



UNIDADE II: Conceitos básico de um computador



Tópicos da aula

- Apresentar as idéias básicas do funcionamento de um computador através do projeto de um processador básico e dos recursos necessários à execução de aplicações elementares

Definições básicas: computador

Um computador é constituído por

Hardware

- Dispositivos eletrônicos e eletromecânicos que realizam uma ou mais das seguintes funções elementares
 - Armazenamento de dados
 - Transferência de dados (entrada e/ou saída)
 - Processamento de dados
 - Controle

Software

- Programas executados que instruem o hardware a executar uma série de ações a fim de cumprir uma determinada tarefa
 - Um programa é portanto constituído por uma seqüência de instruções de máquina
- Em uma visão mais elaborada, um software não é constituído apenas por programas, mas, também, por documentos de especificação e projeto, manuais e textos de ajuda, entre outros.

Definições básicas: computador

❑ Exemplos de hardware

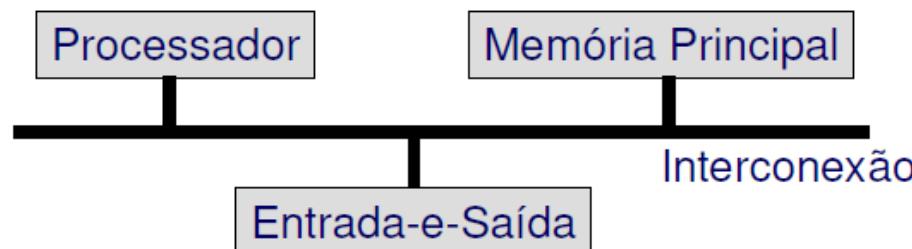
- ❑ Processador (ex. Pentium 4)
- ❑ Memória de chip (RAM, ROM)
- ❑ Driver de disco rígido (FDD)
- ❑ Driver de disco flexível (FDD)
- ❑ Driver de CD-ROM
- ❑ Monitor de vídeo
- ❑ Teclado
- ❑ Mouse
- ❑ Impressora
- ❑ Cabos
- ❑ Placa mãe
- ❑ Placas de expansão
 - ❑ Fax-modem, rede, vídeo, 3D,...

Definições básicas: computador

- ❑ Todos esse componentes podem ser reunidos em três clássicas básicas

- ❑ Processador
- ❑ Memória principal
- ❑ Entrada-e-saída

Os cabos e as placas implementam as interconexões entre os componentes do computador



Definições básicas: computador

❑ Exemplos de software

- ❑ Sistemas operacionais
 - ❑ Windows, Linux, DOS, Unix
- ❑ Programas utilitários (usados no desenvolvimento de outros programas)
 - ❑ Compiladores, montadores, ligadores, depuradores
- ❑ Programas aplicativos
 - ❑ Editores de texto
 - ❑ Planilhas eletrônicas
 - ❑ Jogos
 - ❑ Anti-vírus
 - ❑ Compactadores
 - ❑ Visualizadores de imagens
 - ❑ ...

Definições básicas: instrução de máquina

- ❑ Todo programa executado em um computador é armazenado em um arquivo executável
- ❑ Um arquivo executável é constituído por uma seqüência de instruções de máquina
- ❑ Uma instrução de máquina é formada por um conjunto limitado de bits (dígitos binários) que especificam, basicamente, a ação a ser realizada e os dados a serem manipulados
 - ❑ A ação é denominada OPERAÇÃO
 - ❑ Os dados são denominados OPERANDOS
- ❑ Uma instrução de máquina pode ter vários operandos, mas apenas uma operação
- ❑ Logo, uma instrução de máquina é formada por
 - ❑ OPERAÇÃO + OPERANDO(S)

Definições básicas: instrução de máquina

❑ Exemplo de instrução de máquina

0010101110110010

- ❑ Onde, por exemplo os quatro bits mais à esquerda (**0010**) representam a operação a ser realizada
- ❑ Os outros doze bits (**101110110010**) formam um operando que indica a localização (endereço) de um dado a ser manipulado.

