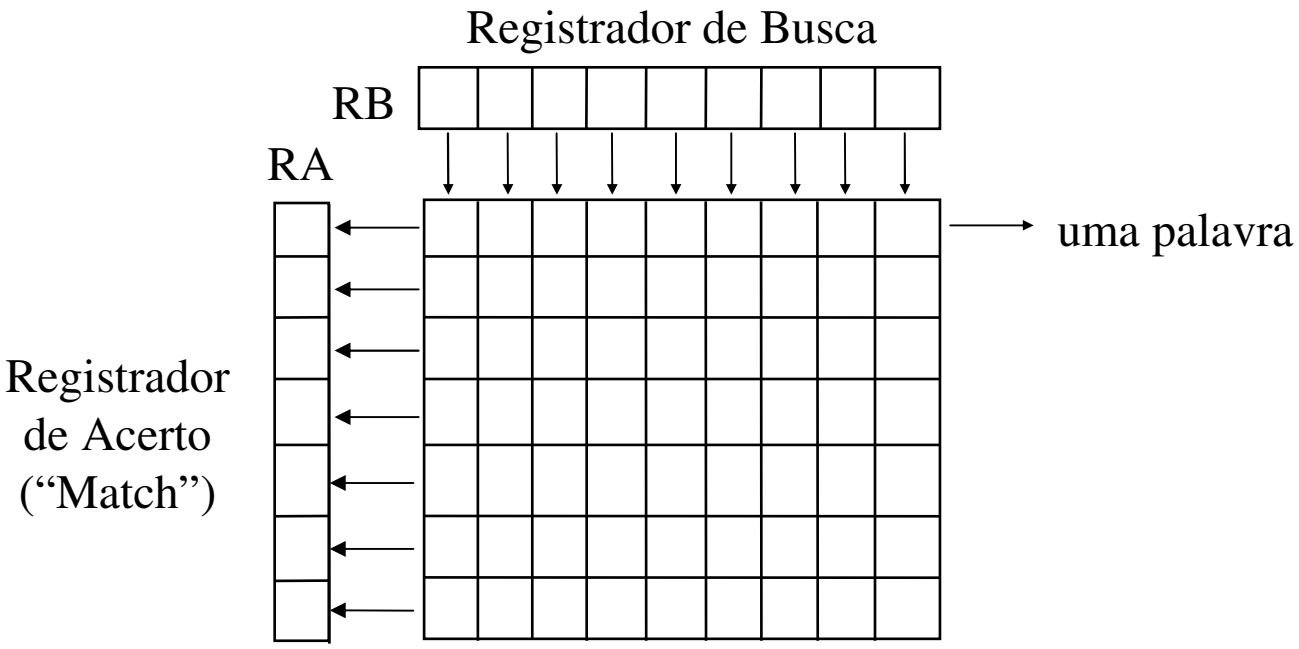


Pulso	Microoperação
1	REM ← endereço
2	RDM ← dado
3	escrita :
	(REM) ← (RDM)
	ou m ← (RDM)

