MC404

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

2006

Prof. Paulo Cesar Centoducatte

ducatte@ic.unicamp.br

www.ic.unicamp.br/~ducatte

MC404

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

"Programa do Curso e Conceitos Básicos"

Programa do Curso e Conceitos Básicos
Sumário

- Programa
- Conceitos Básicos
  - Bits e Bytes
  - Litle Endian e Big Endian
  - Memória
  - Representação de números com e sem sinal
  - Conversão entre bases numéricas

Programa

- Conceitos Básicos
  - Representação de números e caracteres
  - Organização de um computador digital
  - A CPU e a execução de um programa
  - Linguagem de máquina e linguagem de montagem
  - Introdução aos montadores

- Arquitetura do Microprocessador 8086
  - A família INTEL 80x86
  - Organização do Microprocessador 8086/8088
  - Organização de um PC

- Introdução à linguagem de montagem do 8086
  - Sintaxe do assembly 8086
  - Formato de Dados, variáveis e constantes
  - A estrutura do programa
  - Instruções de entrada e saída
  - Criando e rodando um programa

Programa

- O registrador de sinalização – FLAGS
  - Flags de Status e de Controle
  - Overflow
  - Como as instruções afetam os flags

- Instruções de controle de fluxo
  - Salto incondicional
  - Instrução de comparação
  - Salto condicional
  - Estruturas de linguagens de alto nível

- Instruções lógicas e de deslocamentos
  - Instruções lógicas
  - Instruções de deslocamentos
  - Instruções de rotação
  - Entrada/Saída de números binários e hexadecimais

Programa

- A pilha e procedimentos
  - Organização da pilha
  - Procedimentos
  - Chamadas e retorno de procedimentos

- Instruções de Multiplicação e Divisão
  - Instruções de multiplicação
  - Instruções de divisão
  - Extensão do sinal do dividendo
  - Entrada e saída de números decimais

- Arrays e modos de endereçamento
  - Arrays unidimensionais
  - Modos de endereçamento
  - Arrays bidimensionais
  - A instrução XLAT

Conceitos Básicos

Conceitos Básicos

BITs e BYTEs

- Bit = BInary digiT = vale sempre 0 ou 1. Elemento básico de informação
- Byte = 8 bits processados em paralelo (ao mesmo tempo)
- Word = n bytes (depende do processador em questão)
- Double word = 2 words
- Nibble = 4 bits (útil para BCD)

- Posição dos bits:

Para 1 byte:

7	6	5	4	3	2	1	0
0	1	0	1	0	1	0	1

Para 1 word (de 16 bits):

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

high byte | low byte

Conceitos Básicos

Litle Endian X Big Endian

Words são armazenados em bytes consecutivos, em memórias de largura de 8 bits.

Exemplo:

1025_{10} = 00000000 00000000 00000100 00000001_2

Endereço	Representação Big-Endian (MOTOROLA)	Representação Little-Endian (INTEL)
00	00000000	00000001
01	00000000	00000100
02	00000100	00000000
03	00000001	00000000

Conceitos Básicos

Memória

- **Memória: local do computador (hardware) onde se armazenam temporária ou definitivamente dados (números, caracteres e instruções)**
- **Posição de memória ou endereço: localidade física da memória onde se encontra o dado.**
- **Organização da memória:**

Endereço	Conteúdo
...	...
4MB	10110101
...	...
1048576	01001010
...	...
1765	01001101
...	...
4	01010000
3	11111111
2	11101001
1	11011010
0	01100100

Conceitos Básicos

Representação binária de números não sinalizados

Qualquer número em qualquer base ➔ $N = \sum_{i=0}^{n-1} d_i \times base^i$

a) 1 byte

$00100111_2 = 0 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

$= 0 + 0 + 32 + 0 + 0 + 4 + 2 + 1 = 39_{10}$

$= 27_{16}$

b) 1 word

$0101011101101110_2 = 0 \times 2^{15} + 1 \times 2^{14} + \ldots + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

$= 22382_{10}$

$= 576E_{16}$ (mais fácil de representar!)

high byte $= 0101\ 0111b = 57_{16}$

low byte $= 0110\ 1110b = 6E_{16}$

Conceitos Básicos

Conversão entre bases numéricas

Tipo de conversão	Procedimento
Decimal ➔ Binário	Divisões sucessivas por 2 até se obter zero no quociente. Leitura dos dígitos binários de baixo para cima.
Binário ➔ Decimal	Soma de potências de 2 cujo expoente é a posição do bit e cujo coeficiente é o próprio bit.
Hexadecimal ➔ Binário	Expandir cada dígito hexa em quatro dígitos binários segundo seu valor.
Binário ➔ Hexadecimal	Compactar cada quatro dígitos binários em um único dígito hexa segundo seu valor.
Decimal ➔ Hexadecimal	Divisões sucessivas por 16 até se obter zero no quociente; leitura dos dígitos de baixo para cima.
Hexadecimal ➔ Decimal	Soma de potências de 16 cujo expoente é a posição do dígito e cujo coeficiente é o valor do próprio dígito hexa.