Definições básicas: instrução de máquina

Exemplo de programa executável

```
0010101110110010
0011100000001001
0001010011100101
0110011100000110
0010100010010010
0000101010110100
0001010101001001
0111001010000001
0000110000000000
0010010010101001
0110100010101010
0011010010101001
```

Definições básicas: linguagens de programação

Linguagem de máquina

- Antigamente, os programadores escreviam programas construindo-os a partir de instruções de máquina

Linguagem de montagem (assembly)

- Posteriormente, criou-se uma abstração da linguagem de máquina chamada linguagem de montagem (assembly)
- Na linguagem de montagem, as instruções da linguagem de máquina são referenciadas por representações textuais das operações e dos operandos das instruções
 - Exemplo: **ADD A** ao invés de 0001001100010011
- A representação textual da operação na linguagem de montagem é denominada **mnemônico**

Definições básicas: linguagens de programação

❑ Linguagem de montagem (assembly)

- ❑ Exemplo de programa em uma linguagem de montagem

LOAD A

ADD B

SUB C

STO D

- ❑ Contudo, para ser executado, o programa na linguagem de montagem precisa ser traduzido para a linguagem de máquina e essa tradução é realizada por um programa denominado **montador** (ou **assembler**)

❑ IMPORTANTE

- ❑ A função do programa montador é traduzir um programa escrito na linguagem de montagem para a linguagem de máquina

Definições básicas: linguagens de programação

❑ Linguagens de alto nível

- ❑ Programar em uma linguagem de montagem tem limitações
 - ❑ É difícil e bastante suscetível a erros
 - ❑ Programas escritos nas linguagens de montagem de um processador não podem ser executados em um outro processador que tenha uma linguagem de máquina diferente
- ❑ A partir dessas dificuldades foram criadas linguagens de programação ainda mais abstratas e sobretudo independentes do processador
- ❑ Essas linguagens estão muito mais próximas da linguagem utilizada na comunicação entre pessoas do que da linguagem do computador e, por isso, são chamadas de **linguagens de alto nível**
- ❑ A linguagem de montagem é considerada uma linguagem de **baixo nível**

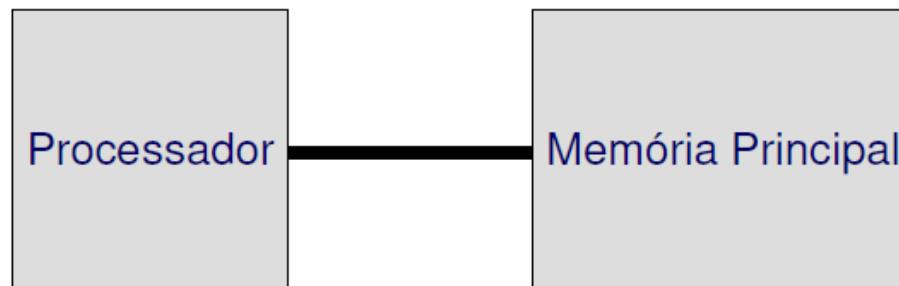
Definições básicas: linguagens de programação

❑ Linguagens de alto nível

- ❑ Exemplos de linguagens de alto nível
 - ❑ C
 - ❑ C++
 - ❑ C#
 - ❑ Pascal
 - ❑ Delphi
 - ❑ Java
 - ❑ Fortran
 - ❑ Cobol
 - ❑ Clipper
 - ❑ ADA
 - ❑ PROLOG
 - ❑ ...

Computador básico

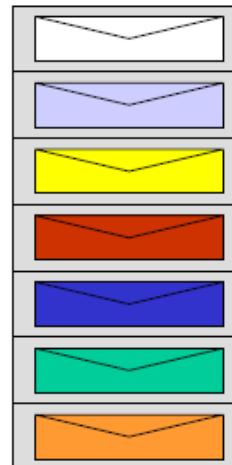
- ❑ **O computador básico é constituído pelo processador e por uma memória principal (RAM)**
 - ❑ O processador executa instruções do programa armazenado na memória principal, a qual também armazena os dados a serem processados pelo processador e os dados resultantes da execução das instruções



Computador básico: a memória principal

- ❑ A memória principal pode ser vista como um conjunto de caixas de correspondências em que cada caixa tem capacidade de armazenar uma correspondência e possui um identificador únicos

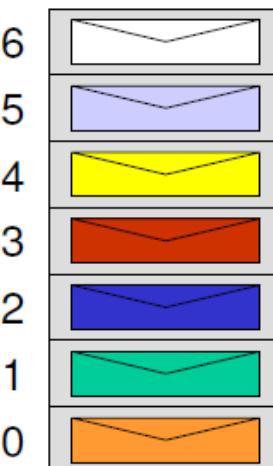
Caixa Postal 006
Caixa Postal 005
Caixa Postal 004
Caixa Postal 003
Caixa Postal 002
Caixa Postal 001
Caixa Postal 000