onde "**(REM) ←**" significa armazenar no endereço contido no REM

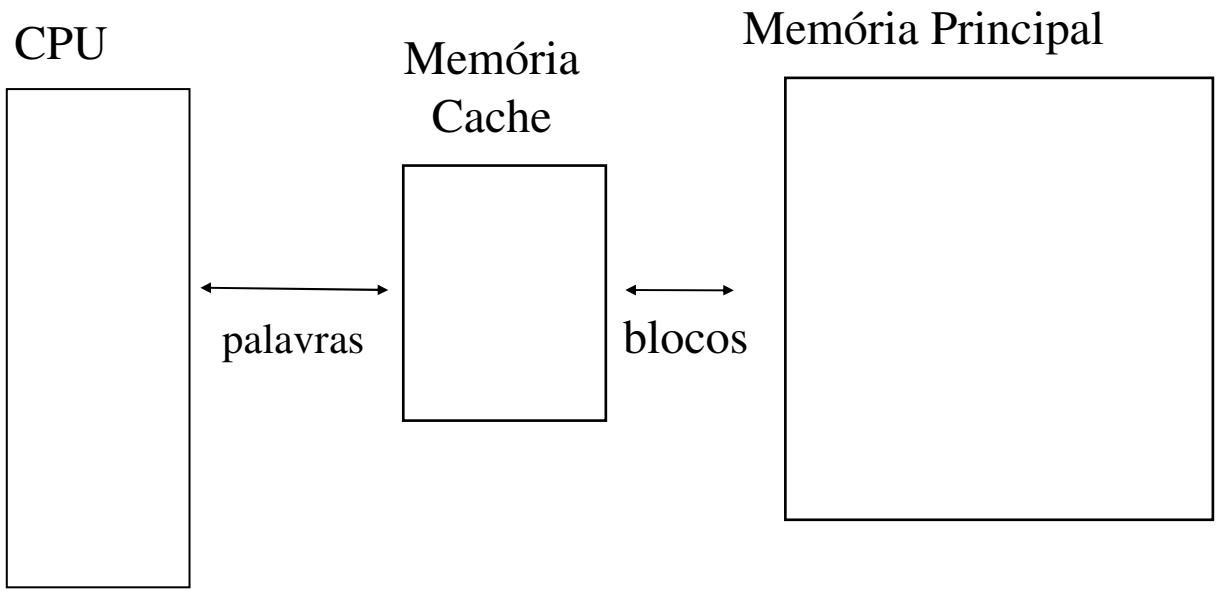
“ Outras memórias”

Memória associativa: o acesso é realizado pelo conteúdo e não pelo endereço.



“ Outras memórias” – cont.

Memória cache: normalmente colocada entre memória principal e CPU e tendo característica de alta velocidade, proporciona um acesso mais rápido.



Princípio da localidade

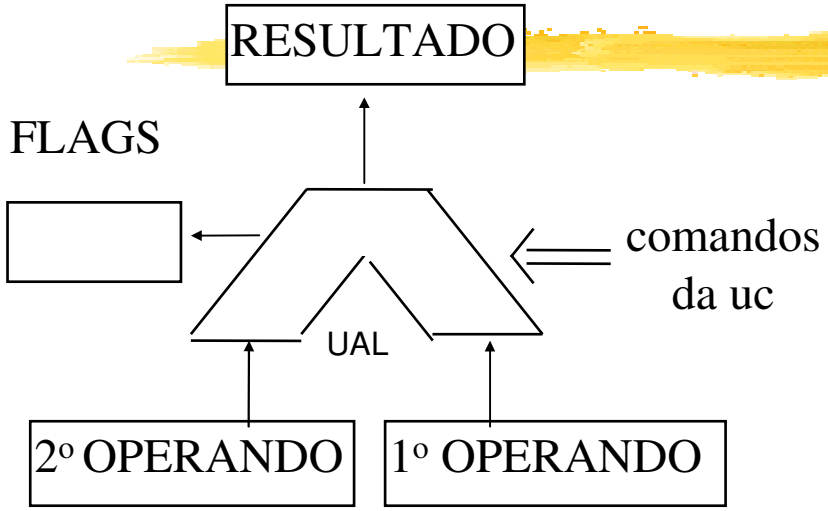
Unidade Aritmética e Lógica - UAL

A **UAL** faz parte da CPU, sendo responsável pela execução de todas as operações sobre os dados.

A UAL pode ser classificada, dependendo de como devem ser especificados os operandos e resultados, em :

- máquina de 3 endereços,
- máquina de 2 endereços,
- máquina de 1 endereço,
- máquina de zero endereços.