Conceitos Básicos

Representação binária de números sinalizados

- **Representação com sinal e magnitude**
  - **O bit mais significativo é o sinal do número ➔ se for 1 o número é negativo se for 0 o número é positivo**

Exemplo 1: 01110001_2

valor não sinalizado = $0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 +$
$+ 0 \times 2^1 + 1 \times 2^0 =$
$= 64 + 32 + 16 + 1 = 113_{10}$

valor sinalizado **bit de sinal = 0 => " + " (positivo)**
$= 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 +$
$= 0 \times 2^1 + 1 \times 2^0 =$
$= 64 + 32 + 16 + 1 = 113_{10}$ ➔logo= $+113_{10}$

Conceitos Básicos

Exemplo 2: 10110001_2

valor não sinalizado **$= 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 +$**
$+ 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 =$
$= 128 + 32 + 16 + 1 = 177_{10}$

valor sinalizado **bit de sinal = 1 => " - " (negativo)**
$= 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 +$
$+ 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
$= 32 + 16 + 1 = 49_{10}$ ➔ logo = $- 49_{10}$

Conceitos Básicos

Exemplo 3:

$70FF_{16}$ = 0111000011111111_2

valor não sinalizado = $0 \times 2^{15} + 1 \times 2^{14} + \ldots + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

valor sinalizado ➔ bit de sinal = 0 => " + " (positivo)

$= + (0 \times 2^{15} + 1 \times 2^{14} + \ldots + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0)$

Exemplo 4:

$C777_{16}$ = 1100011101110111_2

valor não sinalizado = $1 \times 2^{15} + 1 \times 2^{14} + \ldots + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

valor sinalizado ➔ bit de sinal = 1 => " - " (negativo)

$= - (1 \times 2^{14} + \ldots + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0)$

Conceitos Básicos

Representações possíveis de números sinalizados

- **Complemento de 1**

  $-X = (2^n - 1) - X$ **n é o número de bits utilizados na representação**

- **Complemento de 2**

  $-X = 2^n - X$ **n é o número de bits utilizados representação**

Conceitos Básicos

- **Representações possíveis de números sinalizados**

Sinal e Magnitude	Complemento de 1	Complemento de 2
000 = +0	000 = +0	000 = +0
001 = +1	001 = +1	001 = +1
010 = +2	010 = +2	010 = +2
011 = +3	011 = +3	011 = +3
100 = -0	100 = -3	100 = -4
101 = -1	101 = -2	101 = -3
110 = -2	110 = -1	110 = -2
111 = -3	111 = -0	111 = -1

- **Representação em Complemento de 2 ➔ utilizada pois temos apenas uma representação para o zero e podemos fazer a soma e subtração com apenas um circuito.**

Conceitos Básicos

- **Números sinalizados de 32 bits, em Complemento de 2:**

$0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 0_{10}$
$0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0001_2 = +\ 1_{10}$
$0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0010_2 = +\ 2_{10}$
...
$0111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1110_2 = +\ 2{,}147{,}483{,}646_{10}$ — *maxint*
$0111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111_2 = +\ 2{,}147{,}483{,}647_{10}$
$1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000_2 = -\ 2{,}147{,}483{,}648_{10}$ — *minint*
$1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0001_2 = -\ 2{,}147{,}483{,}647_{10}$
$1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0010_2 = -\ 2{,}147{,}483{,}646_{10}$
...
$1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1101_2 = -\ 3_{10}$
$1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1110_2 = -\ 2_{10}$
$1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111_2 = -\ 1_{10}$

Conceitos Básicos

Representação em Complemento de 2 de um número:

- **Partindo-se da representação do negativo do valor a ser achado, nega-se este número (negar ➔ inverter) e somar 1**

Exemplo 1:

-5 em Complemento de 2 (com 1 bit de "sinal" e 4 para a magnitude)

Partindo-se da representação do $5_{10} = 00101_2$ ➔(invertendo os bits) = 11010 ➔ (somando 1) = 11011_2 = - 5 em Complemento de 2

Exemplo 2:

+5 em Complemento de 2 (com 1 bit de "sinal" e 4 para a magnitude)

Partindo-se da representação do $-5_{10} = 11011_2$ ➔(invertendo os bits) = 00100_2 ➔ (somando 1) = 00101_2 = +5 em Complemento de 2

Conceitos Básicos

- **Conversão de números com n bits em números com mais que n bits:**
  - **copiar o bit mais significativo (bit de sinal) nos outros bits (extensão do sinal):**

Exemplo:

0010 → 0000 0010

1010 → 1111 1010

Conceitos Básicos

Operações de soma e adição binárias

- **Como aprenderam no primeiro grau: (vai-um/vem-um)**

```
  0111 (7)      0111 (7)      0110 (6)
+ 0110 (6)    - 0110 (6)    - 0101 (5)
  1101 (13)     0001 (1)      0001 (1)
```

- **Subtração em complemento de 2 é feito como se fosse uma soma (A – B = A + (-B)):**
  - **subtração usando adição de números negativos**

```
   0111  (=+7)
 + 1010  (=−6)
1|0001  (=1)
```

Conceitos Básicos

Overflow

- **Overflow (resultado maior (menor) que a palavra do computador pode representar):**

Exemplo:

- **Quando na operação abaixo ocorre e quando não ocorre overflow ???**

```
  0111 (7) ou (+7)
+ 0001 (1) ou (+1)
  1000
```

Conceitos Básicos

Detecção de Overflow

- Não existe overflow quando adicionamos um número positivo e um negativo
- Não existe overflow quando os sinais dos números são os mesmos na subtração
- Ocorre overflow quando os valores afetam o sinal:
  - Somando dois números positivos dá um número negativo
  - Somando dois números negativos dá um número positivo
  - Subtrai um número negativo de um positivo e dá negativo
  - Subtrai um número positivo de um negativo e dá positivo

Exercício

- Considere as operaçôes A + B e A – B
  - Pode ocorrer overflow se B = 0 ?
  - Pode ocorrer overflow se A = 0 ?

Conceitos Básicos

Multiplicação Binária

- **Exemplo:**

1010 X 101

```
      1 0 1 0
    X 1 0 1
    ---------
      1 0 1 0
    0 0 0 0
  1 0 1 0
---------------
1 1 0 0 1 0
```

Conceitos Básicos

Divisão Binária

- **Exemplo:**

1 1 0 0 1 0 / 101

```
  1 1 0 0 1 0   | 1 0 1
- 1 0 1         -------
  -----           1 0 1 0
  0 0 1 0 1
-     1 0 1
  ---------
      0 0 0 0
```

Conceitos Básicos

Representação de Caracteres Alfanuméricos

- **Tabela ASCII (American Standard Code Interchange Information)**

Exemplo:

64	@		96	´
65	**A**		97	**a**
66	**B**		98	**b**
67	**C**		99	**c**
68	**D**		100	**d**
69	**E**		101	**e**
70	**F**		102	**f**
71	**G**		103	**g**
72	**H**		104	**h**
73	**I**		105	**i**

48	**0**
49	**1**
50	**2**
51	**3**
52	**4**
53	**5**
54	**6**
55	**7**
56	**8**
57	**9**

Conceitos Básicos

Linguagem de programação

- Linguagem de Alto Nível – próximo ao ser humano, escrita de forma textual.
  - Ex: if (a==b) a=b+c;
- Linguagem de Montagem (Assembly) – próximo à linguagem de máquina, escrita em códigos (mnemônicos)
  - Ex: ADD AX,BX;
- Linguagem de Máquina – linguagem que o computador consegue executar – códigos binários
  - Ex: 01010001

