Arquitetura de Computadores

UNIDADE CENTRAL DE PROCESSAMENTO

REFERENCIAS BIBLIOGRAFICAS:

[1] TANEMBAUM, A. S**. Organização Estruturada de Computadores,**Livros Técnicos e Científicos, 2000. 460p.
[2] MONTEIRO, M.A. **Introdução à Organização de Computadores**, 5a ed. Livros Técnicos e Científico Editora SA, 2007. 695p

CURSO SUPERIOR DE ANALISE E DESENVOLVIMENTO DE SISTEMAS
FACULDADE DE TECNOLOGIA DE AMERICANA
Prof. Ms Diógenes de Oliveira (Janeiro/2020)

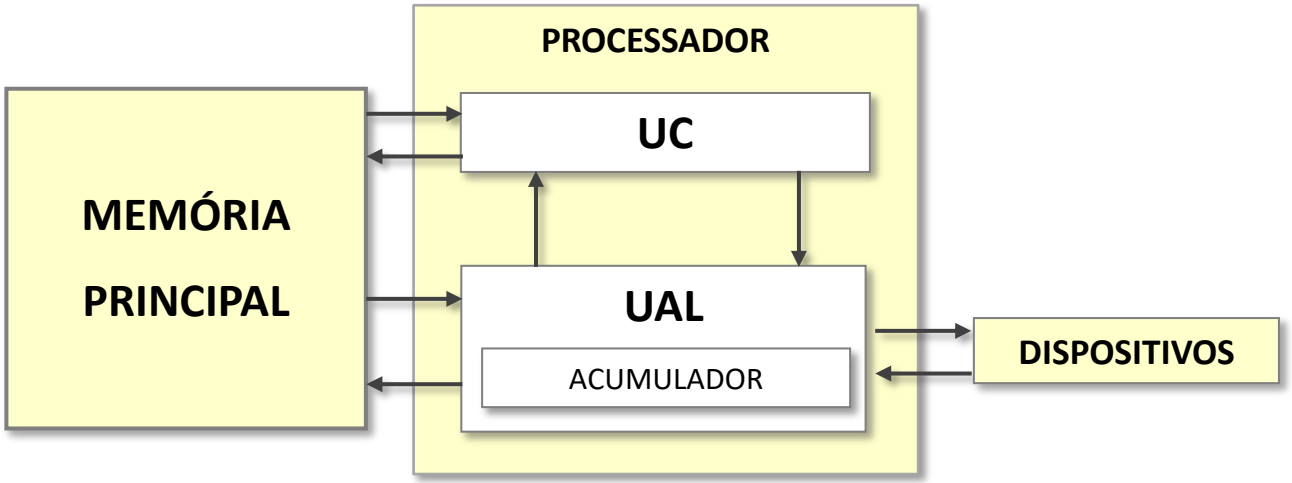
UNIDADE CENTRAL DE PROCESSAMENTO

- Unidades funcionais de um processador
- Registradores
- Instrução
- Formato de Instrução
- Linguagem Assembler
- Como o Processador funcional
- Execução de um programa

Arquitetura de Computadores

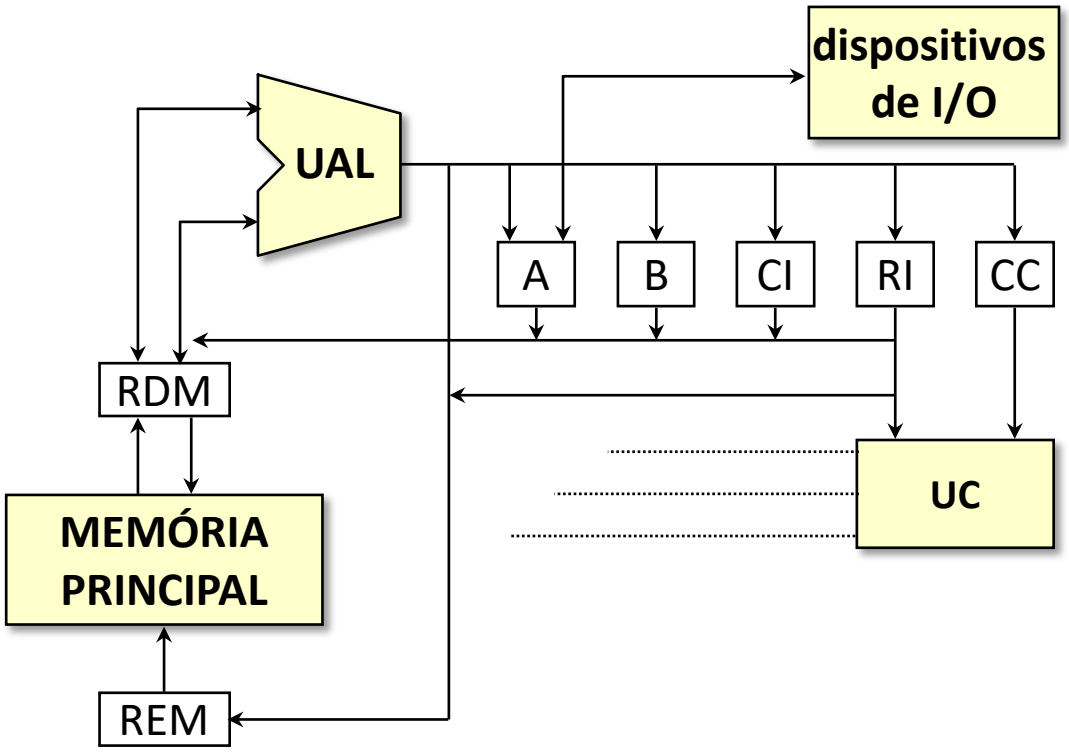
A arquitetura do computador moderno, proposta pelo cientista John von Neumann, caracteriza-se pela possibilidade de uma máquina digital armazenar os programas e seus dados na mesma memória principal, possibilitando a manipulação direta pelo **processador**.

John von Neumann (1903-1957)
(matemático e físico)
O arquiteto do computador moderno.



UCP – Unidade Central de Processamento

Organização estrutural de um computador digital segundo John *von Neumann*



UC **:**Unidade de Controle - Responsável pela decodificação das instruções e envio de sinais de controle.

UAL : Unidade Aritmética e Lógica - Responsável pela execução das operações aritméticas e lógicas.

REM : Registrador de Endereço de Memória

RDM : Registrador de Dados da Memória

UAL : Unidade Aritmética e Lógica

RI : Registrador de Instrução - Contém a palavra que será decodificada e interpretada pela Unidade de Controle

CI : Contador de Instruções - Contém o endereço da próxima instrução a ser executada.

CC : Código de Condição - bits de condição que refletem o resultado da ultima operação executada pela UAL

A,B : Registradores de uso Geral para operações aritméticas e lógicas.

EXERCÍCIOS:

DADOS:
CI = 34bits
M = 8bits

PERGUNTA:
N = xG
T = xK

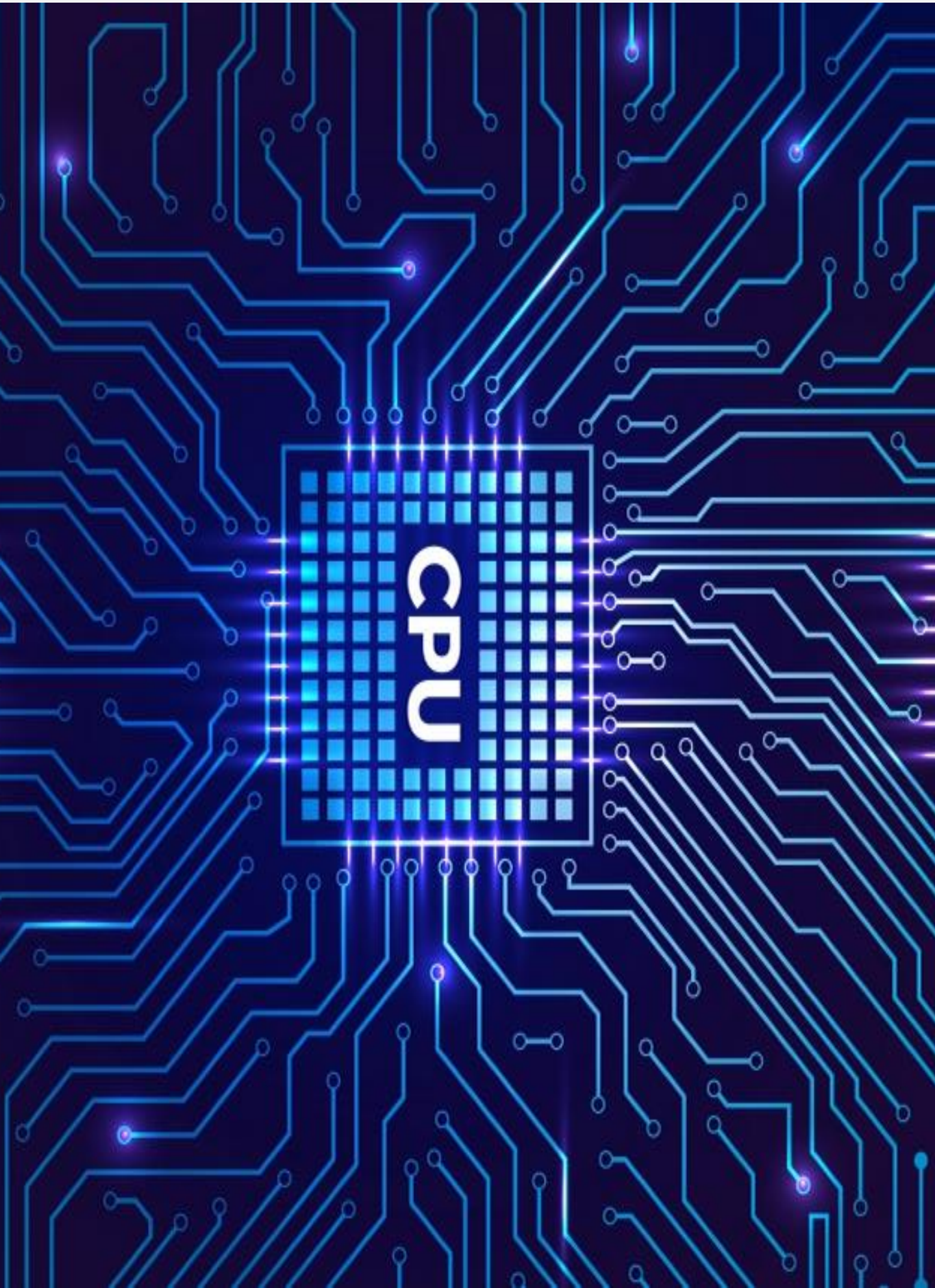
RESPOSTA 1:
N = 2^L
N = 2^34

N em Giga = **2^34 / 2^30 GBytes**

RESPOSTA 2:
T = N * M
T = 2^34 * 8

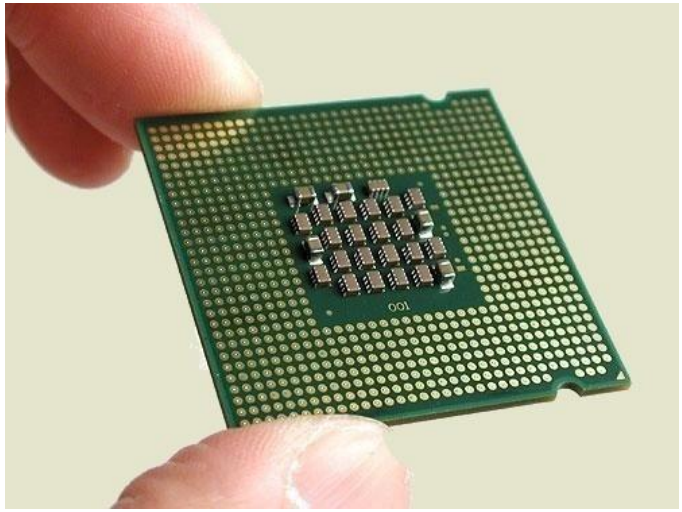
T em Kilo = **2^34 * 8 / 2 ^10 Kbits**

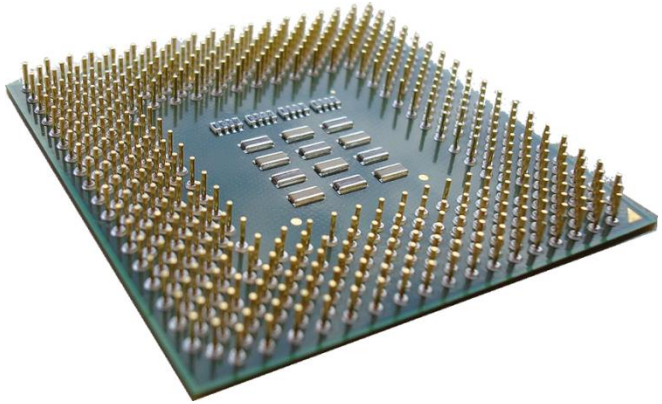
UNIDADE	SIGLA	TAMANHO
bit	b	0 ou 1
Byte	B	8 bits
Kilo	K	2^10
Mega	M	2^20
Giga	G	2^30
Tera	T	2^40
Peta	P	2^50
Exa	E	2^60
Zetta	Z	2^70


CPU

Unidade Central de Processamento

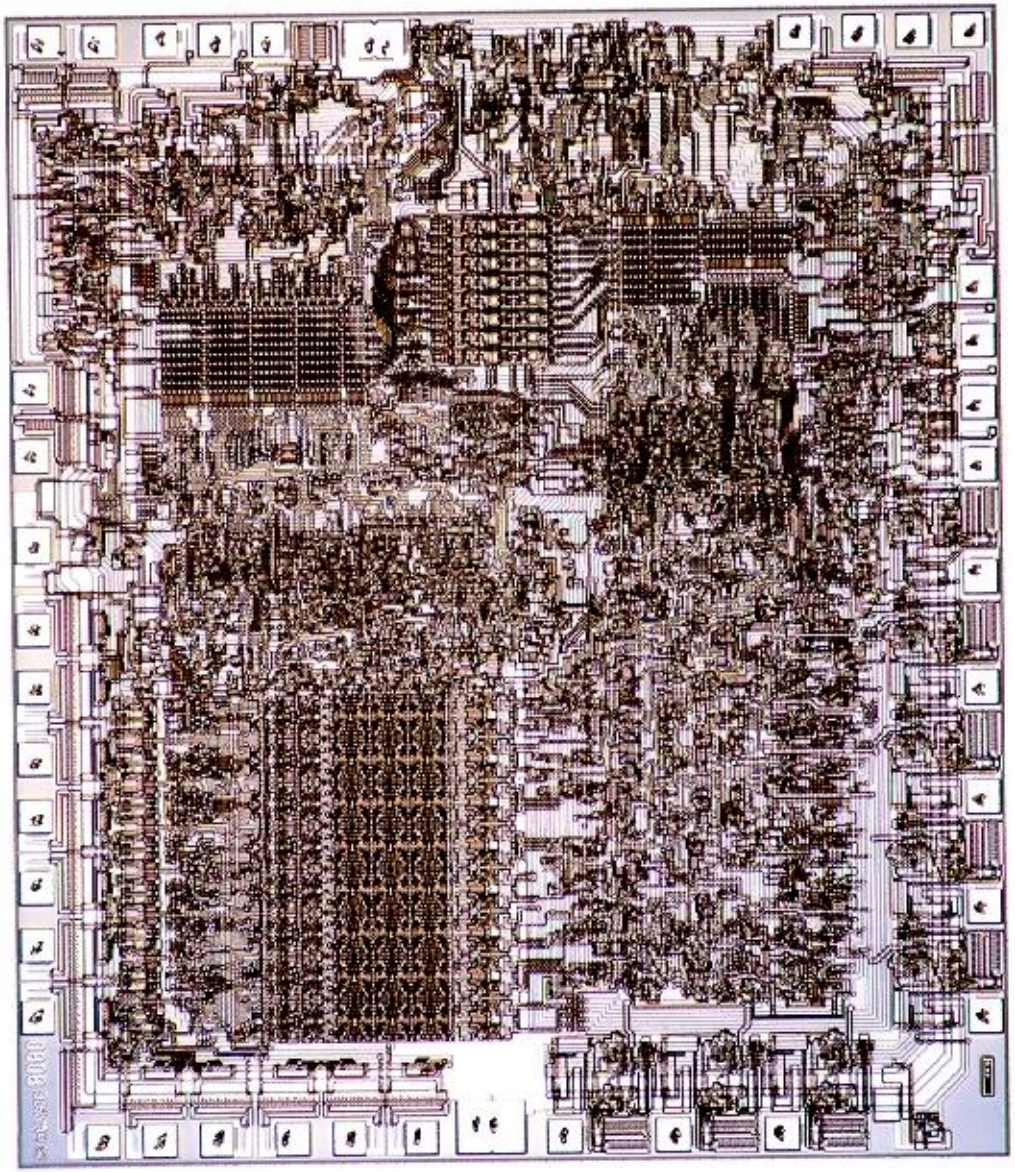
A **Unidade Central de Processamento** ou **CPU** (*Central Processing Unit*), também conhecida como **processador**, é o componente de hardware de um sistema computacional, responsável pela execução das instruções de um programa, para realizar operações aritmética básica, operações lógica e de entrada e saída de dados.