UAL – cont.
Máquina de 3 endereços



Formato da instrução

C.O.	OP1	OP2	RES
código de operação	end. 1º operando	end. 2º operando	endereço resultado

RES ← (OP1) operação (OP2)

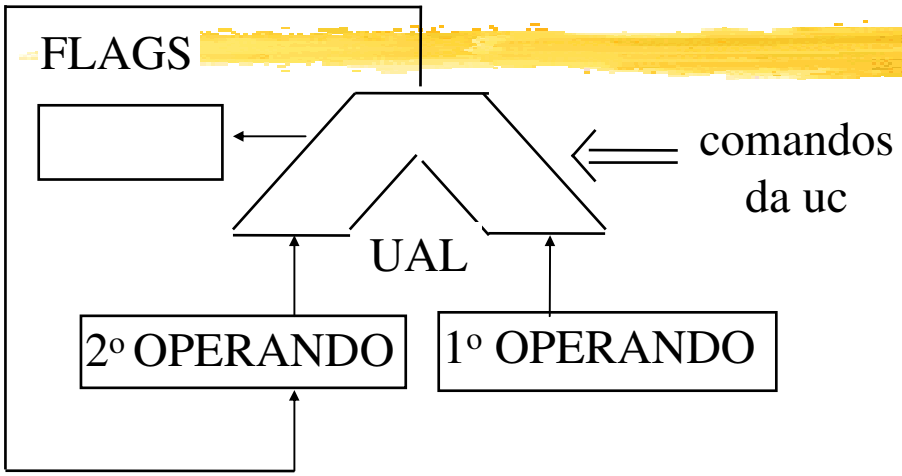
Exemplo de adição: R = S1 + S2 ⇒ R ← (S1) + (S2)

ADD			
código de operação	end. 1º operando	end. 2º operando	endereço resultado

Em linguagem simbólica (Assembly) teríamos, por exemplo:

ADD S1, S2, R ⇒ R ← (S1) + (S2)

UAL – cont.
Máquina de 2 endereços



Formato da instrução

C.O.	OP1	OP2
código de operação	end. 1º operando	end. 2º operando

OP2 ← (OP1) operação (OP2)

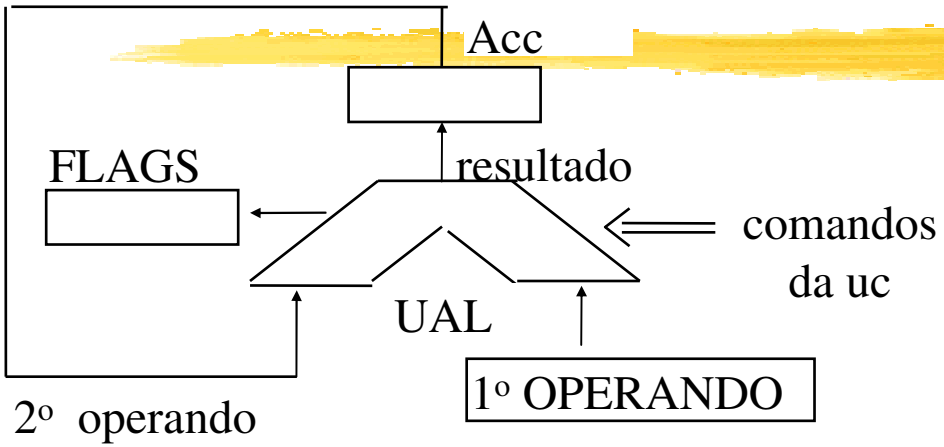
Exemplo de adição: R = S1 + S2 ⇒ R ← (S1) + (S2)

ADD		
código de operação	end. 1º operando	end. 2º operando

Em linguagem simbólica (Assembly) teríamos, por exemplo: (necessitamos de 2 instruções)

MOVE S1, R ⇒ R ← (S1)
ADD S2, R ⇒ R ← (R) + (S2)

UAL – cont.
Máquina de 1 endereço



Formato da instrução

C.O.	OP1
código de operação	endereço operando

Acc ← (Acc) operação (OP1)

Exemplo de adição: R = S1 + S2 ⇒ R ← (S1) + (S2)

ADD	
código de operação	endereço operando

Em linguagem simbólica (Assembly) teríamos, por exemplo: (necessitamos de 3 instruções)

LOAD	S1	⇒	Acc ← (S1)
ADD	S2	⇒	Acc ← (Acc) + (S2)
STA	R	⇒	R ← (Acc)