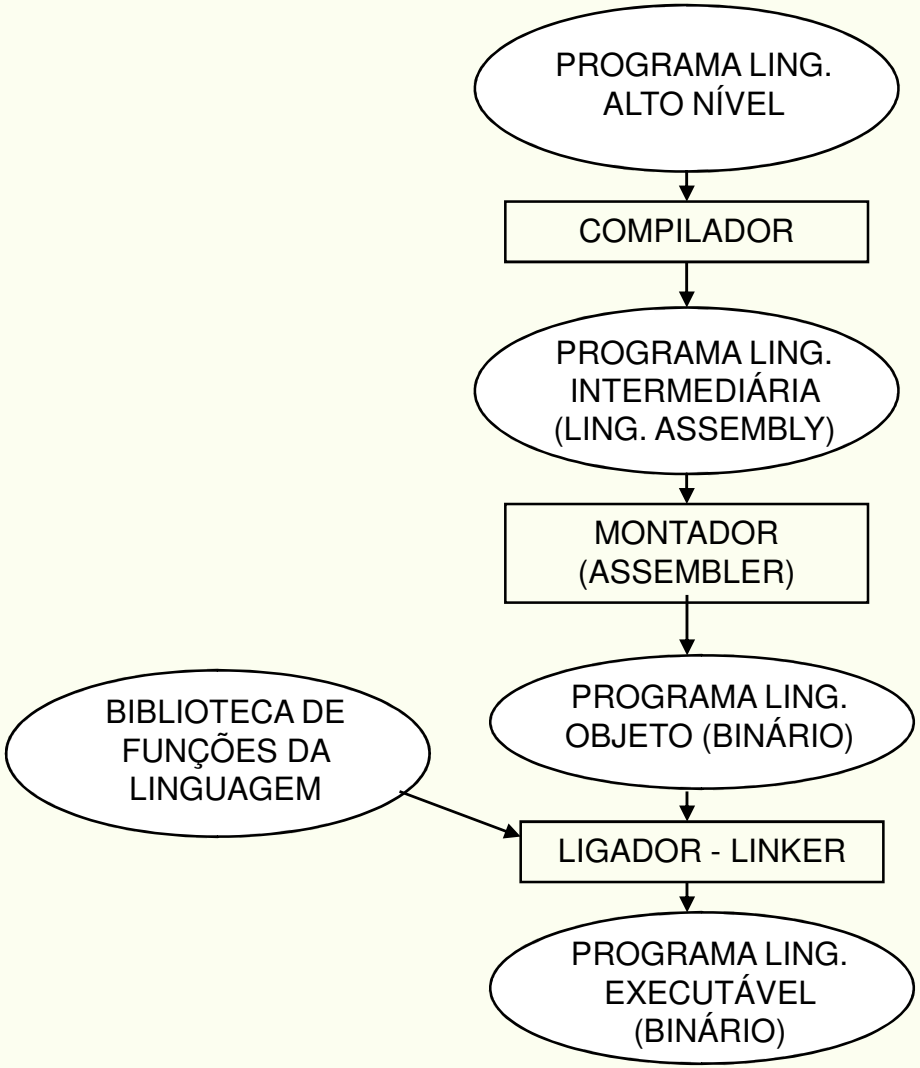
Conceitos Básicos

Execução de um programa

- **Um programa escrito em linguagem de alto nível, para ser executado ele deve:**
  - **Ser traduzido para linguagem de máquina (compiladores, montadores, ligadores);**
  - **Ter seus endereços realocados, conforme posição onde será carregado na memória (loaders);**
  - **Ser alocado em um região da memória (loaders).**