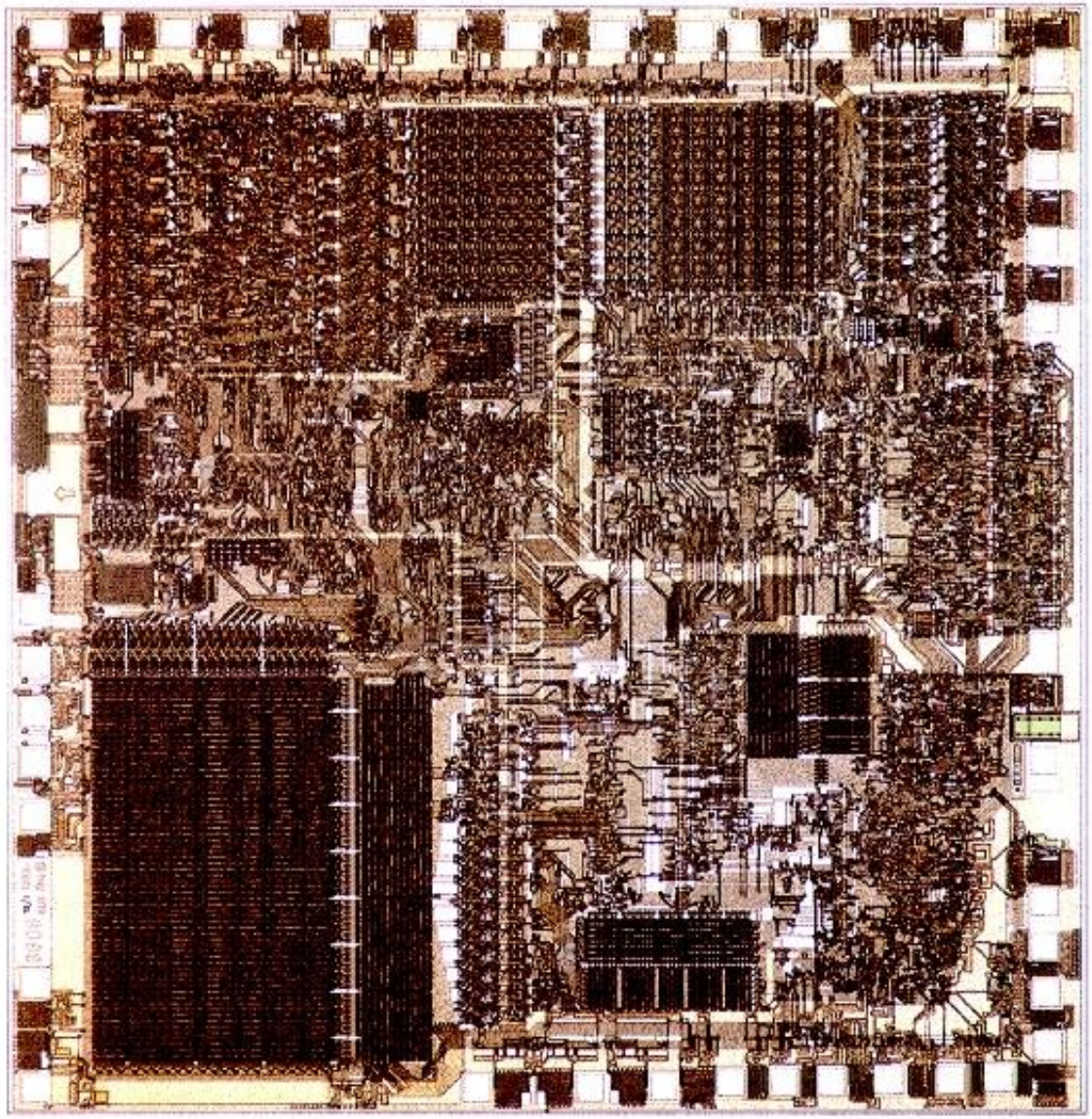
Processador Intel 8080

fonte:http://www.intel.com



8080
Introduction date: April 1974
Clock speed: 2 MHz
0.64 MIPS
Number of transistors: 6,000 (6 microns)
Bus width: 8 bits
Addressable memory: 64 Kbytes
Typical use: Traffic light controller, Altair computer (first PC)
Ten times the performance of the 8008.
Required six support chips versus 20 for the 8008

Processador Intel 8086/8088



fonte:http://www.intel.com

8086

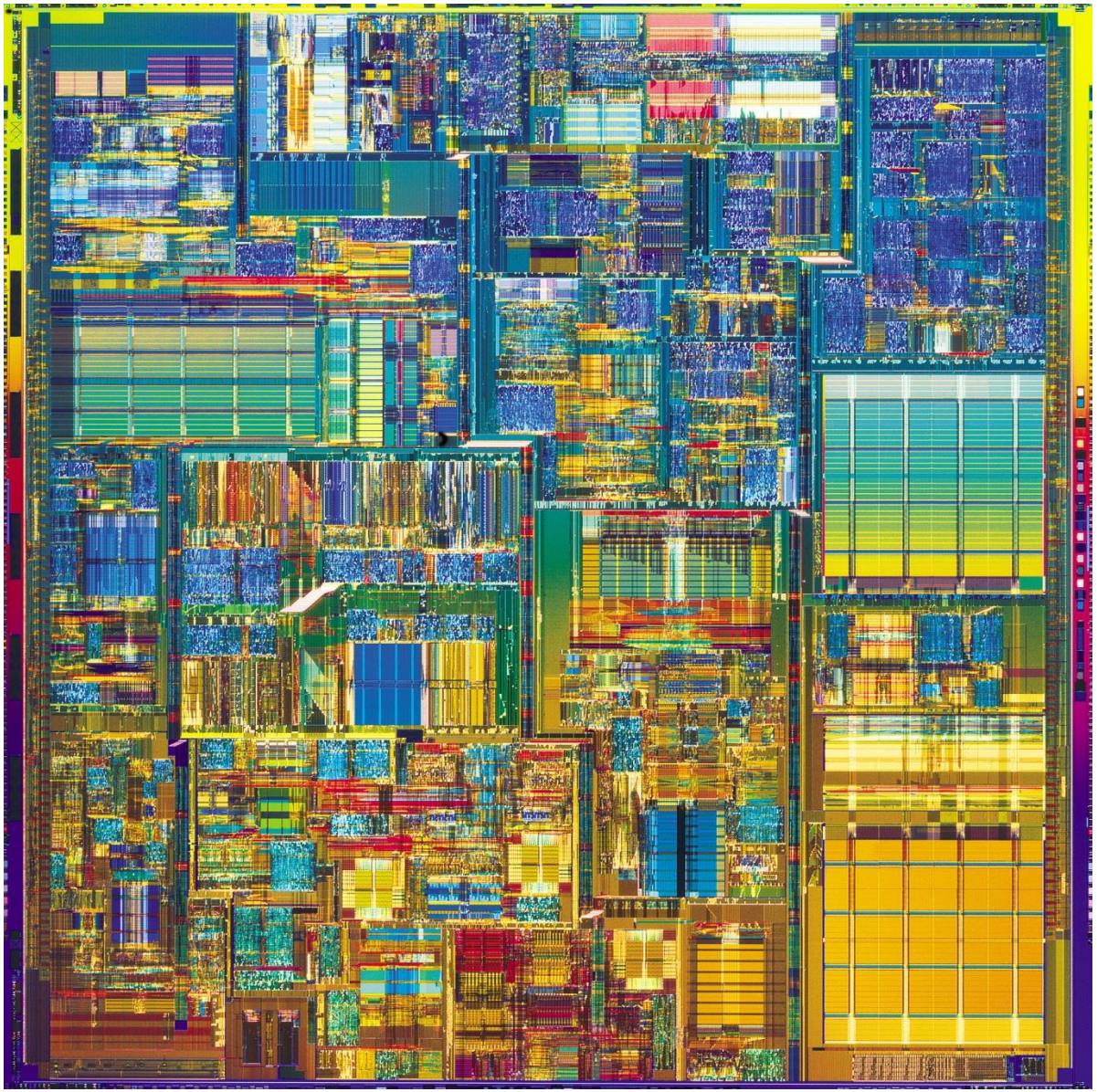
Introduction date: June 8, 1978
Clock speeds: 5 MHz (0.33 MIPS)
8 MHz (0.66 MIPS)
10 MHz (0.75 MIPS)
Number of transistors: 29,000 (3 microns)
Bus width: 16 bits
Addressable memory: 1 Megabyte
Typical use: Portable computing
Ten times the performance of the 8080

8088

Introduction date: June 1979
Clock speeds: 5 MHz (0.33 MIPS)
8 MHz (0.75 MIPS)
Internal architecture: 16 bits
External bus width: 8 bits
Number of transistors: 29,000 (3 microns)
Typical use: Standard microprocessor for all IBM PCs and PC clones
Identical to 8086 except for its 8 bit external bus

Processador Intel Pentium 4

fonte:http://www.intel.com



Pentium® 4

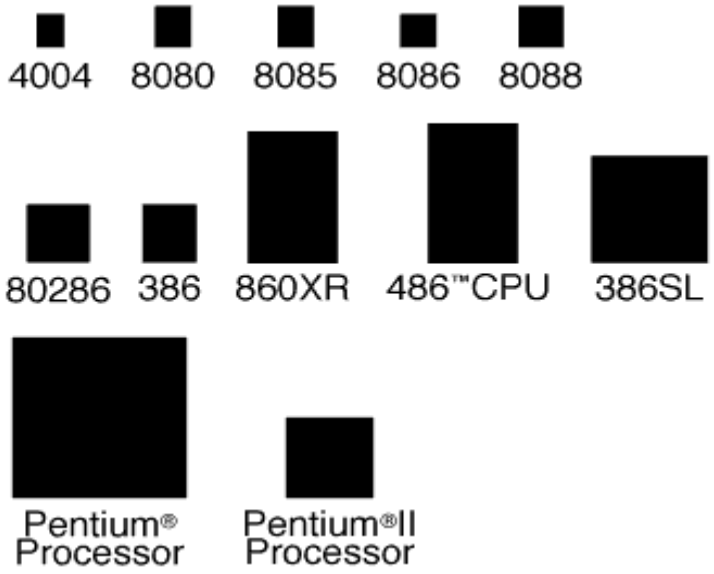
Number of transistors: 42 million transistors and circuit lines of 0.18 microns.

Clock speeds: 1.5 gigahertz (1.5 billion hertz).

fonte:http://www.intel.com

Intel Corporation

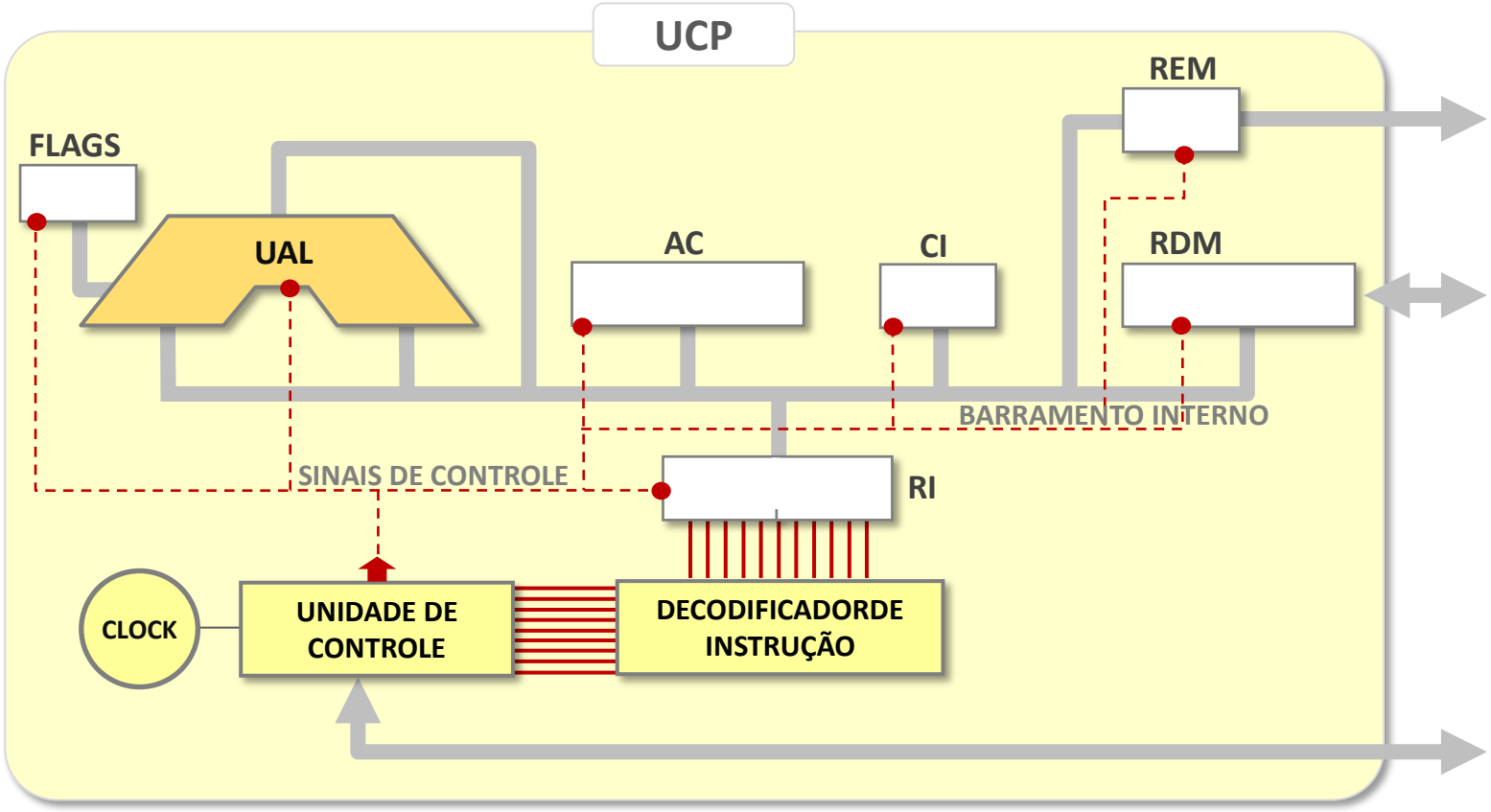
Approximate Size Relationship



Comparação aproximada do tamanho das pastilhas de cilício em cada processador.