UAL – cont.
Máquina de zero endereços

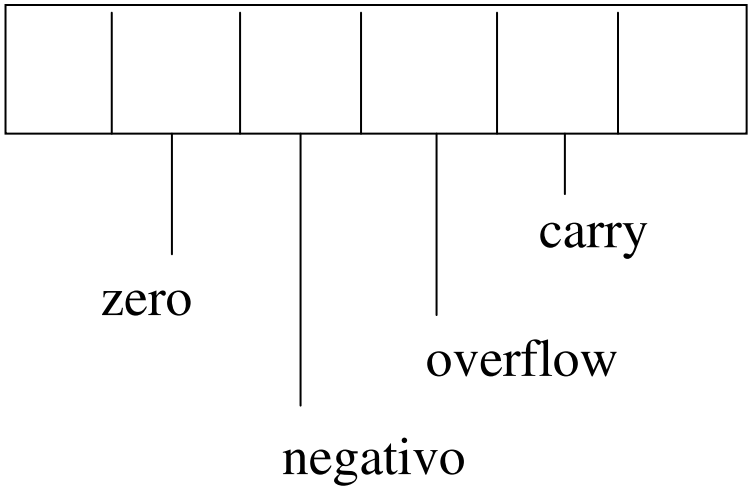
Máquinas de zero endereços também denominadas “stack machines” ou máquinas por pilha, armazenam os operandos em uma pilha necessitando assim somente da operação no corpo da instrução.

Flags ou “bandeiras”

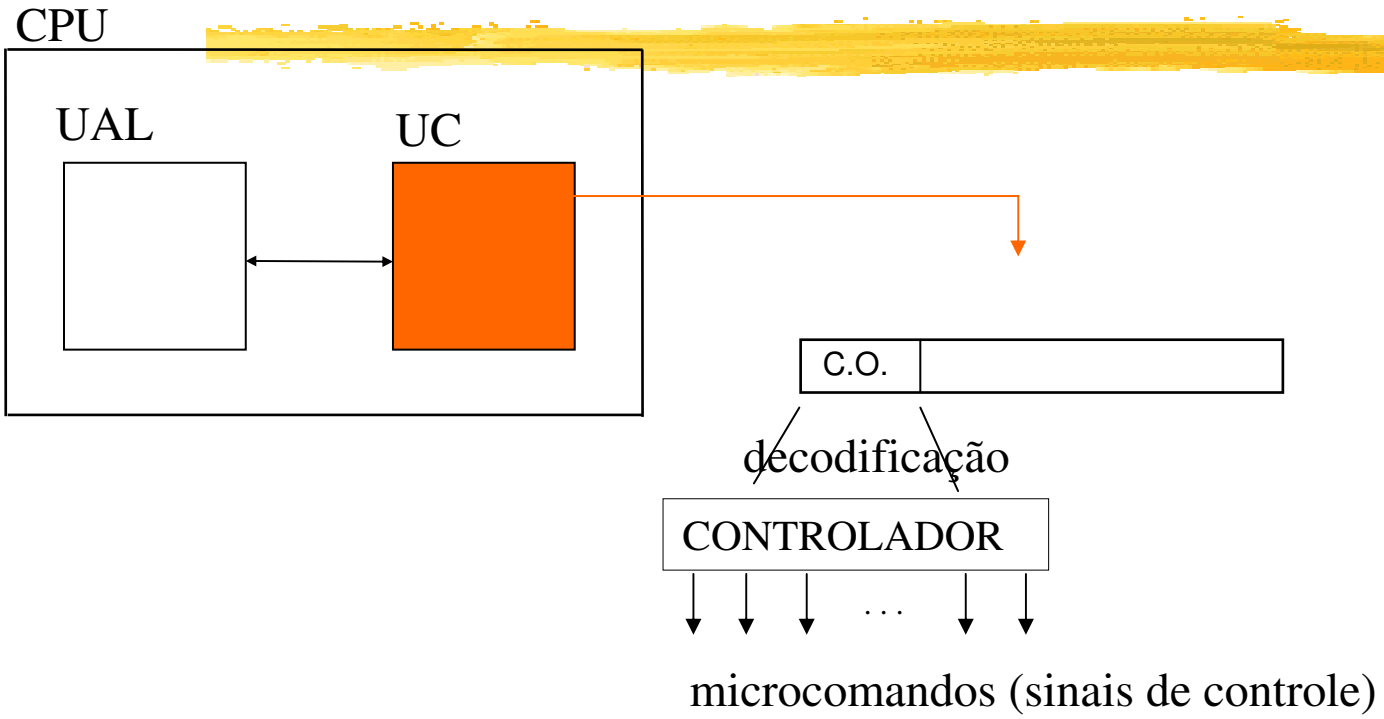
(ver desenhos da UAL nas transparências anteriores)