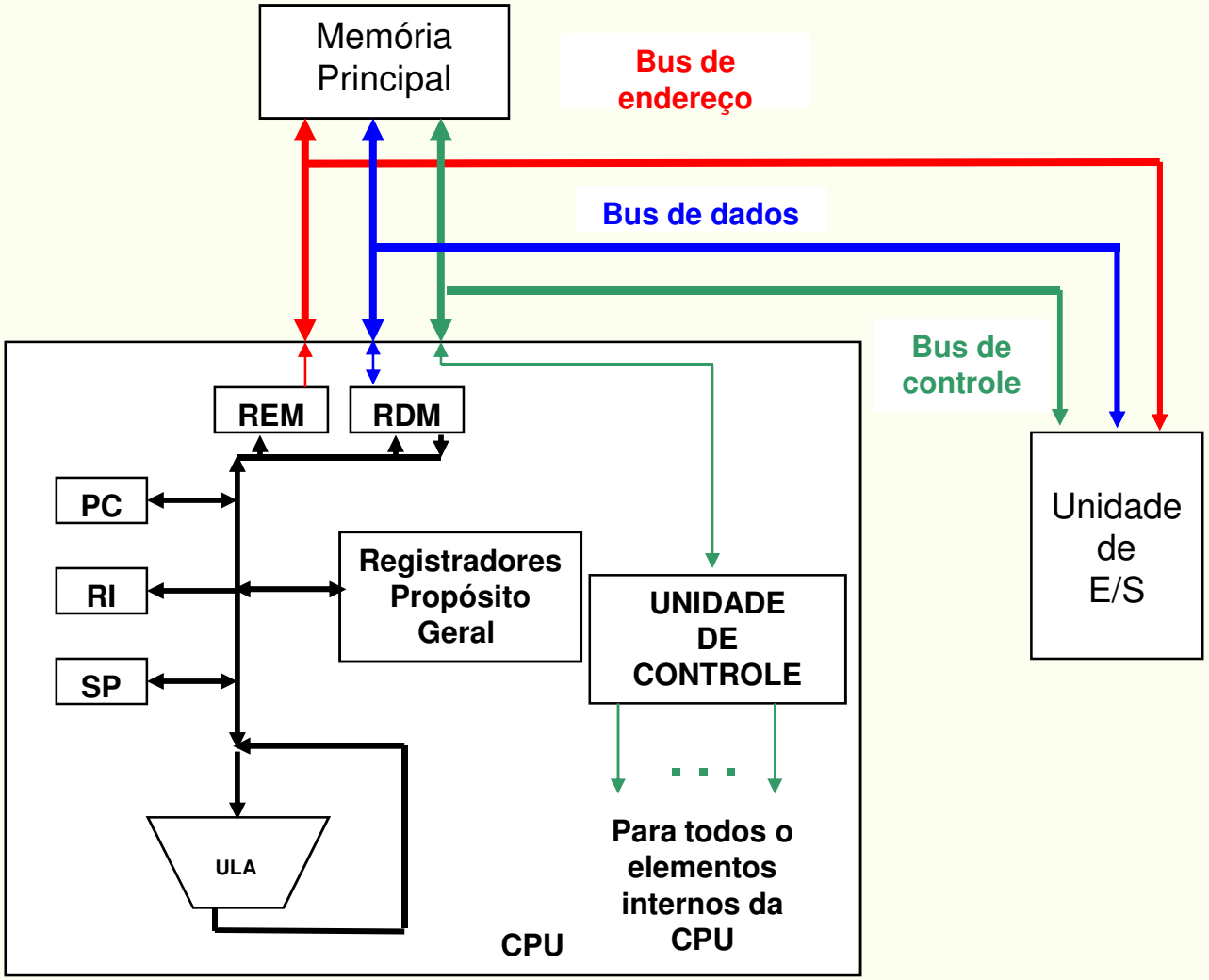
Conceitos Básicos

Processo de tradução de um programa em linguagem de alto nível



Conceitos Básicos

Organização Básica de um Computador Digital



Conceitos Básicos

Organização Básica de um Computador Digital

- **Unidade Central de Processamento – CPU:**
  - **Unidade de Controle – UC;**
  - **Unidade Lógica e Aritmética – ULA;**
  - **Registradores de Propósito Geral – GPR;**
  - **Registradores Específicos.**

- **Unidade de Memória → hierarquia de memória:**
  - **Memória Principal;**
  - **Memória Secundária;**

- **Unidade de Entrada e Saída:**
  - **Interfaces;**
  - **Canais de E/S;**
  - **Processadores E/S.**

- **Barramentos:**
  - **Barramento de Endereços;**
  - **Barramento de Dados;**
  - **Barramento de Controle.**

Conceitos Básicos

Organização Básica de um Computador Digital

- **Unidade Central de Processamento – CPU**
  - **Responsável por todo o processamento (execução de programas) no sistema**
    - **Unidade de Controle: circuito que gera os sinais de controle responsáveis pelo gerenciamento (controle) de todas as atividades do computador.**
    - **Unidade Lógica e Aritmética – ULA: circuito responsável por efetuar todas as operações lógicas e aritméticas.**
    - **Registradores de Propósito Geral – GPR: elementos de memória (circuitos) responsáveis por armazenar os dados que são utilizados durante a execução de um programa (instruções).**

Conceitos Básicos

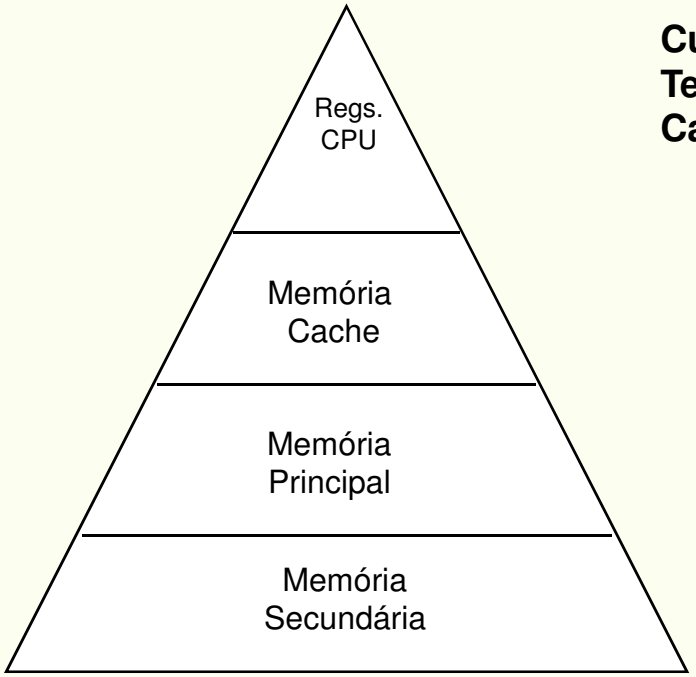
Organização Básica de um Computador Digital

- **Unidade Central de Processamento – CPU (cont.)**
  - **Registradores Específicos:**
    - **Program Counter – PC: armazena o endereço da próxima instrução a ser executada;**
    - **Stack Pointer – SP: armazena o endereço do topo da pilha;**
    - **Registrador de Instrução – RI: armazena a instrução que está sendo executada;**
    - **Registrador de Dados de Memória – RDM: armazena os dados que vem da memória (lidos) ou que vão para a memória (escritos);**
    - **Registrador de Endereços de memória – REM: armazena o endereço enviado para a memória, quando ocorrer um acesso à mesma (leitura ou escrita)**

Conceitos Básicos

Organização Básica de um Computador Digital

- **Unidade de Memória**
  - **Hierarquia de Memória: sistema de memória com objetivo de melhorar o desempenho de um sistema computacional, diminuindo o tempo de acesso médio**



Custo de armazenamento por bit maior
Tempo de acesso menor
Capacidade de armazenamento menor

Custo de armazenamento por bit menor
Tempo de acesso menor
Capacidade de armazenamento maior