Unidade Central de Processamento Ex2020

PROCESSADOR SIMPLIFICADO



UAL – Unidade Aritmética e Lógica
UC – Unidade de Controle
FLAGS – Registrador de estado

AC – Acumulador (Registro de Dados)
CI – Contador de Instrução;
RI – Registrador de Instrução;
REM – Registro de Endereço de Memória;
RDM – Registro de Dados de Memória;

UCP – Unidade Central de Processamento

Unidades funcionais do processador

- **Unidade de Controle**

  Controla as operações de escrita/leitura na memória principal;
  Decodifica instrução;
  Controla operações internas da CPU.
  Controla os sinais externos “interrupções” vindos dos dispositivos.

- **Unidade Aritmética e Lógica**

  Efetua as operações aritméticas e as operações lógicas com os dados.

- **Registradores**

  Memória temporária de alta velocidade para armazenamento de dados e instruções.

- **Clock**

  Gera pulsos para sincronizar as operações internas do processador.

Unidades funcionais

REGISTRADORES

REM: Registro de Endereço de Memória, conectado ao BE;
RDM: Registrador de Dados da Memória, conectado ao BD;
RI: Registrador de Instrução (armazena a instrução em execução)
CI: Contador de Instrução (**endereço** da próxima instrução)
CC: Código de Condição (bits de estado da ultima operação da UAL)
A,B: Registradores DADOS para as operações na UAL. A partir de agora será denominado **AC** e haverá apenas um.

REGISTRADORES ASSOCIADOS AO BARRAMENTO DE ENDEREÇO DA MP:

REM: Registro de Endereço de Memória, conectado ao BE;
CI: Contador de Instrução (endereço da próxima instrução)

REGISTRADORES ASSOCIADOS AO BARRAMENTO DE DADOS:

RDM: Registrador de Dados da Memória, conectado ao BD;
RI: Registrador de Instrução (armazena a instrução em execução)
AC: Registradores dados para as operações na UAL.

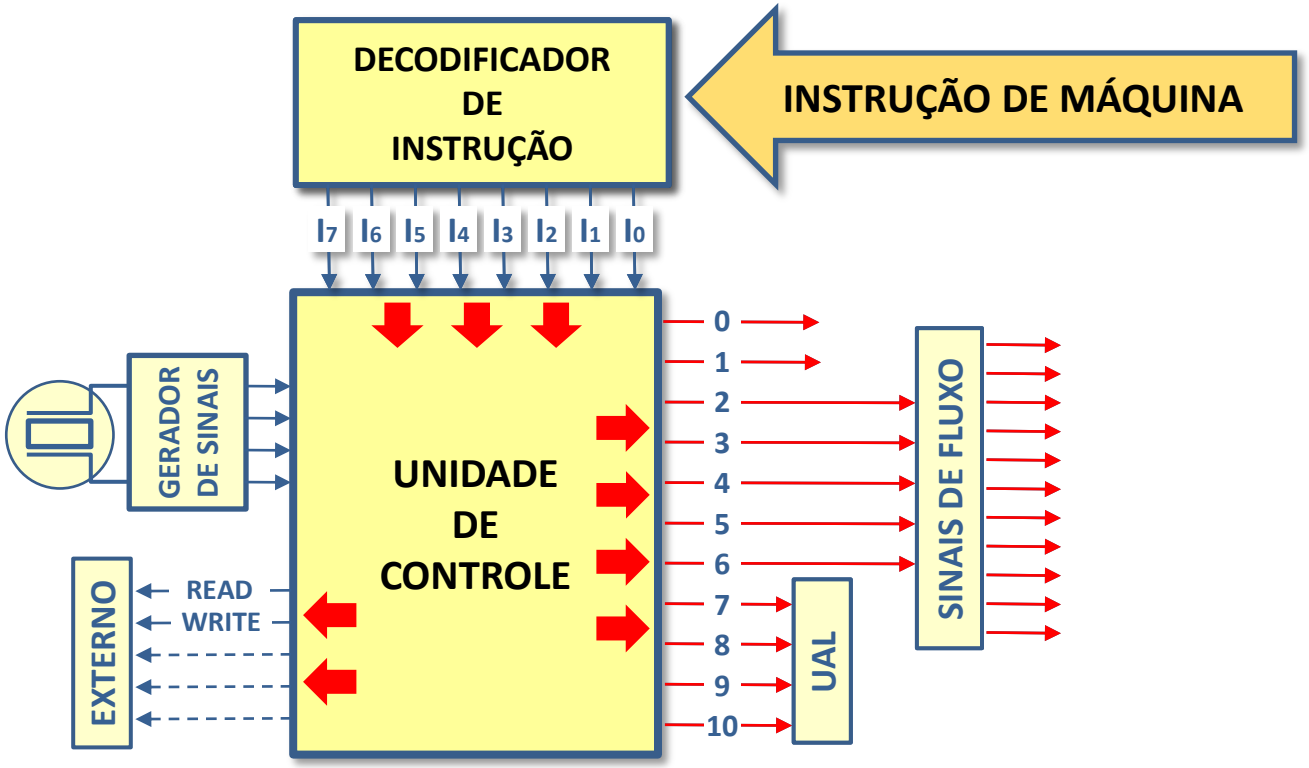
Unidade Central de Processamento

UC – Unidade de Controle

CICLO DE INSTRUÇÃO

Busca → Decodifica → Executa

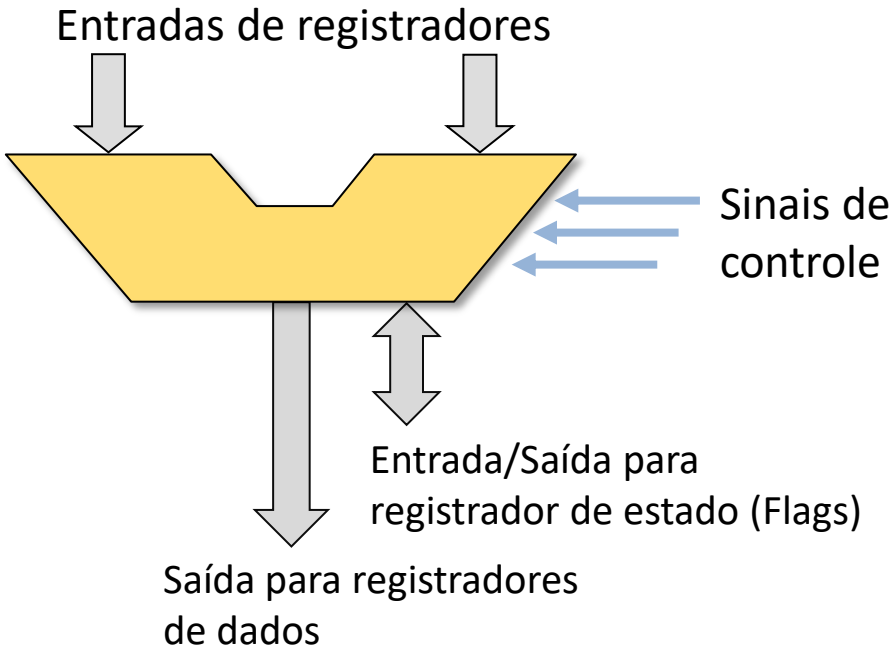
Busca a instrução na MP;
Decodifique a instrução pela UC;
Execute a instrução.



Unidade Central de Processamento

UAL – Unidade Aritmética e Lógica

A UAL contém todos os **circuitos lógicos** (Nível de Máquina) para realizar as operações aritméticas, lógicas e outras operações com os dados.



Operações (instruções):

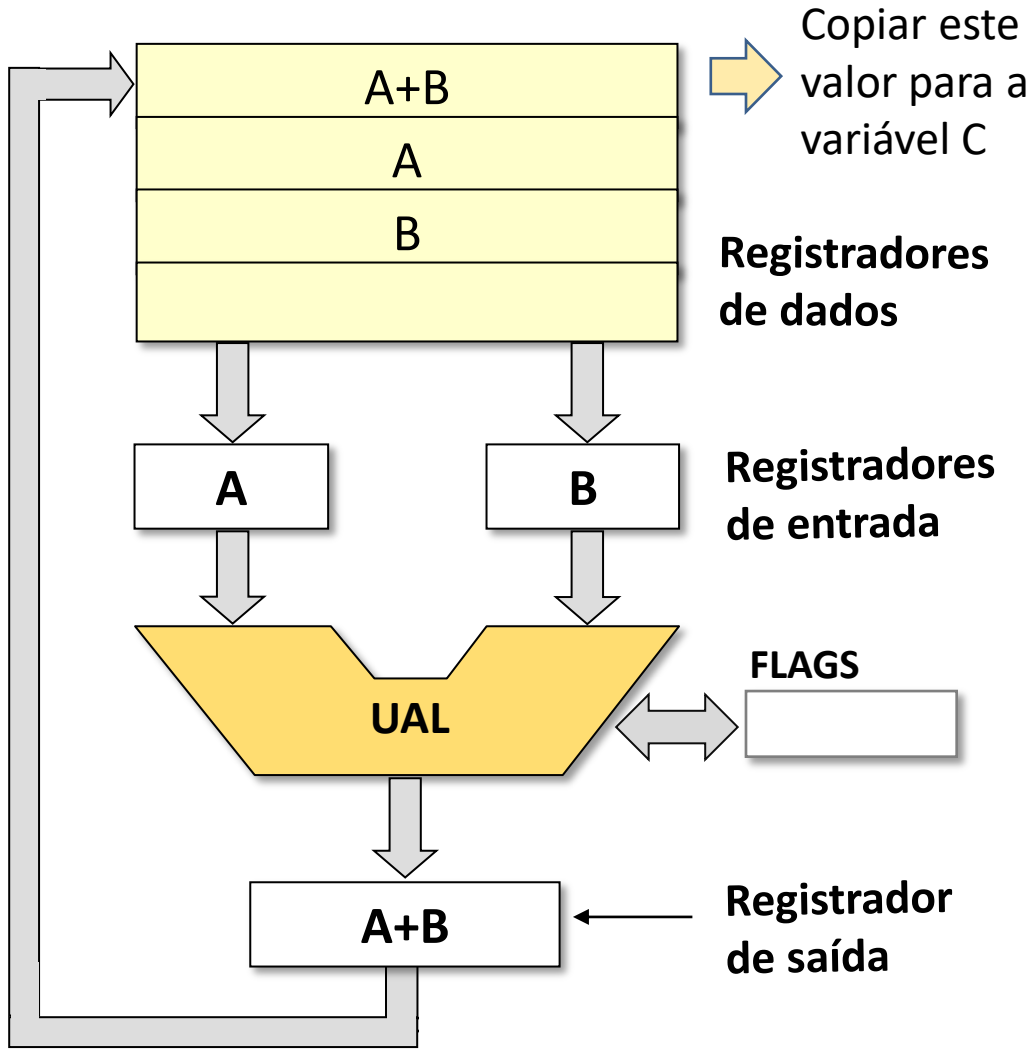
- Soma
- Subtração
- Divisão
- Multiplicação
- Operações lógicas
- Deslocamento
- Incremento
- Complemento

UCP – Unidade Central de Processamento

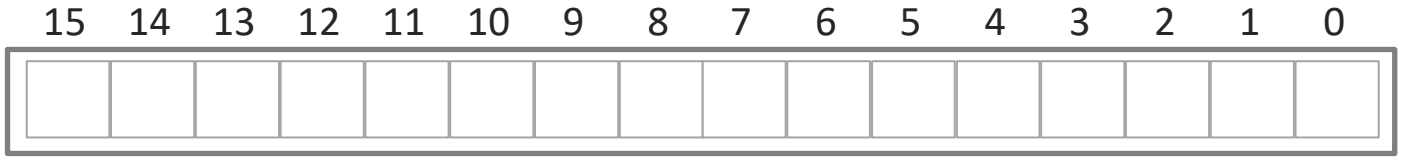
Fluxo de dados na execução de uma expressão aritmética pela UAL

Resolução de uma expressão aritmética simples : **C = A + B**

Some o valor da variável **A** ao valor da variável **B** e armazene o resultado na variável **C**



Registrador de estado “Flags”



Bit		Descrição
0	CF	**Carry Flag**
1		Reserved
2	PF	Parity flag
3		Reserved
4	AF	Adjust flag
5		Reserved
6	ZF	**Zero flag**
7	SF	**Sign flag**
8	TF	Trap flag
9	IF	Interrupt enable flag
10	DF	Direction flag
11	OF	Overflow flag

INSTRUÇÃO DE MÁQUINA

Uma **instrução de máquina** é a formalização de um **comando primitivo** que o processador é capaz de executar diretamente.

O conjunto de todas as instruções do processador constitui a sua "linguagem de máquina"

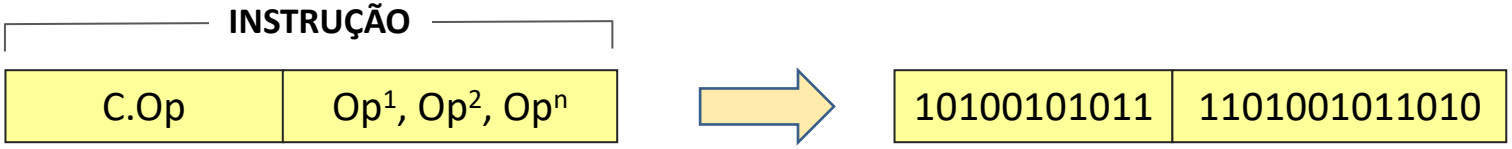
Tipos de operações:

- Operações Aritméticas;
- Operações Lógicas;
- Complemento;
- Deslocamento;
- Movimentação de Dados
  - Leitura na MP;
  - Gravação na MP;
- Entrada / Saída;
- Controle.

INSTRUÇÃO DE MÁQUINA

FORMATO

Toda instrução de máquina possui um formato dividido em dois campos.
O primeiro "**código de operação**" informa qual é a operação e como ela será executada e o segundo campo "**operando**" indica qual o dado envolvido na execução (se existir).



C.Op - Código de Operação - É o campo que contém a identificação da operação a ser executada. Cada operação possui um código identificador único. O tamanho, em bits, deste campo determina o número de instruções que o processador pode executar.