Flags sinalizam características do resultado da **última** operação realizada pela UAL

Por exemplo: se houve carry (transporte), se o resultado é zero, positivo ou negativo; se houve overflow.



Unidade de Controle - UC



registradores ligados à operação

- reg. **de Instrução** – armazena o CO para a decodificação.
- reg. **PC** – contador de programa, indica a próxima instrução a ser executada.
- reg. **PSW** – reg. de estado, indica os flags da última operação.

Microoperações e Microcomandos

MICROOPERAÇÃO	SINAIS DE CONTROLE (microcomandos)
1. Acc $\leftarrow$ (BUS)	W_A
2. BUS $\leftarrow$ (R1)	R_{R1}
3. TMP $\leftarrow$ (BUS)	W_T
4. TMP $\leftarrow$ (R2)	R_{R2}, W_T

Instruções de máquina - tipos

1. Instruções de transferência de dados: transferem dados, ou blocos de dados, entre diferentes registradores ou regiões de memória.

Exemplo: MOVE R1, R2 R2 ← (R1)

2. Instruções de tratamento de operandos: realizam operações aritméticas ou lógicas.

Exemplo: ADD R1, R2 R2 ← (R1) + (R2)

3. Instruções de desvio: determinam desvio no fluxo do programa, isto é, controlam a sequência de execução do programa.

Exemplos: JMP, Jcond, JSR - salto para subrotina, RET - retorno de subrotina

4. Instruções de Entrada/Saída: realizam a comunicação entre a UCP e as Interfaces de entrada e saída.

Exemplo: IN port Acc ← (port)

5. Miscelânea: instruções especiais, às vezes específicas do processador em particular.

Exemplos: Instruções que manipulam bits

BTST D1,D2 ; testa em D2 o bit indicado em D1

Instruções que manipulam campos no registrador de estado

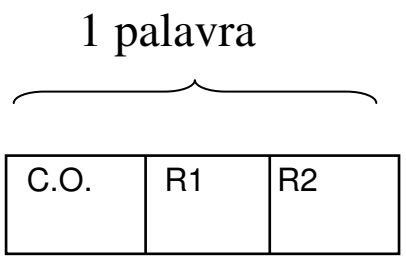
MOVE D0, SR ; transferência envolvendo registradores especiais (no caso o reg. SR ("Status Register")

Instruções de máquina - comprimento

Instruções podem ter diferentes comprimentos, dependendo da quantidade e tipos de operandos envolvidos.

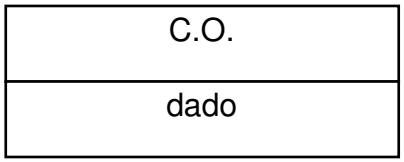
- Instruções de 1 palavra :

  MOVE R1, R2 ;R2 ← (R1)