Conceitos Básicos

Organização Básica de um Computador Digital

- **Memórias**
  - **Semicondutoras: fabricadas com materiais semicondutores (silício) – circuitos integrados.**
    - **RAM – Random Access Memory : memória de acesso aleatório, volátil.**
      - **SRAM – RAM estática: seu conteúdo só se altera quando se escreve nela ou quando se desliga a tensão de alimentação. Exemplo – registradores da CPU, memória cache.**
      - **DRAM – RAM dinâmica: periodicamente é necessário reescrever o seu conteúdo (refresh de memória) pois há diminuição de cargas elétricas.**

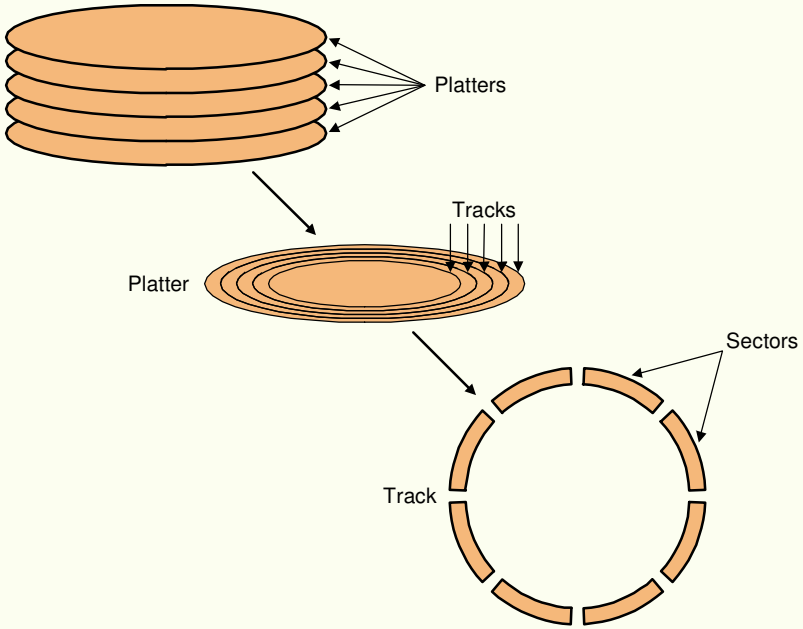
        Exemplo – memória principal.
    - **ROM – Read Only Memory: memória somente de leitura, não volátil.**
      - **ROM: gravação feita pelo fabricante da memória, não apagável;**
      - **PROM – Programmable ROM: programação feita pelo usuário, não apagável;**
      - **EPROM – Erasable PROM: programação feita pelo usuário, apagável através de luz ultra-violeta;**
      - **EEPROM – Electrical EPROM: programação feita pelo usuário, apagável eletricamente;**
    - **Flash – memória semicondutora, não volátil e de escrita e leitura, apagável.**

Conceitos Básicos

Organização Básica de um Computador Digital

- **Memórias (continuação)**
  - **Magnéticas**
    - **Discos – Hard Disk – HDs**
    - **Opticos – CD-ROM, DVD, etc.**
    - **Fitas – cartchos, rolos, etc.**

Exemplo: memórias secundárias



Disco Magnético ➔ pratos, lados, trilhas e setores

Conceitos Básicos

Organização Básica de um Computador Digital

- **Unidade de Entrada e Saída: responsável por gerenciar a ligação entre CPU-Memória-barramentos e os periféricos.**
  - **Interfaces – circuitos simples que apenas compatibilizam a comunicação (protocolo). O controle da transferência é feita pela CPU. Exemplo: interface serial RS232, interface paralela, interface USB;**
  - **Canais de E/S – circuitos que controlam e compatibilizam a comunicação. A CPU apenas inicia a transferência. Exemplo – Controlador de Acesso Direto à Memória (DMA – Direct Access Memory);**
  - **Processadores de E/S – são CPUs dedicadas a fazer E/S de dados. Iniciam e controlam a comunicação.**

Conceitos Básicos

Organização Básica de um Computador Digital

- **Barramentos: Conjunto de fios que fazem a ligação física entre as diversas unidades.**
  - **Barramento de Endereços: Por onde trafegam os endereços;**
  - **Barramento de Dados: Por onde trafegam os dados;**
  - **Barramento de Controle: por onde trafegam os sinais de controle;**

- **Observação:**

  Internamente à CPU, existe um barramento interno de dados que liga os registradoes com a ULA e a UC, e um barramento interno de controle que liga a UC a todos os elementos da CPU.