Op - Operando(s) - são os campos que contém o **dado** ou a **localização**, o endereço de memória onde está o dado a serem processado.

INSTRUÇÃO DE MÁQUINA

Não existe um processador que possua uma única instrução capaz de resolver a expressão matemática:

A = B + C ⇦ Escrito no programa de alto nível, ex: C#

Esta expressão é resolvida pela máquina (CPU) através da decomposição, **pelo compilador**, em partes menores, denominadas **instruções primitivas simples**, que combinadas em uma sequência lógica resolvem o problema proposto.

Solução em português:

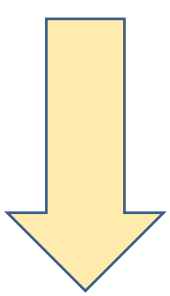
1. Leia o valor de **B** para o registrador **AC**
2. Adicione o valor de **C** ao do registrador **AC** e armazene o resultado em **AC**
3. Grave na variável **A** o valor do registrador **AC**

Sequência de INSTRUÇÕES PRIMITIVAS que o processador irá executar para a resolução do problema.

PROCESSADOR – EXECUÇÃO DE UM PROGRAMA

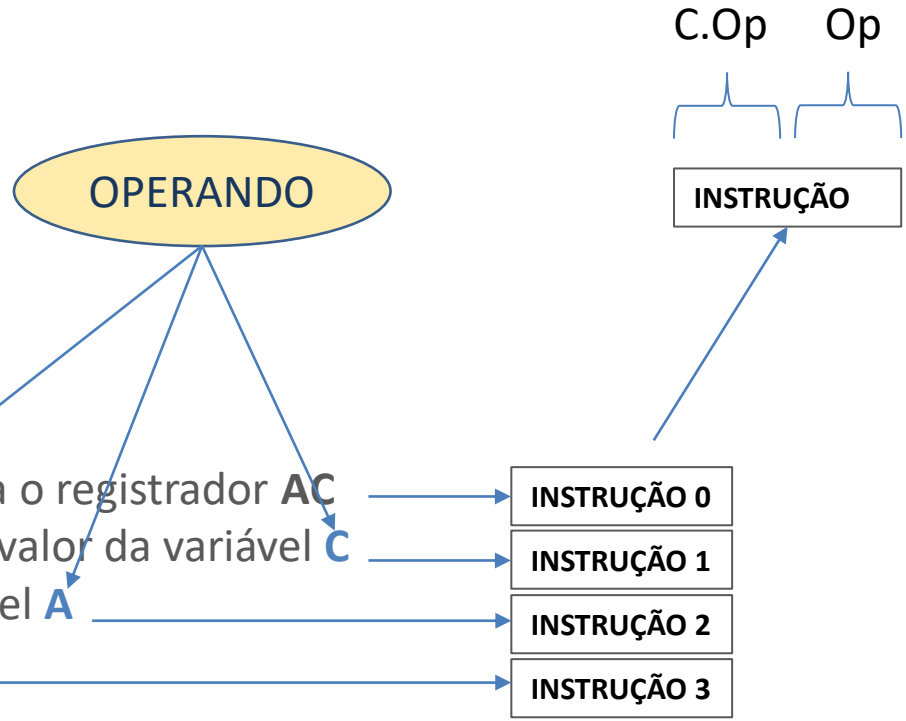
PROGRAMA FONTE

A = B + C



INSTRUÇÕES

1. Leia o valor da variável **B** para o registrador **AC**
2. Adicione ao registrador **AC** o valor da variável **C**
3. Grave o valor de **AC** na variável **A**
4. Finalize a execução

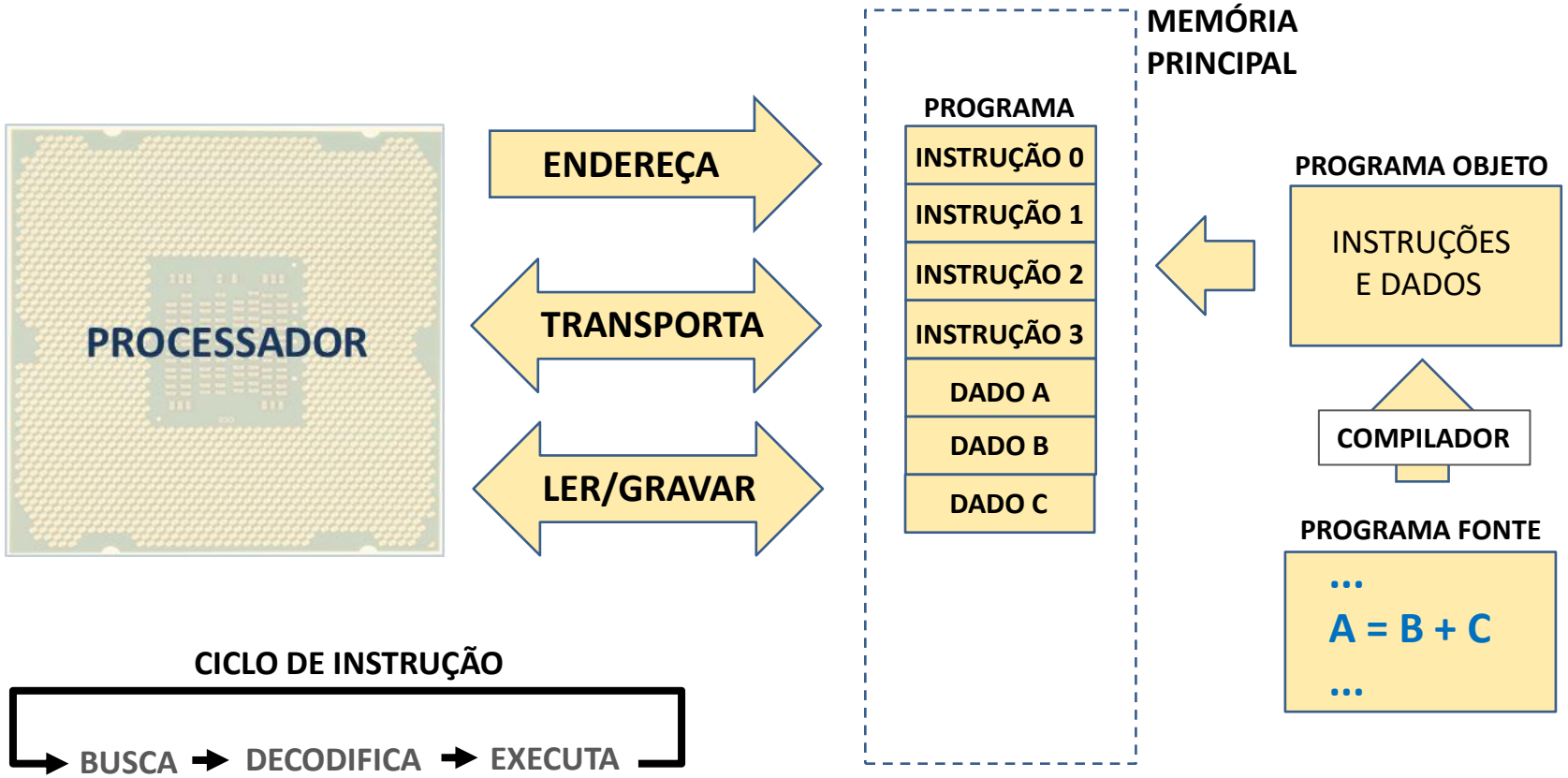


PROCESSADOR E O PROGRAMA NA MEMÓRIA PRINCIPAL

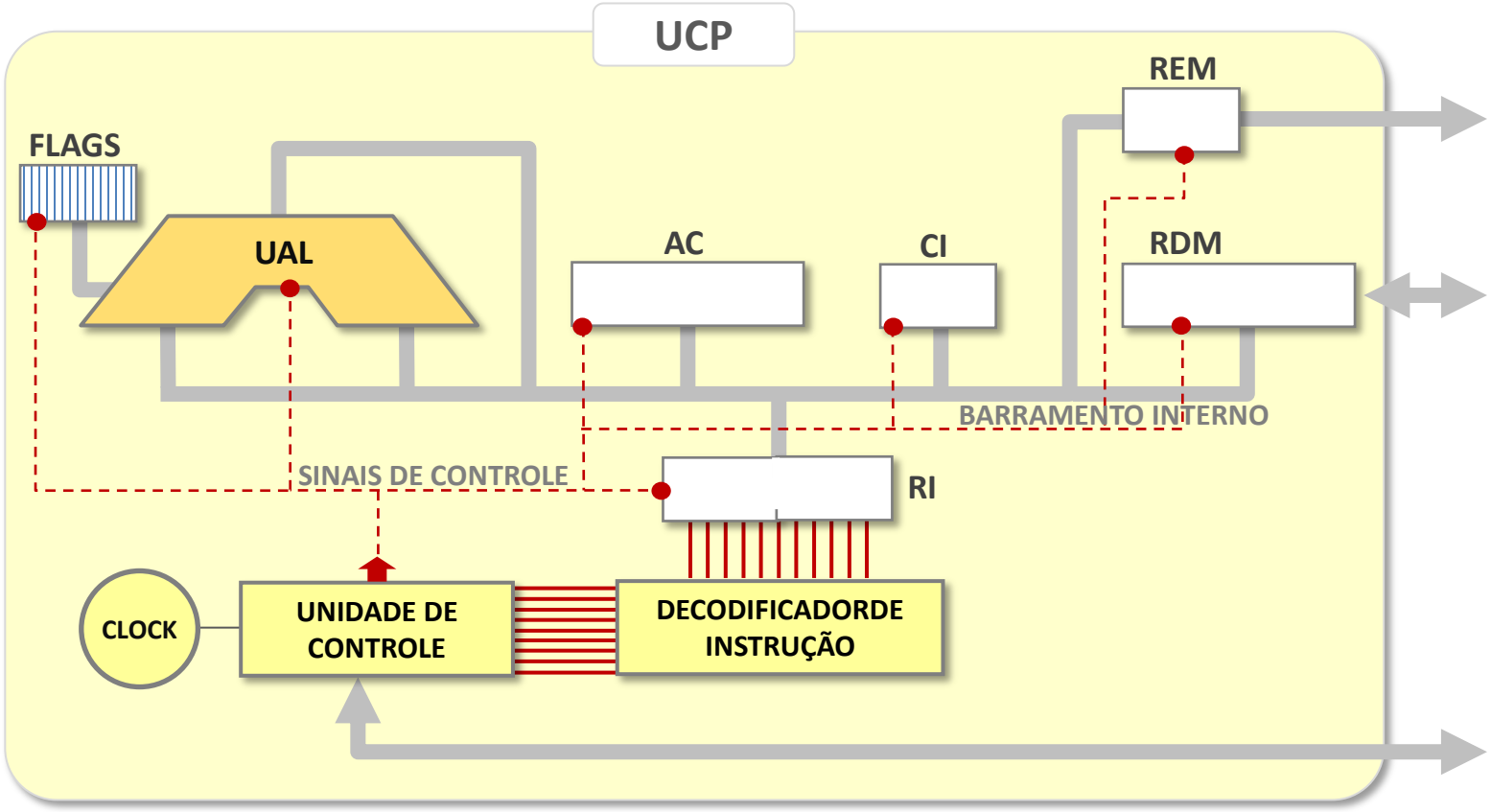
O processador é o “cérebro” do computador

Sua função é executar as instruções do programa e processar seus dados.

Um **programa de computador** é um conjunto de **instruções de máquina** e **dados** que descrevem de forma lógica uma tarefa a ser realizada por um computador.



PROCESSADOR



UAL – Unidade Aritmética e Lógica
UC – Unidade de Controle
FLAGS – Registrador de estado

AC – Acumulador (Registro de Dados)
CI – Contador de Instrução;
RI – Registrador de Instrução;
REM – Registro de Endereço de Memória;
RDM – Registro de Dados de Memória;

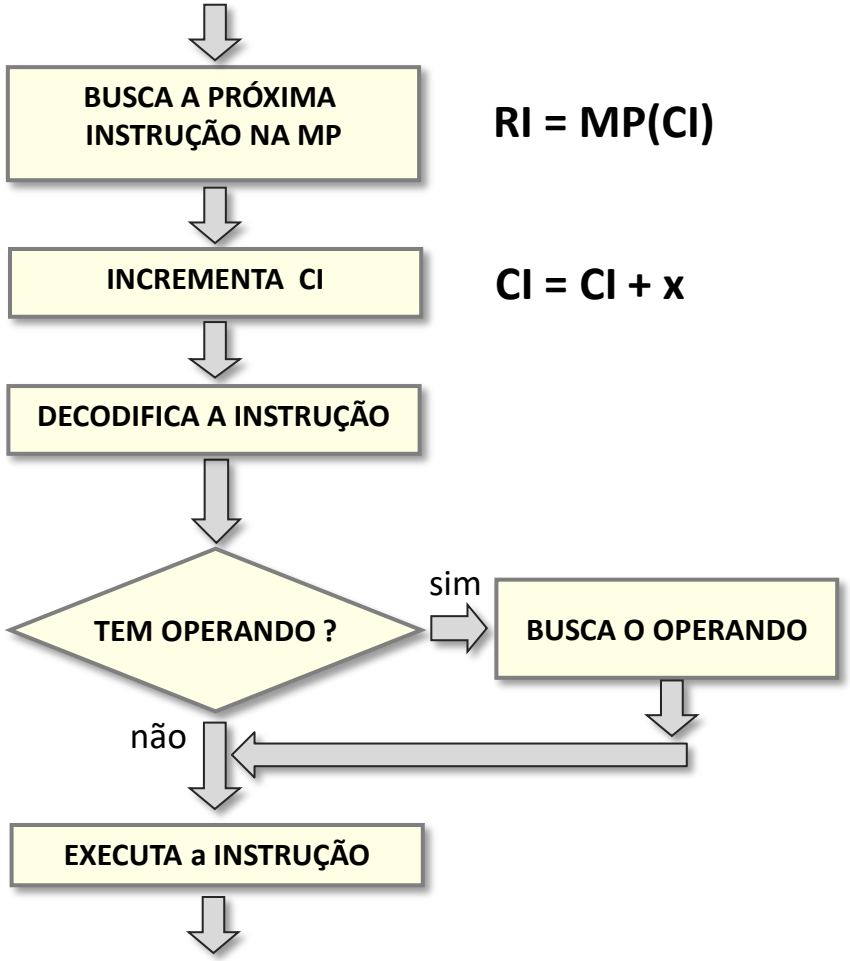
Unidade Central de Processamento

CICLO DE INSTRUÇÃO

A unidade de controle executa três ações básicas intrínsecas e pré-programadas pelo fabricante do processador para a execução de uma instrução, são elas: busca, decodificação e execução.

Assim sendo, todo processador, ao executar de um programa, realiza uma operação cíclica para cada instrução "comando" do programa, tendo como base essas três ações. Este ciclo é denominado "**ciclo de instrução**"

10100101011	1101001011010



Unidade Central de Processamento

ESPECIFICAÇÃO TÉCNICA:

- Tamanho de Palavra = 16 bits;
- Registros de endereço = 8 bits (256 células de memória);
- Tamanho da célula de memória = 16 bits;
- Operações somente com inteiros;
- Formato das instruções: um código de operação e um operando.