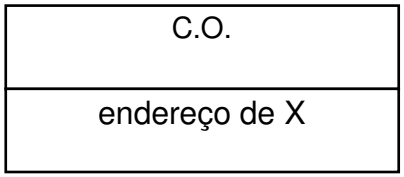


- Instruções de 2 palavras :

  a. ADI dado ; Acc ← (Acc) + dado



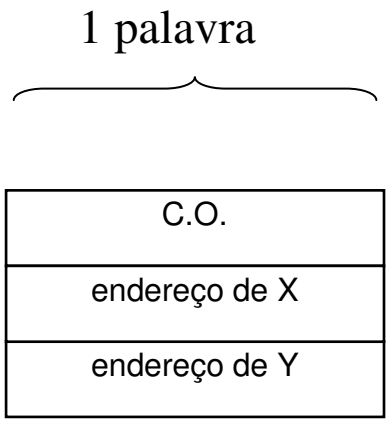
  b. ADD X ; Acc ← (Acc) + (X)



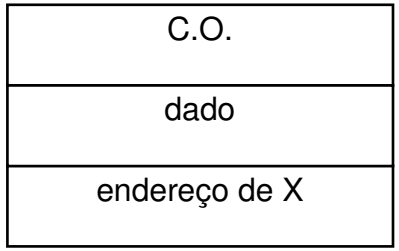
Instruções de máquina – comprimento (cont.)

- Instruções de 3 palavras :

a. ADD X, Y ; $Y \leftarrow (X) + (Y)$



b. ADD #dado, X ; $X \leftarrow (X) + dado$



Instruções de máquina - execução

Ciclo de Máquina é uma sequência de microoperações que são executadas, caracterizando uma operação mais complexa.

Ciclo de Máquina tem um significado vital no entendimento do funcionamento do computador.

A execução de uma instrução é dividida em **Ciclos de Máquina**.

1 instrução de máquina ⟺ j ciclos de máquina ⟺ n microoperações ⟺ m microcomandos

tipos

ciclos de acesso: acessos externos à CPU (de memória e de E/S)

ciclos de exceção: tratamento de exceções

Execução de uma instrução de máquina:

FASE DE BUSCA

A fase de busca compreende o *Ciclo de Máquina* para a leitura do código de operação (C.O.), ou seja, da primeira palavra da instrução. Desta forma, a *Fase de Busca* é idêntica para qualquer instrução.

FASE DE EXECUÇÃO

A fase de execução compreende a execução dos *Ciclos de Máquina* necessários para a leitura das palavras restantes da instrução, se existirem, e da execução da operação identificada pela instrução. A fase de execução é diferente para cada tipo de instrução.

A execução de uma instrução está sempre ligada a *Ciclos de Máquina* que englobam as microoperações necessárias para a execução de cada passo da instrução.