Conceitos Básicos

Organização Básica de um Computador Digital

- **Formato das Instruções**
  - **Tamanho (número de bits) e o significado de cada campo de bits de uma instrução de linguagem de máquina.**
- **Conjunto de Instruções**
  - **Cada processador tem o seu conjunto de instruções de linguagem de máquina (ISA – Instruction Set Architecture). Este conjunto contém todas as instruções, em linguagem de máquina, que o processador pode executar.**

Conceitos Básicos

Execução de uma instrução pela CPU

- **Ciclo de execução de uma instrução:**
- **Leitura da instrução da memória principal – Fetch da Instrução**

  REM ← PC
  Read (sinal de controle)
  PC ← PC atualizado
  RDM ← [REM] (instrução lida)

  – **Decodificação da instrução**

    RI ← RDM (instrução)
    É feita a decodificação pela Unidade de Controle

  – **Busca dos operandos da instrução na memória – se houver**

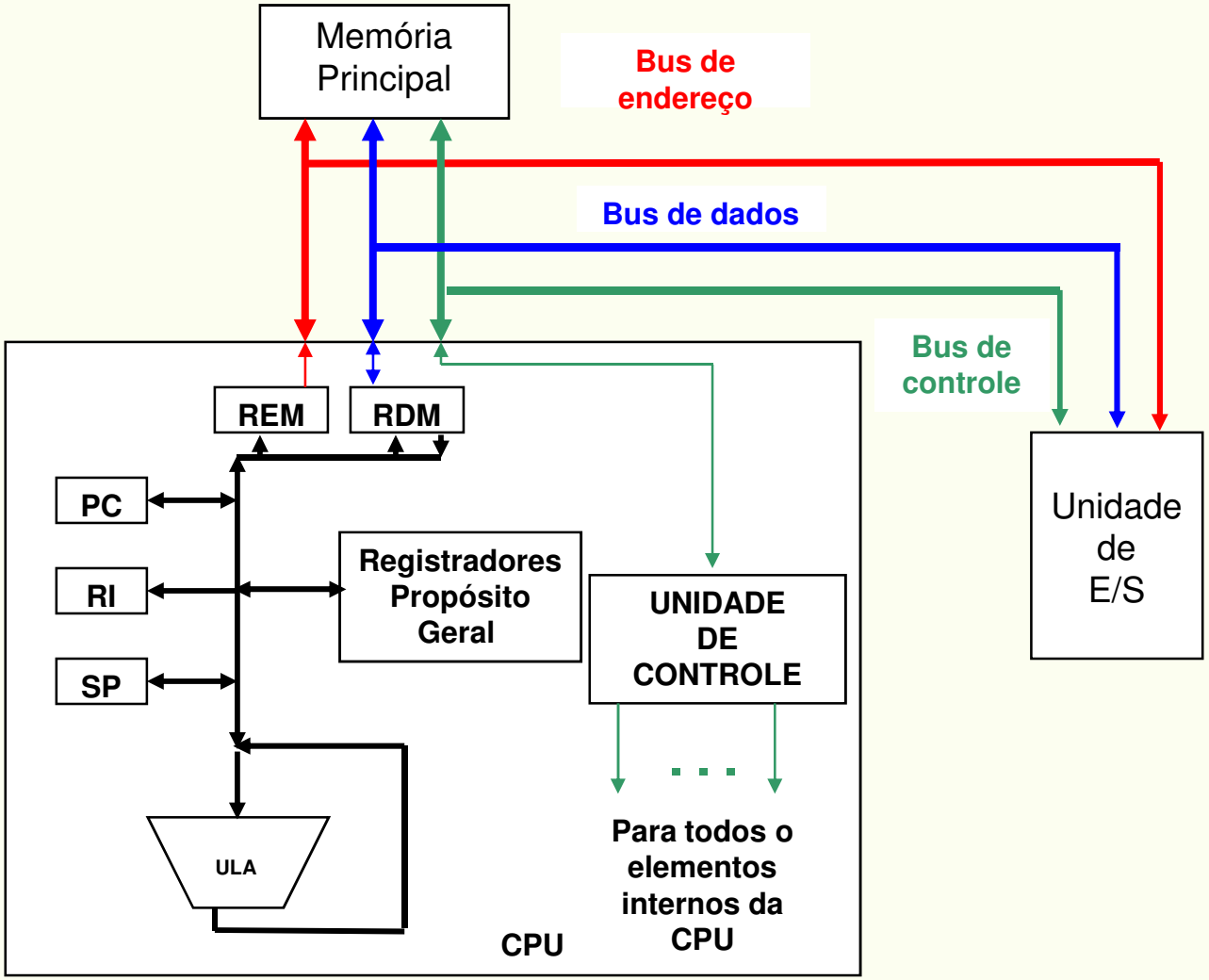
    REM ← PC
    Read (sinal de controle)
    PC ← PC atualizado
    RDM ← [REM] (operando lido)

  – **Execução da instrução – depende da instrução**

- **Obs – Quando usamos [..], significa que estamos acessando um conteúdo de memória, cujo endereço está dentro dos colchetes.**