C.Op = 8 bits Op[1 =] = 8 bits

REGISTRADORES:

DESCRIÇÃO	SIGLA	TAMANHO
Registro de Dados	AC	16 bits
Registro de Instrução	RI	16 bits
Contador de Instrução	CI	8 bits
Registro de Endereço de Memória	REM	8 bits
Registro de Dados de Memória	RDM	16 bits

LINGUAGEM ASSEMBLER

Na programação de computadores , a **linguagem Assembly** (ou **linguagem assembler**), é uma linguagem de programação de baixo nível na qual existe uma correspondência muito forte entre as **instruções da linguagem** e as **instruções de máquina**.

A linguagem Assembly usa um **mnemônico** para representar cada instrução de máquina.
mnemônico = fácil de ser memorizado

Exemplo:

LDA = LOAD - Carregar um dado da MP para um registro de dado do processador.

LDA B Leia na MP o conteúdo da variável B e carregue no registrador de dado

OPERANDO

CÓDIGO DE OPERAÇÃO

Unidade Central de Processamento

CONJUNTO DAS INSTRUÇÕES

CÓDIGO DE OPERAÇÃO		SIGLA	DESCRIÇÃO
BINÁRIO	HEX		
00000001	01	END	Finaliza a execução do programa
00000010	02	LDA Op	Move um valor (palavra) da memória principal para o registrador AC. AC = MP(Op)
00000011	03	STR Op	Move o valor (palavra) do registrador AC para a memória principal . MP(Op) = AC
00000100	04	ADD Op	Soma ao conteúdo de AC o valor (palavra) da posição de memória OP e coloca o resultado em AC. AC = AC + MP(Op)
00000101	05	SUB Op	Subtrai do conteúdo de AC o valor (palavra) da posição de memória de OP e coloca o resultado em AC. AC = AC - MP(Op)
00000110	06	JZ Op	Se AC = 0, então coloque em CI o valor de Op. CI = Op
00000111	07	JP Op	Se o valor de AC > 0, então coloque em CI o valor de Op. CI =Op
00001000	08	JN Op	Se o valor de AC < 0, então coloque em CI o valor de Op. CI =Op
00001001	09	JMP Op	Atribui o valor de Op ao registro CI.
00001010	0A	GET Op	Lê um valor (palavra) da porta de entrada e armazena no endereço de memória Op. MP(Op) = (entrada)
00001011	0B	PRT Op	Envia para o vídeo o valor (palavra) contido no endereço de memória Op. (saída) = MP(Op)

Exemplo de um programa em linguagem Assembler

Execute a expressão **A = B + C** em linguagem assemble

Programa em português

1. Leia o valor da variável B para o registrador **AC**
2. Adicione ao registrador **AC** o valor da variável C
3. Grave o valor de **AC** na variável A

Programa em assemble

```
LDA B
ADD C
STR A
```

Programa para o Processador Ex.?

O PROBLEMA:

A = B + C

A LÓGICA: (processador Ex.)

1. Copie da MP(B) para o registrador AC;
2. Some o conteúdo de MP(C) ao valor de AC;
3. Copie o valor de AC para MP(A).

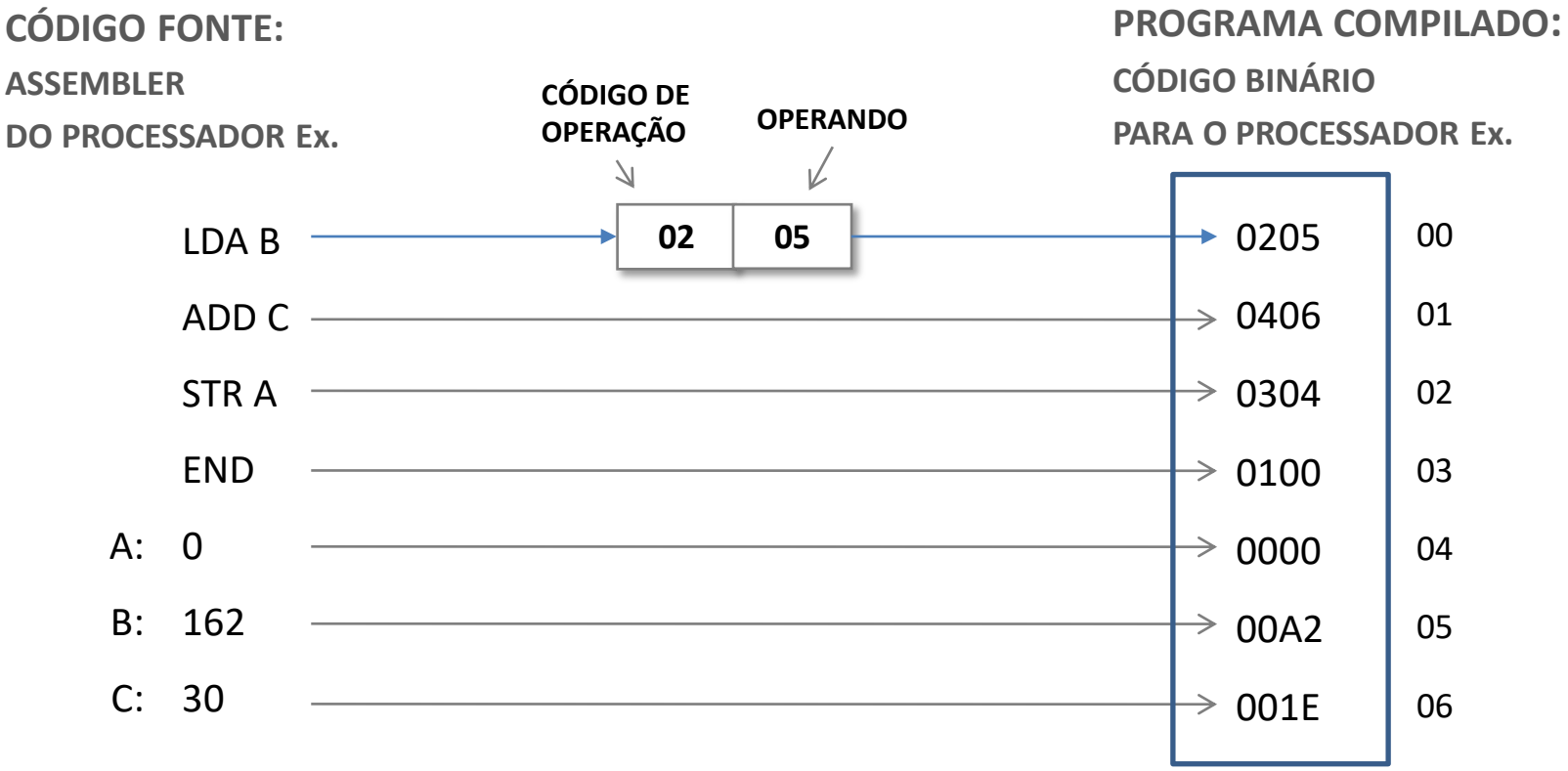
CÓDIGO FONTE:

ASSEMBLE

DO PROCESSADOR Ex.

LDA B	Linha 0
ADD C	Linha 1
STR A	Linha 2
END	Linha 3
A: 0	Linha 4
B:162	Linha 5
C:30	Linha 6

Compilação do Programa com o compilador Ex.



Execução do Programa pelo Processador Ex.

Carregamento

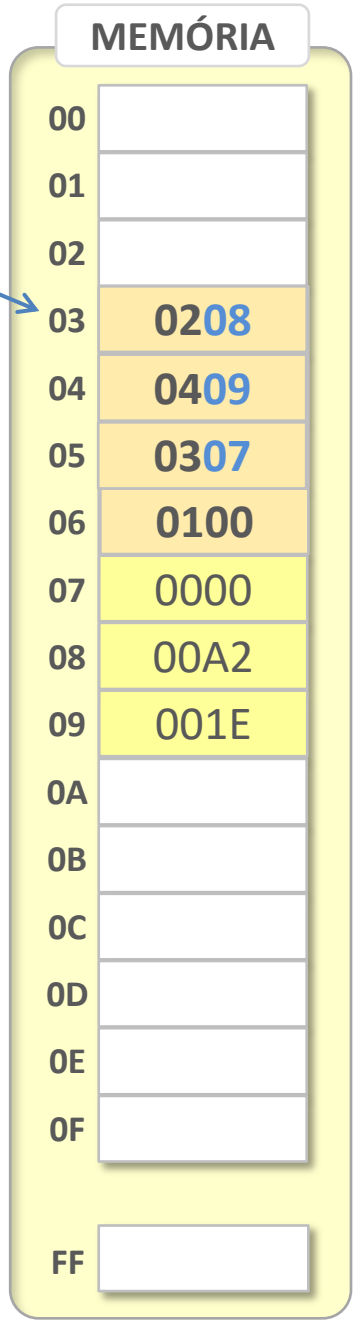
PROGRAMA COMPILADO:
CÓDIGO BINÁRIO
PARA O PROCESSADOR Ex.

INSTRUÇÕES E DADOS	ENDEREÇO RELATIVO
0205	00
0406	01
0304	02
0100	03
0000	04
00A2	05
001E	06

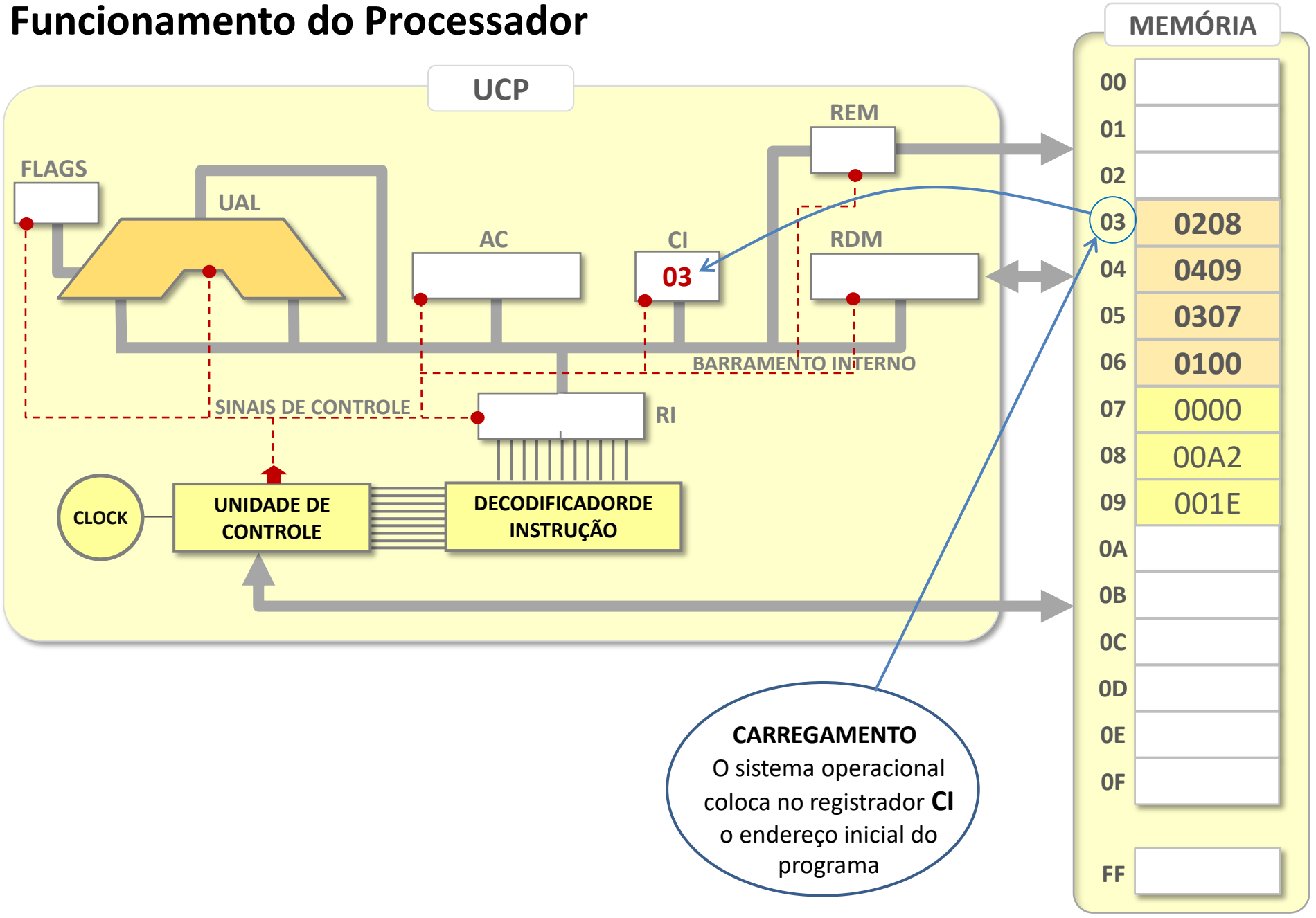
ARQUIVO NA MEMÓRIA SECUNDÁRIA

LDA B

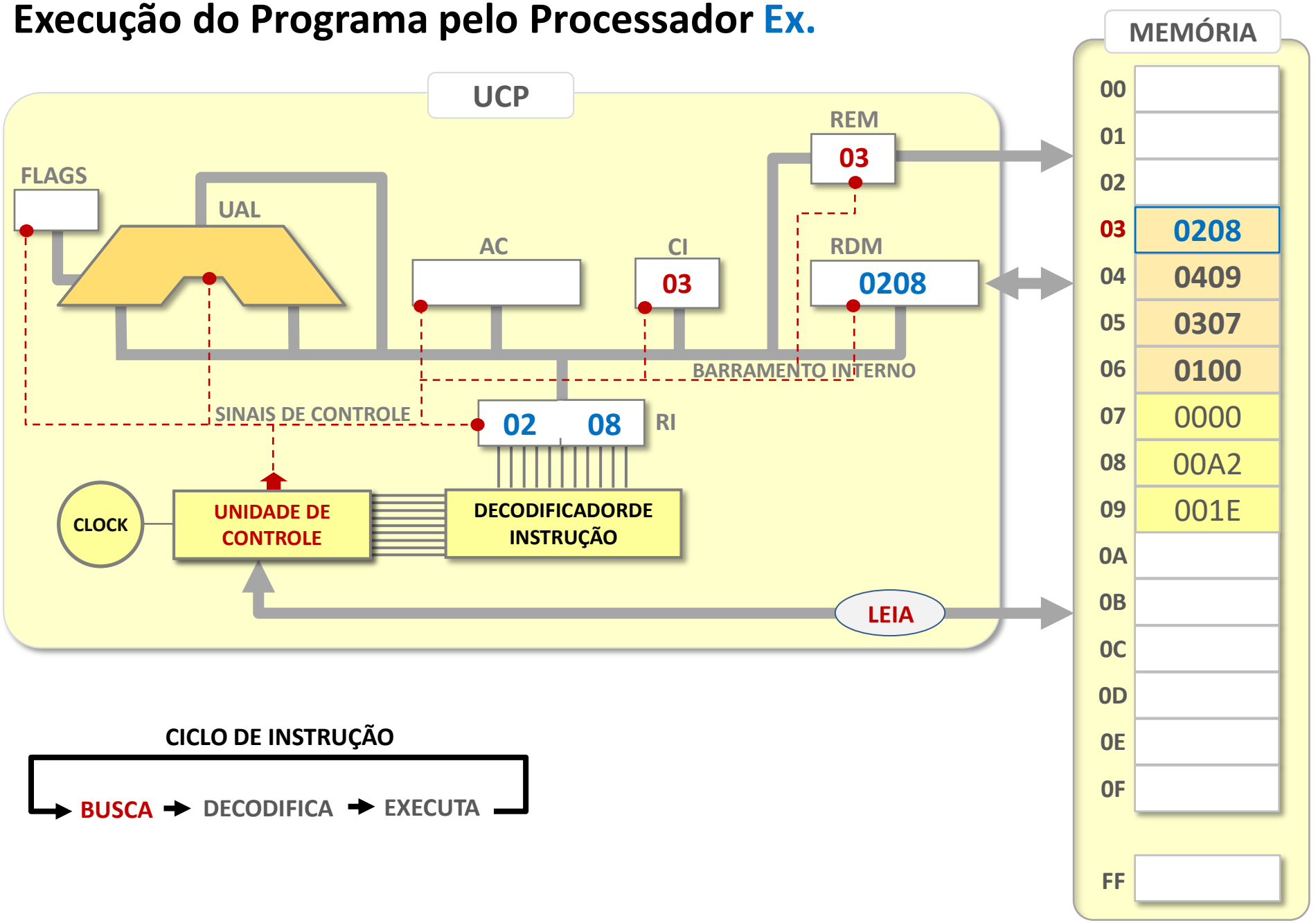
O **SISTEMA OPERACIONAL** carrega o programa em um espaço livre da Memória Principal e recalcula os endereços relativos.