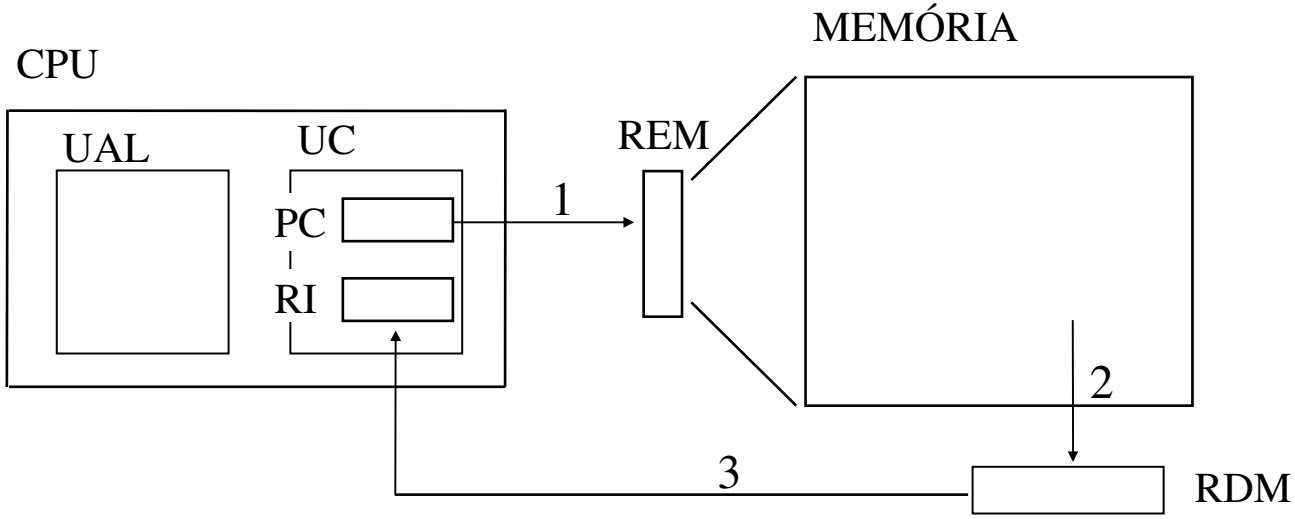
Exemplo: execução da instrução STA end

STA end ⇒ end ← (Acc)

formato da instrução

C.O.
endereço

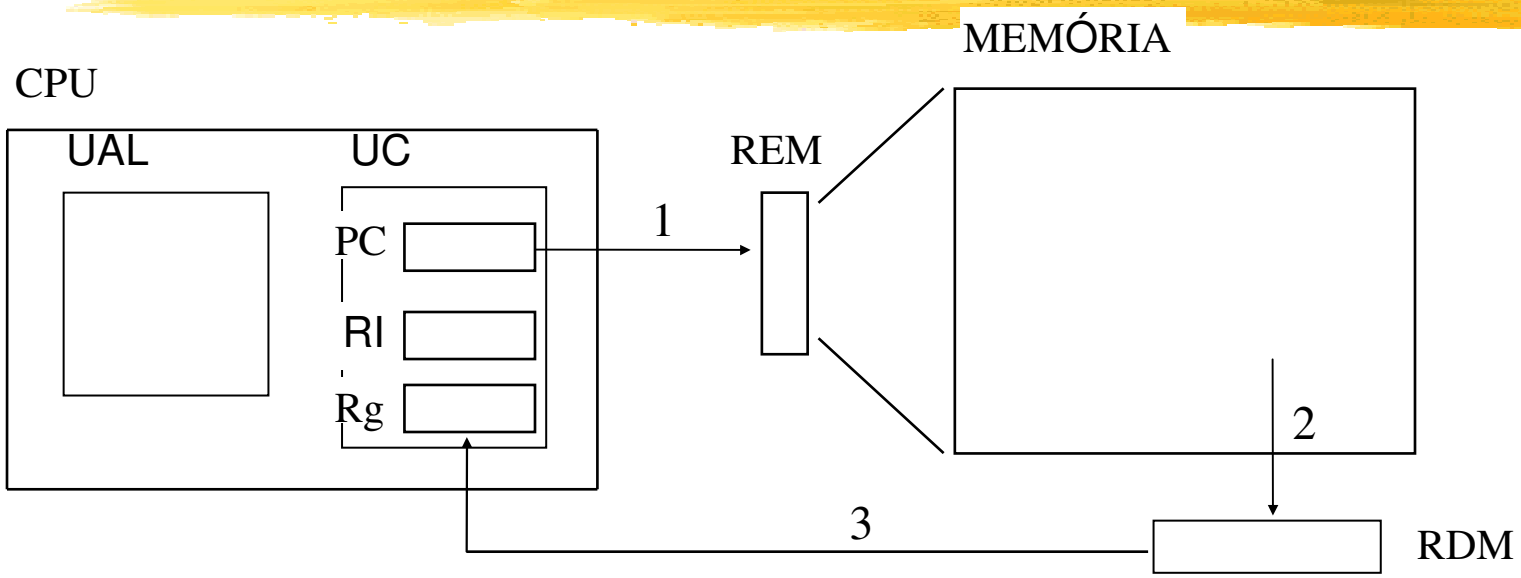
1º ciclo de máquina : FASE DE BUSCA - busca do C.O.