Computador básico: a memória principal

- ❑ Os identificadores das caixas postais podem ser denominados ENDEREÇOS (ADDRESSES ou ADDR)
- ❑ O número de dígitos usados para especificar o endereço depende do número máximo de caixas postais possíveis de serem implementadas
- ❑ Se existirem no máximo 10 caixas postais, um único dígito decimal é suficiente



Computador básico: a memória principal

- ❑ A memória principal de um computador possui muito mais do que dez posições (em geral de milhões a bilhões), mas existem sistemas computacionais de pequeno porte com menos de mil endereços em memória
- ❑ A memória principal é de fato organizada em elementos denominados células, e, assim como as caixas postais, cada célula possui um endereço (ou posição)
- ❑ Uma célula tem capacidade de armazenar uma quantidade limitada de bits (ex: 8 ou 16 bits)
- ❑ A capacidade da memória será dada pelo número de células vezes a capacidade de cada célula
 - ❑ Ex: 1024 células x 8 bits/célula = 8.192 bits

Computador básico: a memória principal

- Uma célula da memória pode armazenar diferentes tipos de informação (mas apenas uma de cada vez)
 - Dado
 - Endereço de outra célula
 - Instrução de programa na linguagem de máquina

7	D	Dados
6	C	
5	B	
4	A	
3	STO D	
2	SUB C	
1	ADD B	
0	LOAD A	

7	D	Instruções
6	C	
5	B	
4	A	
3	STO D	
2	SUB C	
1	ADD B	
0	LOAD A	

Computador básico: a memória principal

- ❑ Durante a execução de um programa, o processador precisa, constantemente, acessar a memória para carregar (*to load*):
 - ❑ as instruções a serem executadas e
 - ❑ os dados a serem processados
- ❑ Ele também precisa acessar a memória para armazenar (*to store*)
 - ❑ os resultados de algumas operações realizadas

Computador básico: a memória principal

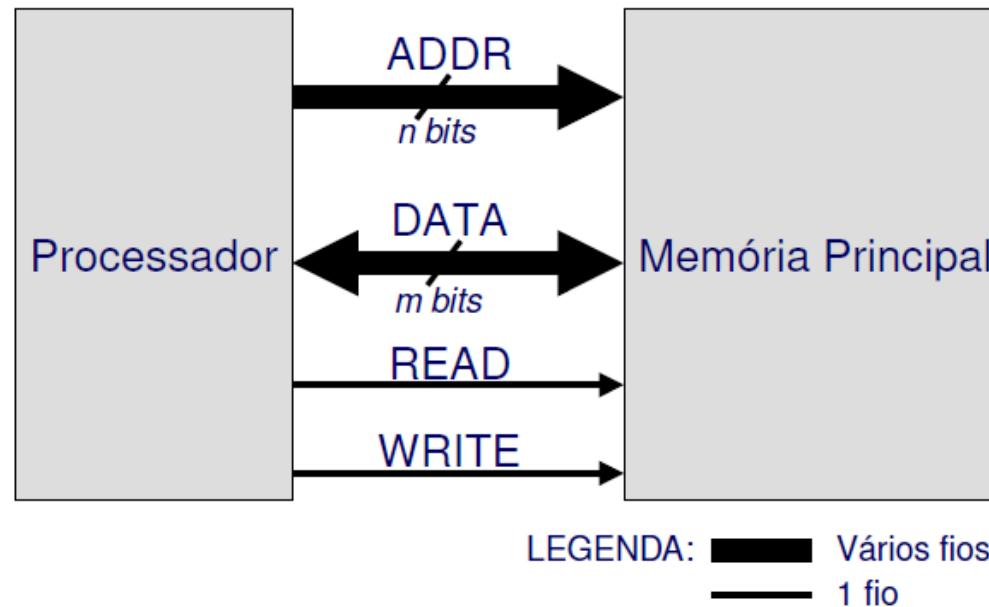
- ❑ **Logo, o processador precisa de meios para**
 - ❑ Selecionar uma célula através do seu endereço
 - ❑ Transferir uma informação (instrução ou dado)
 - ❑ Indicar o sentido da transferência (leitura ou escrita)
- ❑ **Isso é tipicamente implementado por três conjuntos de fios (ou vias)**
 - ❑ Endereço (ADDR)
 - ❑ Indica a célula a ser acessada
 - ❑ Dado (DATA)
 - ❑ Usado para transferir a informação entre da memória para o processador e vice-versa
 - ❑ É na verdade usado para transferir tanto dados quanto instruções
 - ❑ Controle
 - ❑ READ: quando igual a 1 comanda uma transferência da memória para o processador
 - ❑ WRITE: quando igual a 1 comanda uma transferência do processador para a memória