Funcionamento do Processador



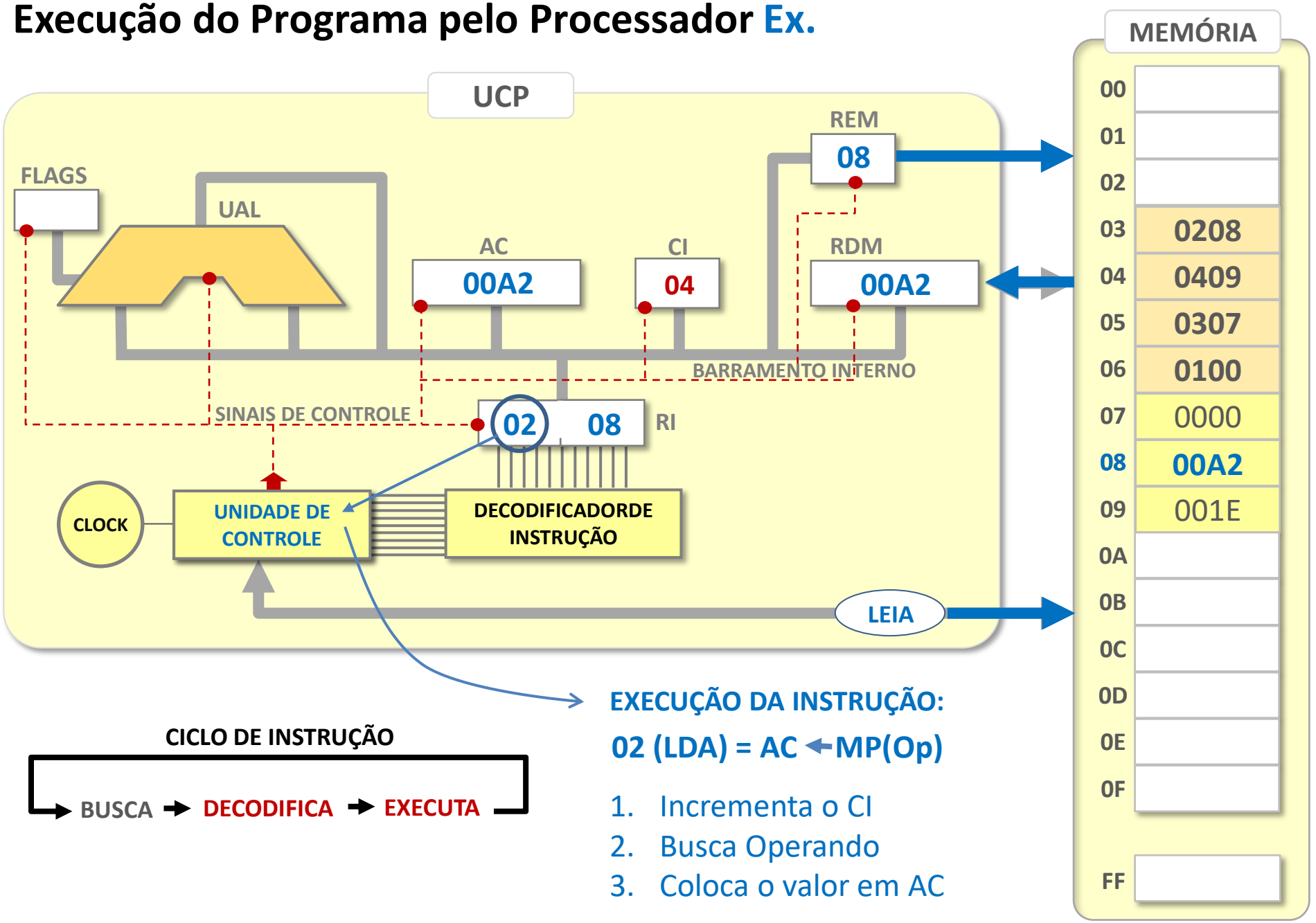
Execução do Programa pelo Processador Ex.



CICLO DE INSTRUÇÃO

BUSCA → DECODIFICA → EXECUTA

Execução do Programa pelo Processador Ex.



CICLO DE INSTRUÇÃO

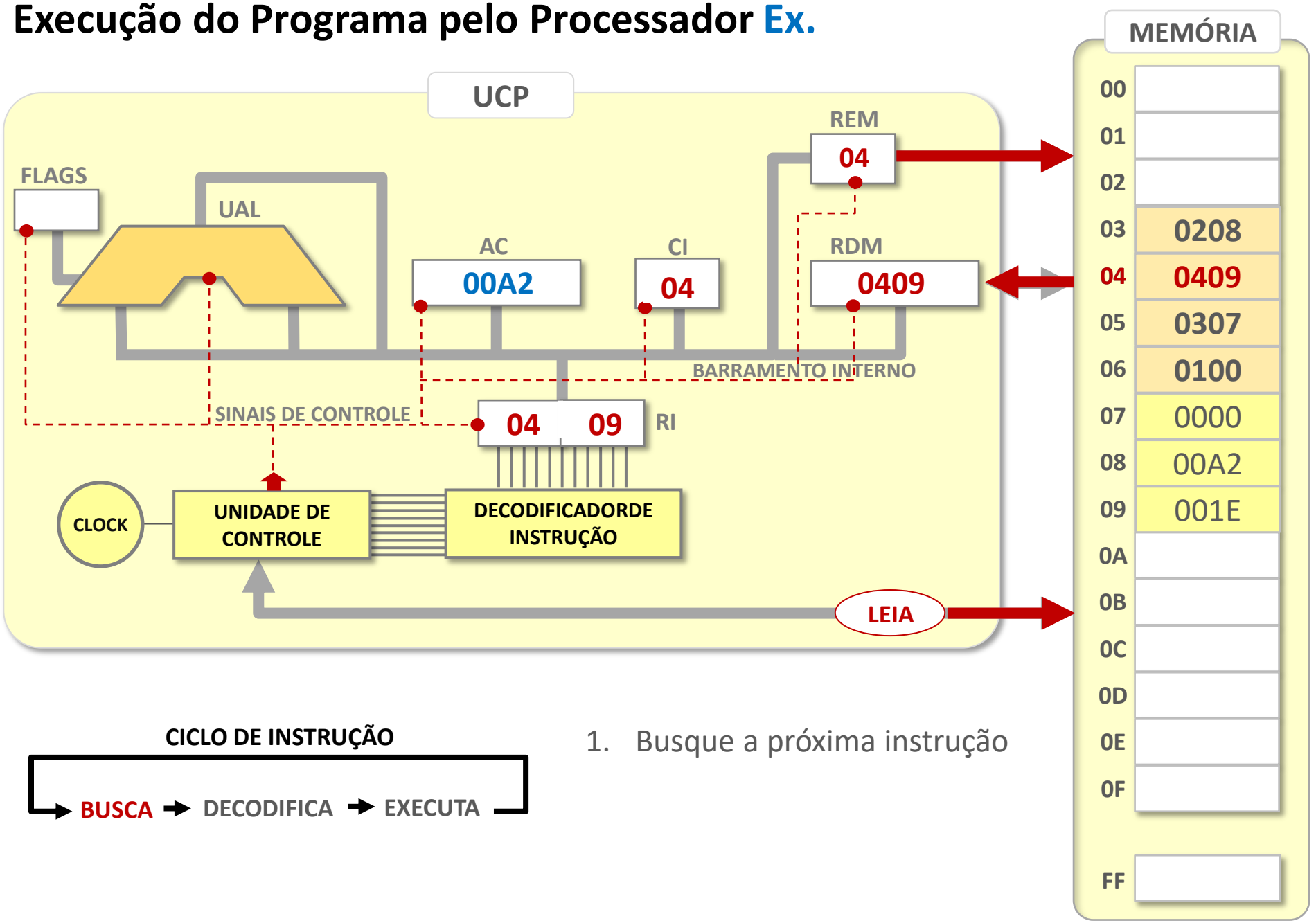
BUSCA → DECODIFICA → EXECUTA

EXECUÇÃO DA INSTRUÇÃO:

02 (LDA) = AC ← MP(Op)

1. Incrementa o CI
2. Busca Operando
3. Coloca o valor em AC

Execução do Programa pelo Processador Ex.

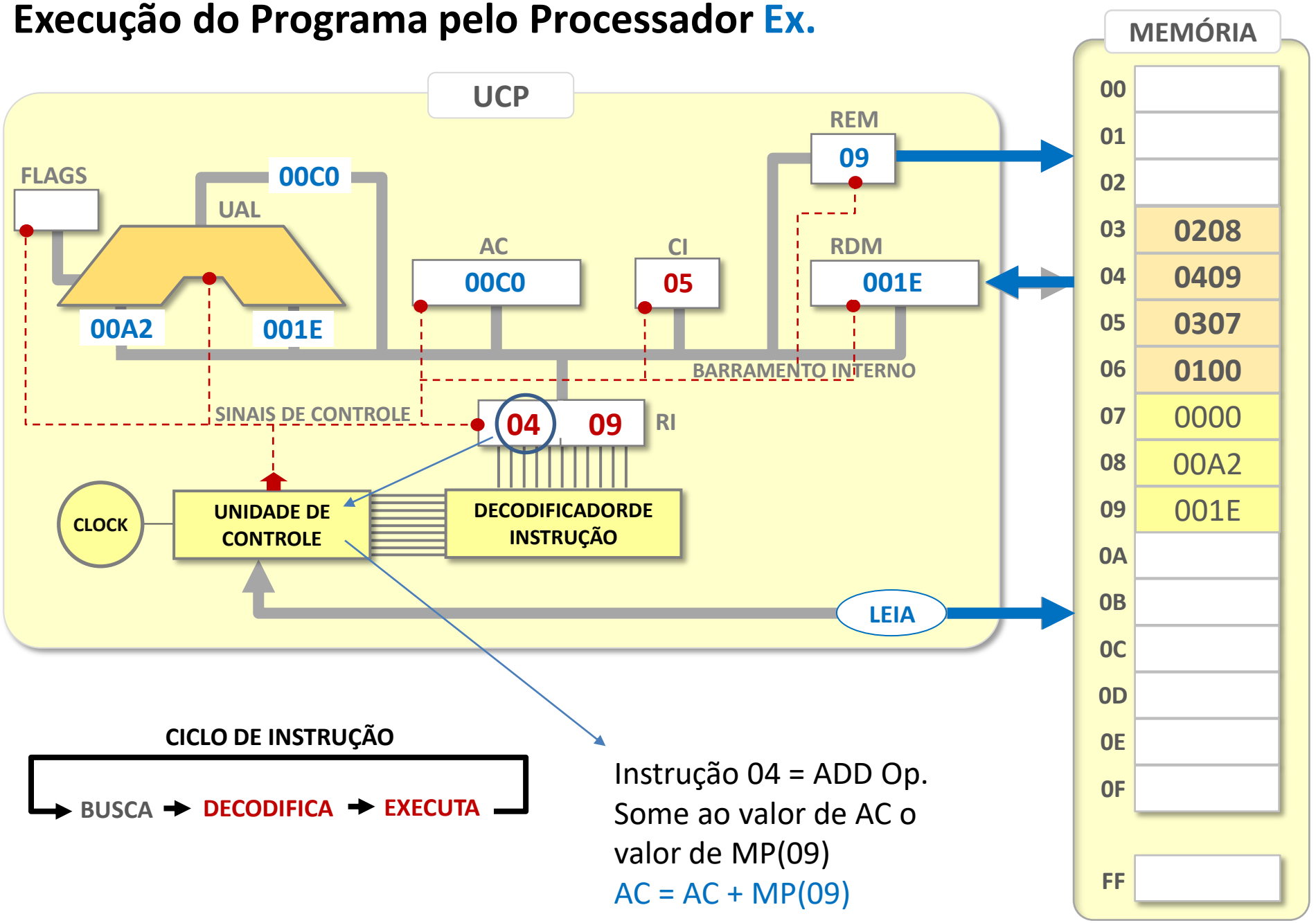


CICLO DE INSTRUÇÃO

BUSCA → DECODIFICA → EXECUTA

1. Busque a próxima instrução

Execução do Programa pelo Processador Ex.