PULSO	MICROOPERAÇÃO
1.	REM ← (PC)
2.	leitura: RDM ← ((REM)) ou RDM ← (m)
	PC ← (PC) + 1 ; operação interna
3.	RI ← (RDM)

Exemplo: execução da instrução STA end

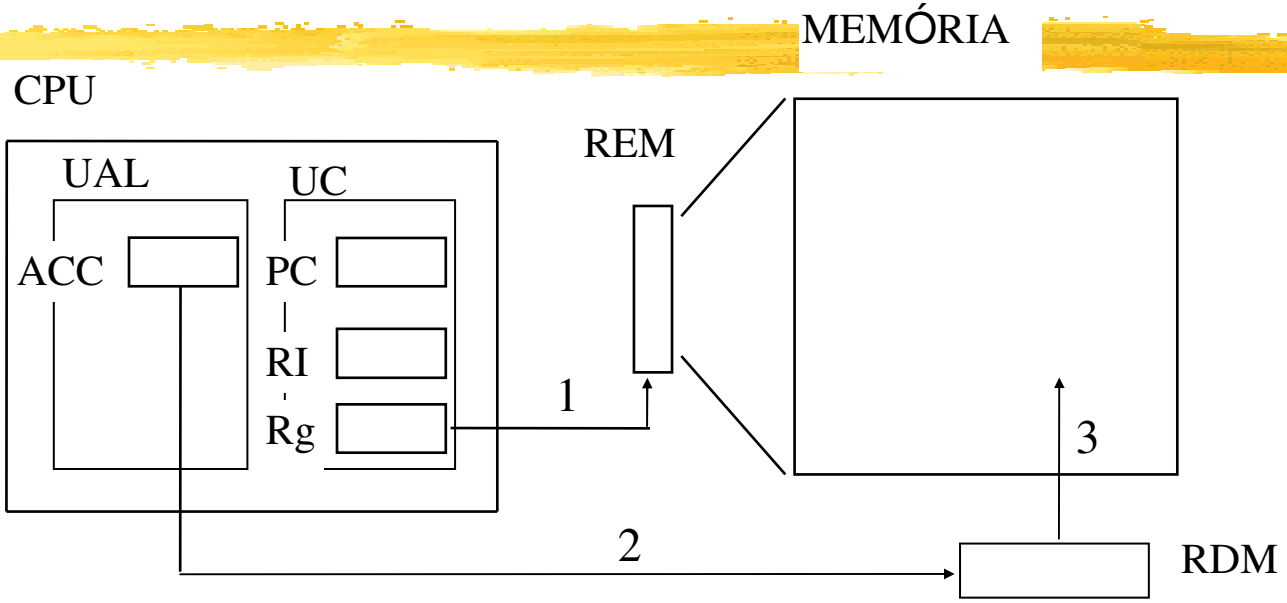
2º ciclo de máquina : Leitura do endereço



PULSO	MICROOPERAÇÃO
1.	REM ← (PC)
2.	leitura: RDM ← ((REM)) ou RDM ← (m)
	PC ← (PC) + 1
3.	Rg ← (RDM)

Exemplo: execução da instrução STA end

3º ciclo de máquina : Transferência do conteúdo de Acc para a memória (endereço end)



PULSO	MICROOPERAÇÃO
1.	REM ← (Rg)
2.	RDM ← (Acc)
3.	escrita: (REM) ← (RDM) ou m ← (RDM)

Comentário:

Neste exemplo:

Fase de Busca: ciclo 1

Busca do código de STA

Fase de Execução: ciclos 2 e 3

Leitura do endereço do operando

Armazenamento do dado no endereço obtido

3.7 Exercícios sobre organização

1. Exemplifique um uso de memória associativa.
2. Exemplifique um uso de memória cache. Explique o princípio da localidade.
3. Exemplifique a operação de uma máquina de zero endereços.
4. Usando os diagramas das transparências 199 - 201 mostre os passos para a execução (fases de busca e de execução) da instrução de desvio:

   JNAcc end ; caso Acc < 0, PC ← end

   se não PC ← (PC) + 1

   (considere que a instrução ocupa 2 palavras)
5. Estude o exemplo 2 apresentado na página 61 do livro texto.