Conceitos Básicos

Organização Básica de um Computador Digital

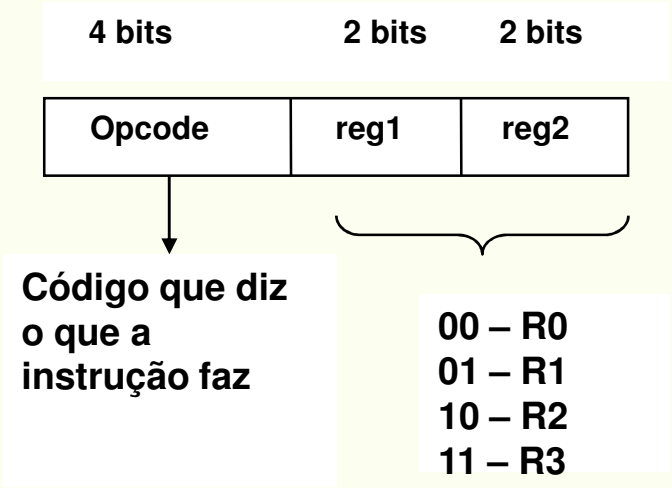


Conceitos Básicos

ESTUDO DE CASO - CPU HIPOTÉTICA

- **Formatos das instruções da CPU HIPOTÉTICA:**

  – **Formato tipo I – Uma palavra de 8 bits, com os seguintes campos:**



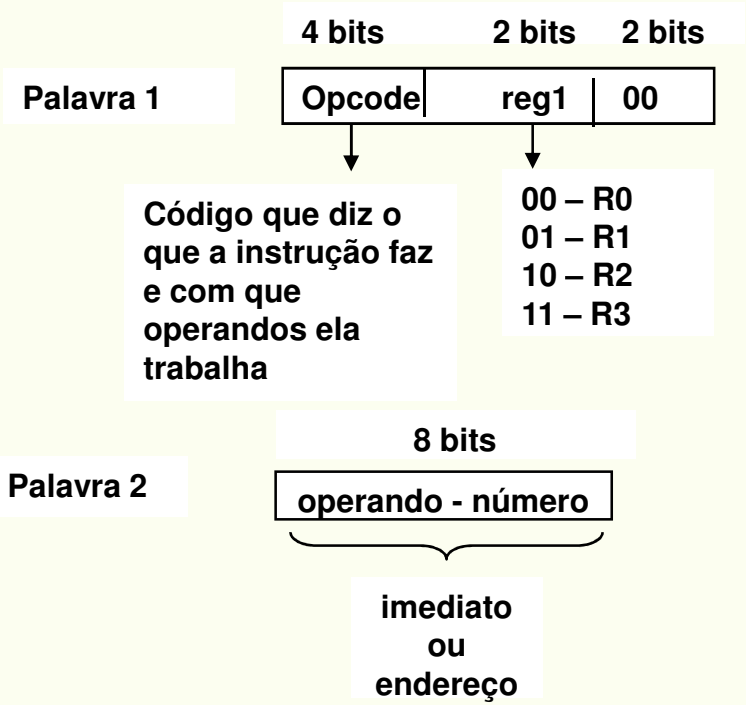
Exemplo: MOV R0,R1 ; R0 ← R1

Conceitos Básicos

ESTUDO DE CASO - CPU HIPOTÉTICA

- **Formatos das instruções da CPU HIPOTÉTICA:**

  – **Formato tipo II – Duas palavras de 8 bits, com os seguintes campos:**



Exemplos:

MOV R0, 5 ; R0 ← 5

MOV R0, [5] ; R0 ← [5]

Conceitos Básicos

Mnemônico	Operandos	Opcode	Significado
Instruções de Movimentação de Dados			
MOV	Reg1,Reg2	0000	Reg1 ←Reg2
MOV	Reg,imed	1000	Reg ← imed
MOV	Reg,[end]	1001	Reg ← [end]
MOV	[end],Reg	1010	[end] ← Reg
Instruções Aritméticas e Lógicas			
ADD	Reg1,Reg2	0001	Reg1 ← Reg1 + Reg2
ADD	Reg,imed	1011	Reg ← Reg + imed
SUB	Reg1,Reg2	0010	Reg1 ← Reg1 - Reg2
SUB	Reg,imed	1100	Reg ← Reg – imed
AND	Reg1,Reg2	0011	Reg1 ← Reg1 *e* Reg2
AND	Reg,imed	1101	Reg ← Reg *e* imed
OR	Reg1,Reg2	0100	Reg1 ← Reg1 *ou* Reg2
Instruções de Manipulação de Pilha			
PUSH	Reg	0101	SP-- , [SP] ← Reg
POP	Reg	0110	Reg←[SP], SP++
Instruções de Controle de Fluxo de Execução			
JMP	end	1110	PC ← end
CALL	end	1111	SP-- , [SP]←PC , PC←end
RET	---	0111	PC←[SP] , SP++