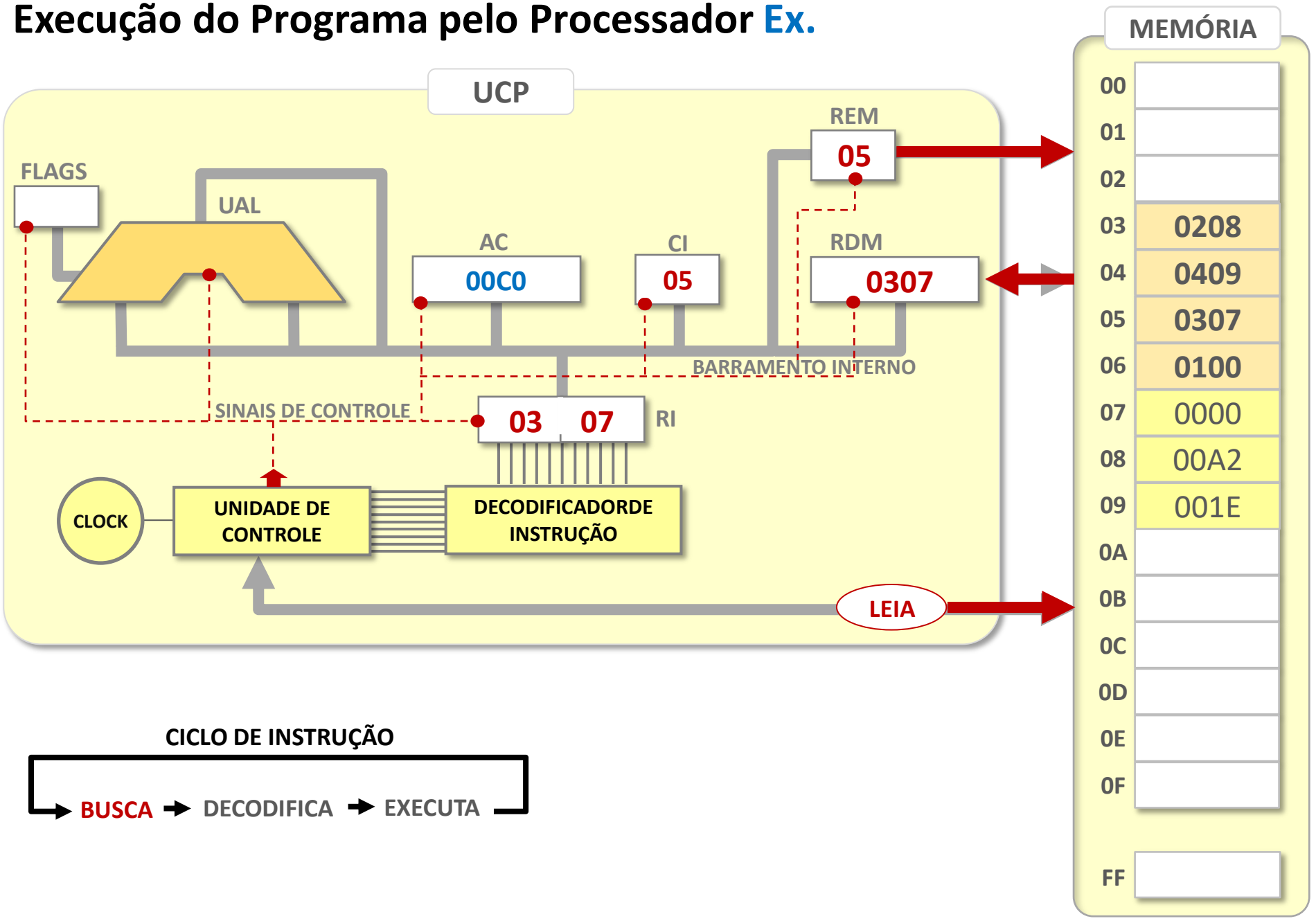


CICLO DE INSTRUÇÃO

BUSCA → DECODIFICA → EXECUTA

Instrução 04 = ADD Op.
Some ao valor de AC o valor de MP(09)
AC = AC + MP(09)

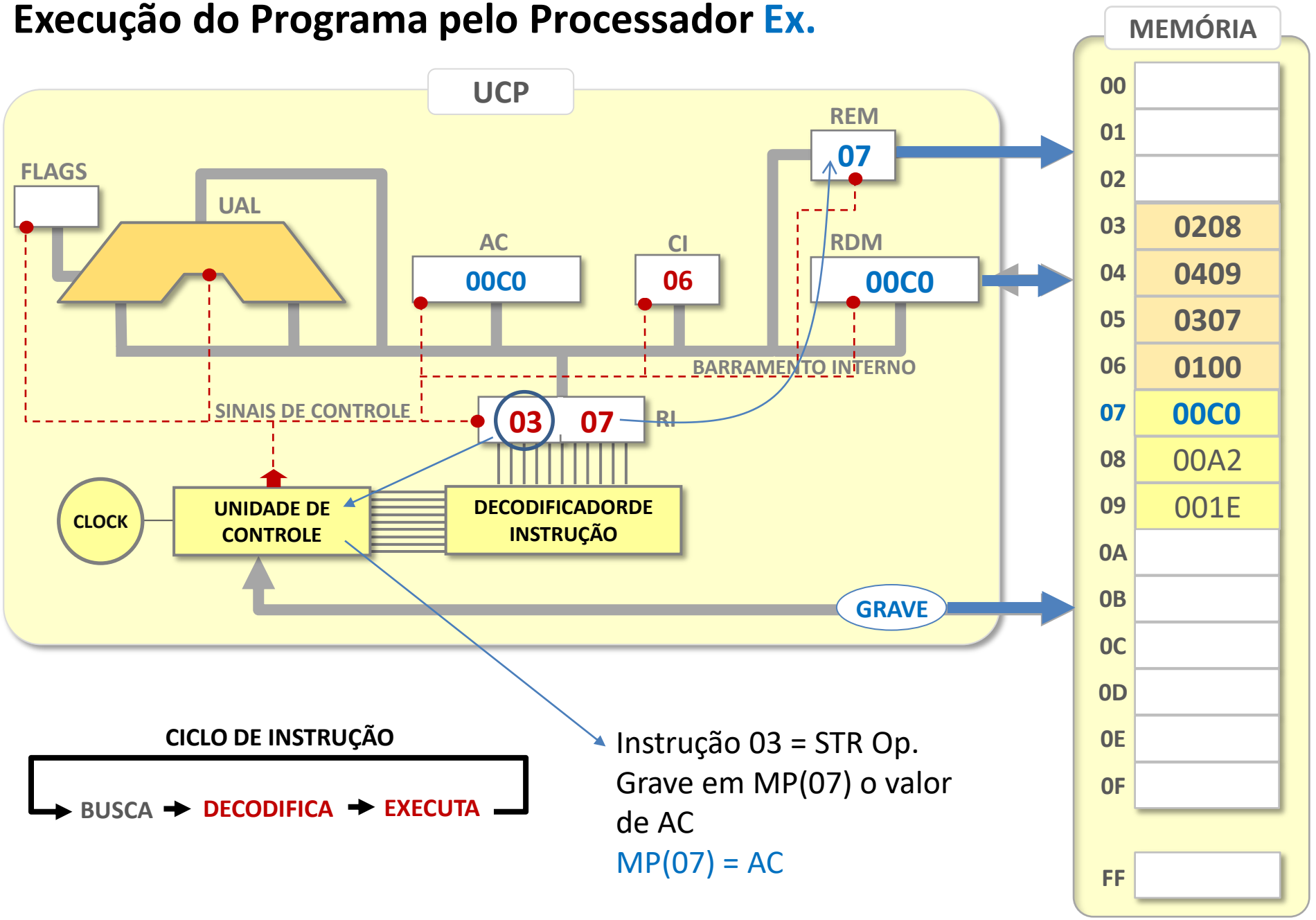
Execução do Programa pelo Processador Ex.



CICLO DE INSTRUÇÃO

BUSCA → DECODIFICA → EXECUTA

Execução do Programa pelo Processador Ex.

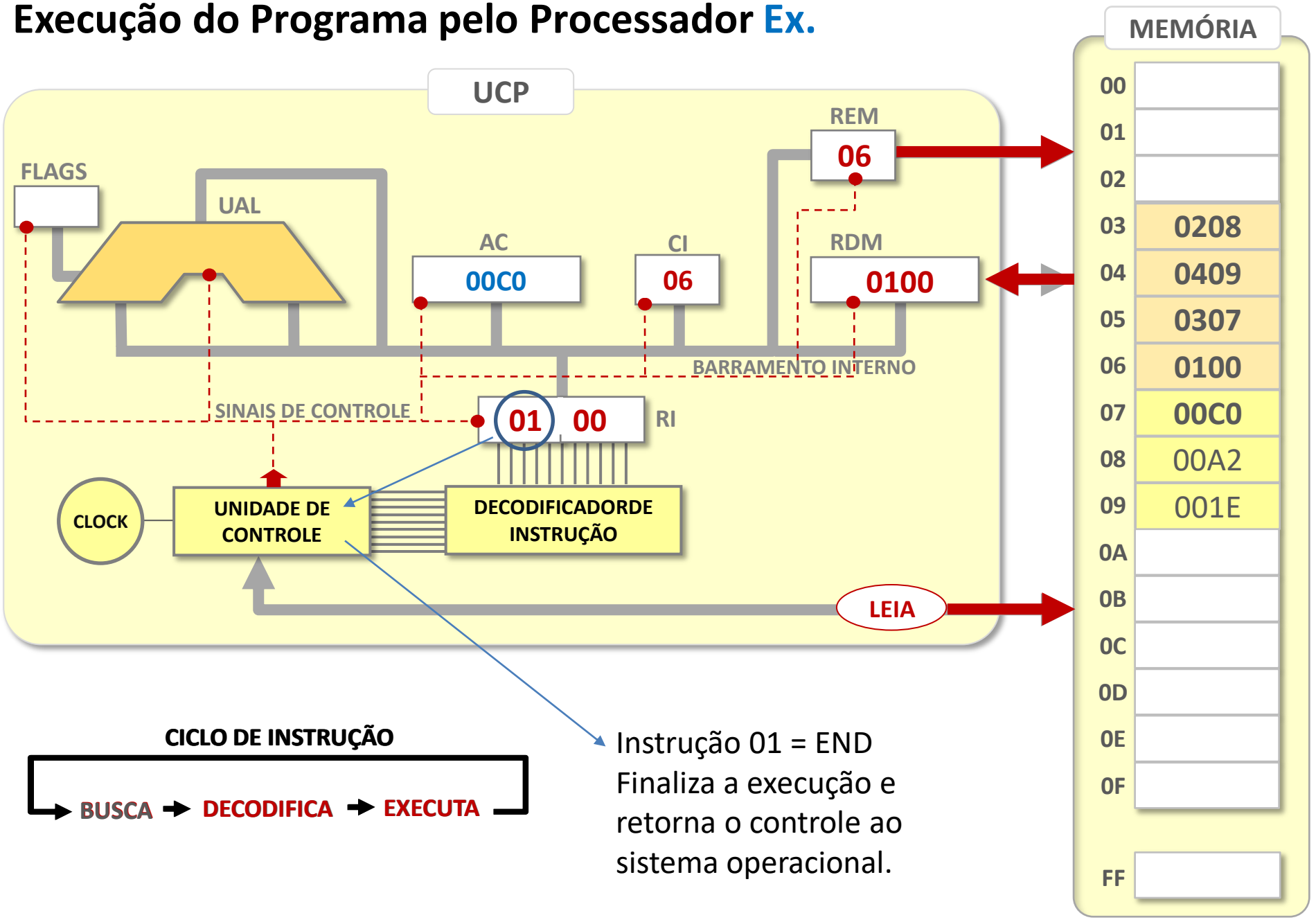


CICLO DE INSTRUÇÃO

BUSCA → DECODIFICA → EXECUTA

Instrução 03 = STR Op.
Grave em MP(07) o valor de AC
MP(07) = AC

Execução do Programa pelo Processador Ex.

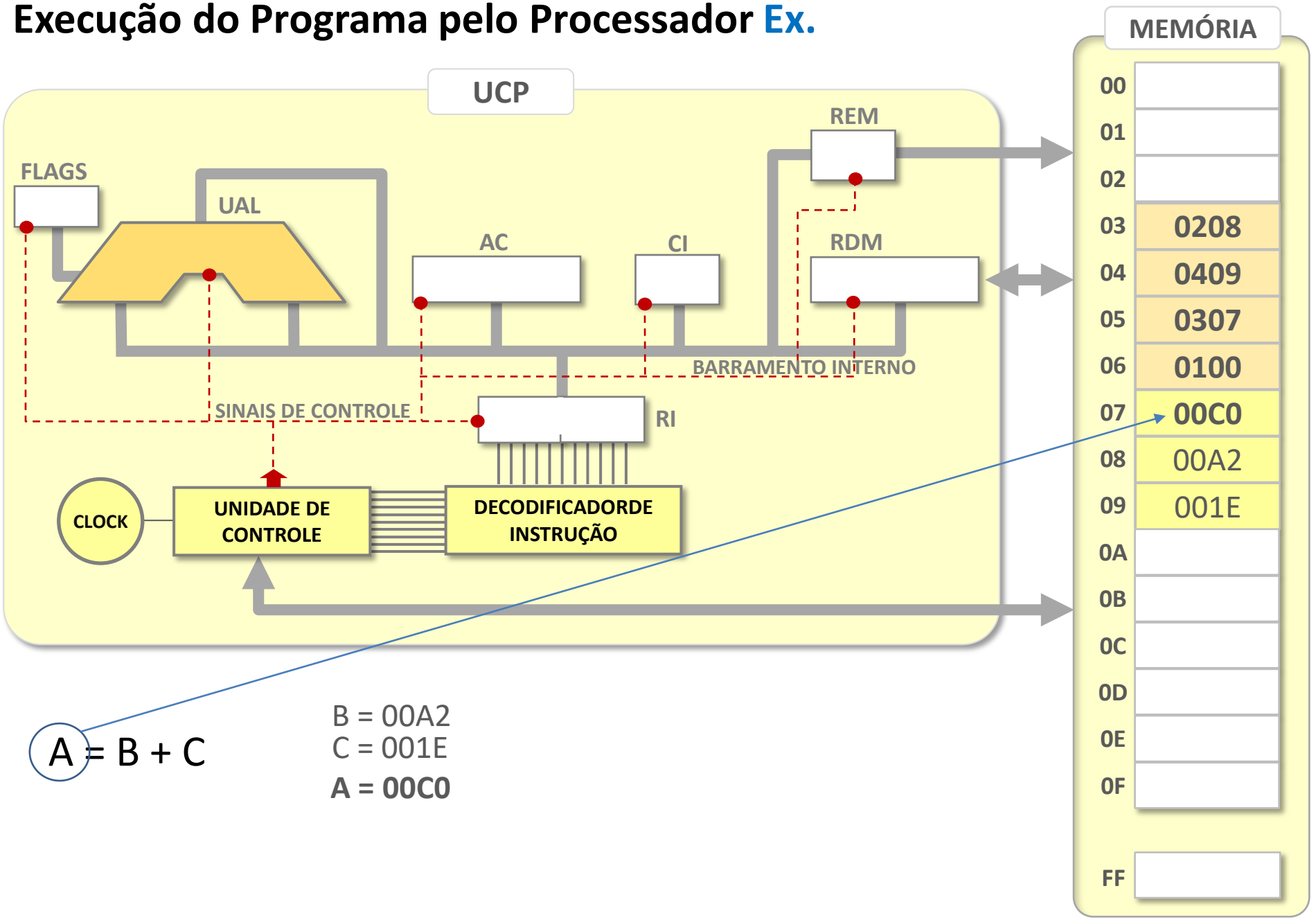


CICLO DE INSTRUÇÃO

BUSCA → DECODIFICA → EXECUTA

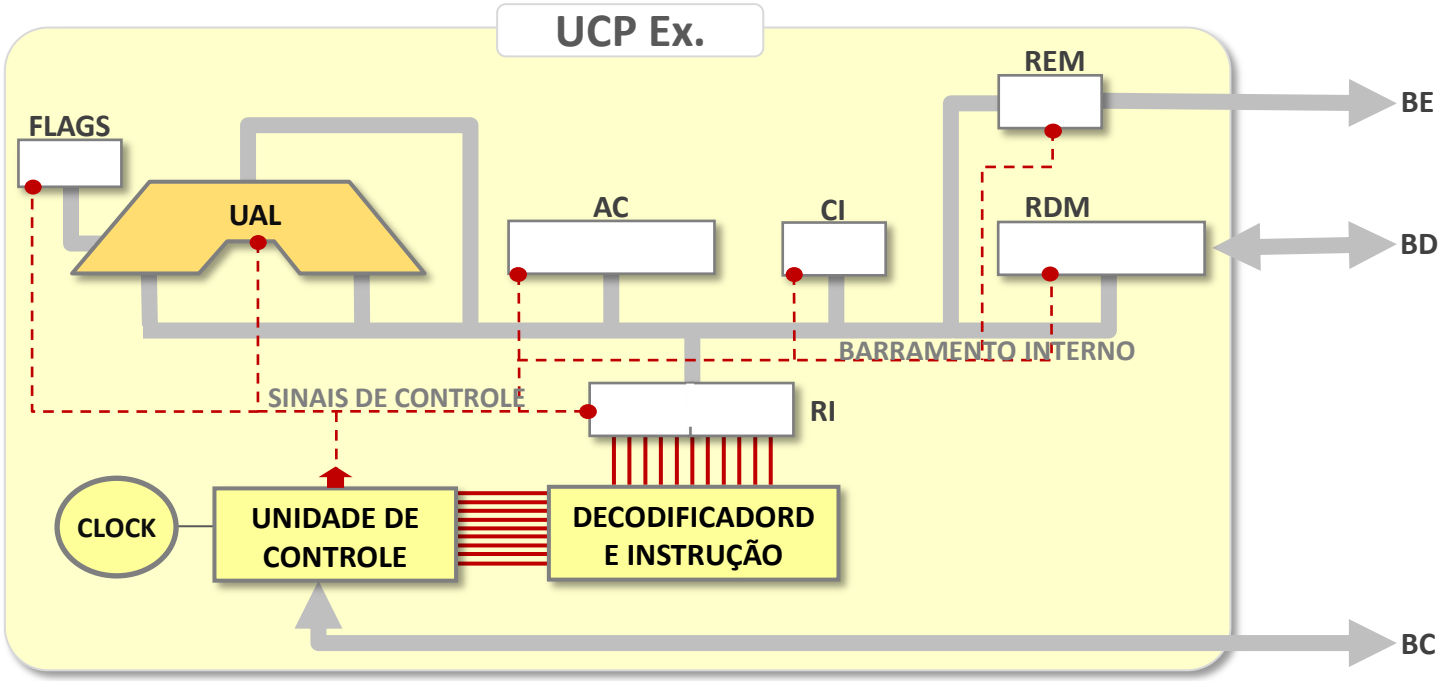
Instrução 01 = END
Finaliza a execução e retorna o controle ao sistema operacional.

Execução do Programa pelo Processador Ex.



A = B + C

B = 00A2
C = 001E
A = 00C0



END	Finaliza programa
LDA Op	AC = MP(Op)
STR Op	MP(Op) = AC
ADD Op	AC = AC + MP(Op)
SUB Op	AC = AC - MP(Op)
JZ Op	Se AC = 0 desvie para Op
JP Op	Se AC > 0 desvie para Op
JN Op	Se AC < 0 desvie para Op
JMP Op	Desvie para Op
GET Op	MP(Op) = (entrada)
PRT Op	(saída) = MP(Op)

ESPECIFICAÇÃO TÉCNICA:

- Tamanho de Palavra = 16 bits;
- Registros de endereço = 8 bits (256 células de memória);
- Tamanho da célula de memória = 16 bits;
- Operações somente com inteiros;
- Formato das instruções: um código de operação e um operando.

C.Op = 8 bits **Op**[1] = = 8 bits

UAL – Unidade Aritmética e Lógica
UC – Unidade de Controle
AC – Acumulador (Registro de Dados)
CI – Contador de Instrução;
RI – Registrador de Instrução;
REM – Registro de Endereço de Memória;
RDM – Registro de Dados de Memória;

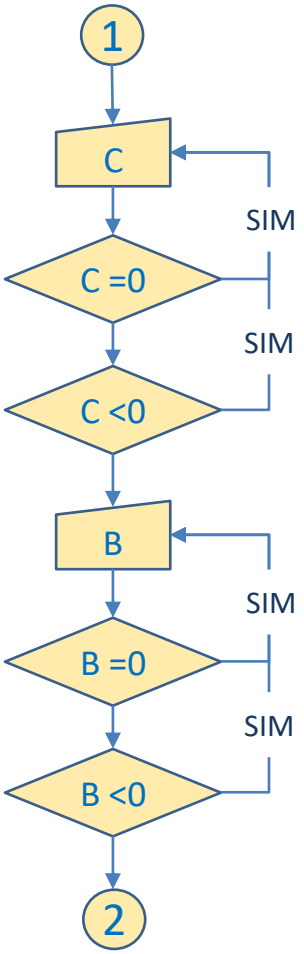
Um programa de multiplicação para o Ex.

O PROBLEMA:

A = C * B

A LÓGICA: (processador Ex.)

1. Entre com um valor para C maior que zero;
2. Entre com um valor para B maior que zero;



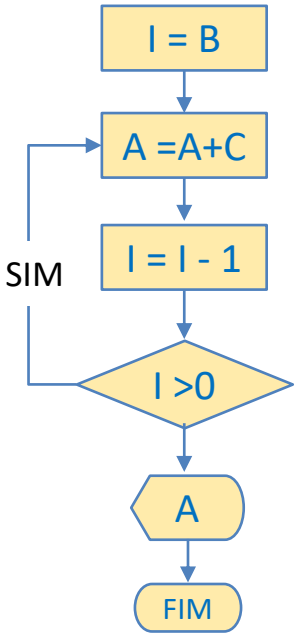

```
linha0: GET C
        LDA C
        JZ linha0
        JN linha0

linha4: GET B
        LDA B
        JZ linha4
        JN linha4
```

INSTRUÇÕES DO Ex.

END	Finaliza programa
LDA Op	AC = MP(Op)
STR Op	MP(Op) = AC
ADD Op	AC = AC + MP(Op)
SUB Op	AC = AC - MP(Op)
JZ Op	Se AC = 0 desvie para Op
JP Op	Se AC > 0 desvie para Op
JN Op	Se AC < 0 desvie para Op
JMP Op	Desvie para Op
GET Op	MP(Op) = (entrada)
PRT Op	(saída) = MP(Op)

A = C * B

Multiplicação através de somas sucessivas

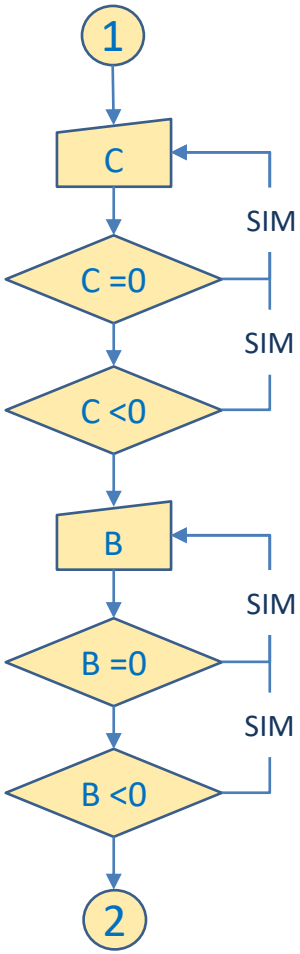



```
i = B;
while(i>0){
    A= A + C;
    i = i -1;
}
printf("%d",A);
```

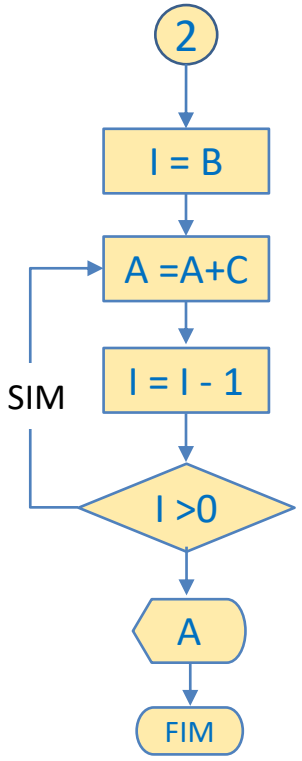
Um programa de multiplicação para o Ex.

O PROBLEMA:

A = C * B




```
linha0: GET C
        LDA C
        JZ linha0
        JN linha0

linha4: GET B
        LDA B
        JZ linha4
        JN linha4
```




```
linha8: LDA B
        STR I
linha10: LDA A
        ADD C
        STR A
        LDA I
        SUB Um
        STR I
        JP linha10
        PRT A
        END
A: 0
B: 0
C: 0
I: 0
Um: 1
```

Exercício

Execute a expressão A = B + C - D em linguagem assemble

Programa em português

1. Leia o valor da variável B para o registrador **AC**
2. Adicione ao registrador **AC** o valor da variável C
3. Subtraia o valor da variável D de **AC**
4. Grave o valor de **AC** na variável A

Programa em assemble

```
LDA B
ADD C
SUB D
STR A
```

Exercício

Escreva um programa em linguagem assemble para resolver o código abaixo:

A = B + C

D = A – F

Programa em português

1. Leia o valor da variável B para o registrador **AC**
2. Adicione ao registrador **AC** o valor da variável C
3. Grave o valor de **AC** na variável A
4. Subtraia o valor da variável F de **AC**
5. Grave o valor de **AC** na variável D

Programa em assemble

```
LDA B
ADD C
STR A
SUB F
STR D
```

Como funcione o Processador

Exercício:

Escreva um programa em linguagem C que atenda os seguintes requisitos:

1. Entre com um valor maior que zero para a variável B;
2. Entre com um valor maior que 100 para a variável C;
3. Subtraia B de C e guarde o resultado na variável A;
4. Exiba no vídeo o valor da variável A.

Resolução:

```
int A=0,B=0,C=0;
while (B<=0){
 scanf(“%d”,&B);
}
while (C<=100){
 scanf(“%d”,&C);
}
A = C – B;
printf(“%d”,A);
```

Programa C

```
int A=0,B=0,C=0;
while (B<=0){
 scanf("%d",&B);
}
while (C<=100){
 scanf("%d",&C);
}
A = C – B;
printf("%d",A);
```

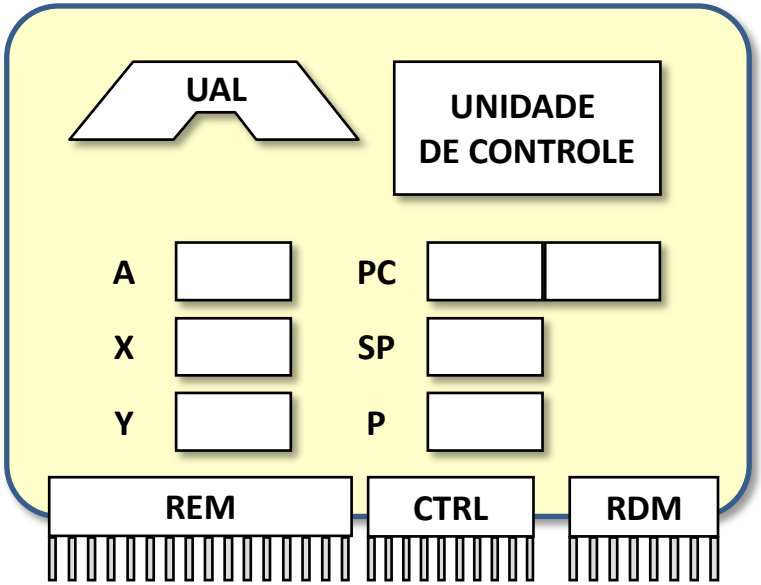
Programa Assemble

```
varB:   GET B
        LDA B
        JZ  varB
        JN varB
varC:   GET C
        LDA C
        SUB D
        JZ  varC
        JN varC
        LDA C
        SUB B
        STR A
        PRT A
        END
A:       0
B:       0
C:       0
D:       100
```

Processador

CPU – Motorola(6502) - Apple

Registradores de 8 bits
Barramento de endereço de 16 bits
Barramento de Dados de 8 bits



Registradores:

A : Acumulador - Armazena um valor a ser operado na unidade aritmética e lógica

X,Y : Armazena o índice a ser somado ao endereço nas instruções com endereçamento indexado

PC : Program Counter - Registrador de 16 bits - Armazena o Endereço da próxima instrução do programa a ser executada.

SP : Stack Pointer - Armazena o endereço do topo da pilha

P : Status do processador - Flags - Armazena informações sobre o estado da ultima operação efetuada. Cada bit deste registrador é utilizado separadamente.

Registrador P - Flags

N	V		B	D	I	Z	C

C : Carry (vai um)
Z : Resultado é Zero ou Não
I : Interrupção
D : Indica o modo Decimal selecionado
B : Break
V : OverFlow
N : Negative - Indica o Sinal

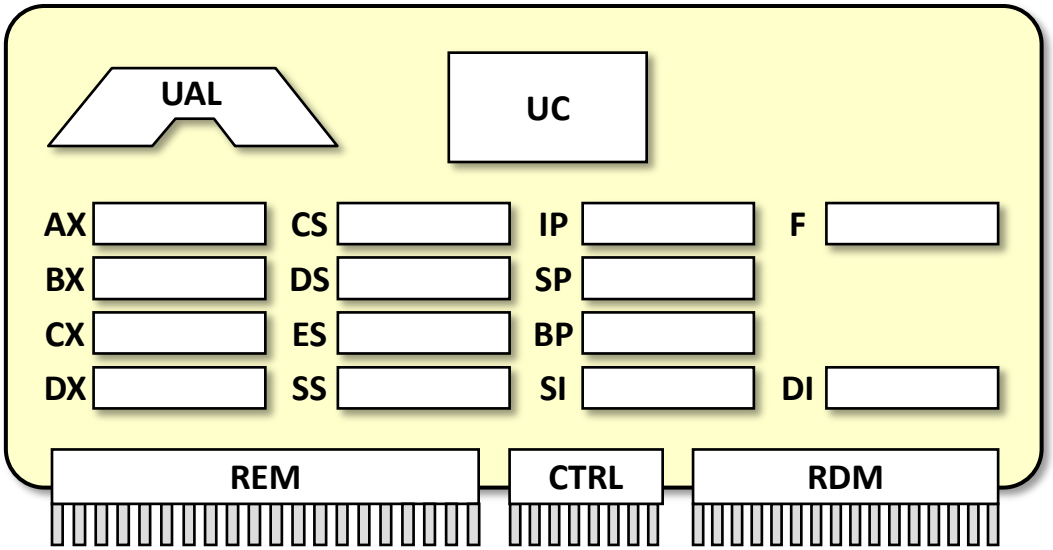
Processador

CPU – Intel (8086) – PC XT

Registradores de 16 bits

Barramento de Endereço de 20 bits

Barramento de Dados de 16 bits



Registradores de Uso Geral

AX : Acumulador — AH | AL

BX : Base — BH | BL

CX : Contador — CH | CL

DX : Dados — DH | DL

Registradores de Segmento

CS : Segmento de Código

DS : Segmento de Dados

ES : Segmento Extra

SS : Segmento da Pilha

Registrador dos Flags

F : Flags

Registradores de Deslocamento

DI : Índice de Destino

IP : Ponteiro de Instrução

BP : Ponteiro Base

SI : Índice de Origem

SP : Ponteiro da Pilha

REFERENCIAS BIBLIOGRAFICAS:

[1] TANEMBAUM, A. S**. Organização Estruturada de Computadores,**Livros Técnicos e Científicos, 2000. 460p.
[2] MONTEIRO, M.A. **Introdução à Organização de Computadores**, 5a ed. Livros Técnicos e Científico Editora SA, 2007. 695p