Computador básico: a memória principal

- ❑ As vias de endereço (ADDR) e de dado (DATA) terão tantos fios quantos forem necessários para selecionar todas as posições da memória e para transferir todos os bits de uma célula, respectivamente
- ❑ Quem aciona a via ADDR jogando um endereço em seus fios é processador (é unidirecional)
- ❑ A via DATA pode ser acionada pelo processador ou pela memória, conforme o sentido da transferência – leitura ou escrita (é bidirecional)
- ❑ Os fios de controle (READ e WRITE) são acionados exclusivamente pelo processador (são unidirecionais)

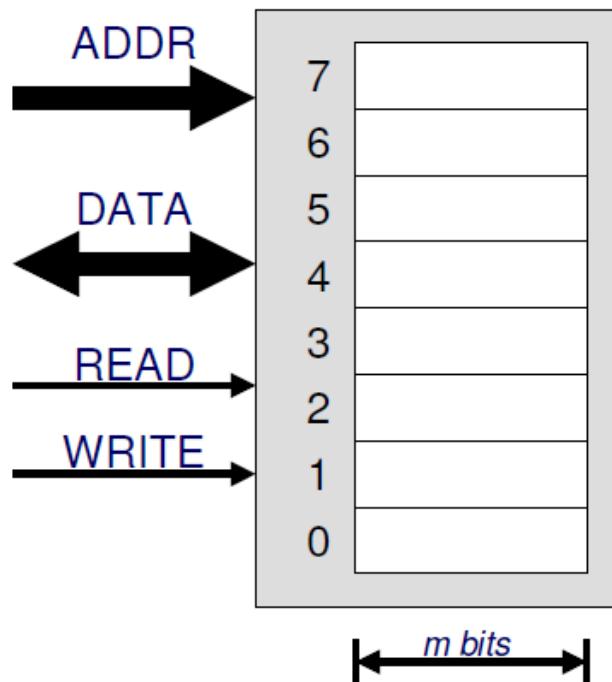
Computador básico: a memória principal

- Logo, a conexão entre o processador e a memória principal pode ser representada da seguinte forma



Computador básico: a memória principal

- A memória principal pode ser representada da seguinte forma



Computador básico: a memória principal

- ❑ O número de fios na via de dados é dado pela largura das células da memória (*m bits*).
- ❑ Exemplo

Largura da célula	Número de fios	Nome
8 bits	8 fios	DATA[7..0]
16 bits	16 fios	DATA[15..0]
32 bits	32 fios	DATA[31..0]

Computador básico: a memória principal

- ❑ O número de fios da via ADDR é dado da seguinte forma

$$\text{Largura_ADDR} = \log_2 (\text{Número_de_Células}) \quad (\text{bits})$$

- ❑ Exemplos:

Número de Células	Largura ADDR (bits)
2	1
4	2
32	5
1024	10
4294967296	32

Computador básico: a memória principal

□ De uma maneira mais simples, a largura da via ADDR é dada pelo número de bits necessário para representar o maior endereço da memória

□ Exemplos

□ Com 2 fios consegue-se representar 4 combinações de 2 bits que resultam em 4 endereços diferentes

$$00_{(2)} = 0_{(10)}$$

$$01_{(2)} = 1_{(10)}$$

$$10_{(2)} = 2_{(10)}$$

$$11_{(2)} = 3_{(10)} \quad \text{onde } 3 \text{ é o maior endereço representável}$$



Estes índices indicam o sistema de numeração utilizado para representar o número

Computador básico: a memória principal

□ Outros exemplos

- Com 4 fios consegue-se representar 16 combinações de 4 bits:

$$0000_{(2)} = 0_{(10)}$$

... ...

$$1111_{(2)} = 15_{(10)} \quad \text{o maior endereço representável}$$

- O número máximo de endereços representáveis é dado por

$$\text{Número_de_endereços} = 2^{\text{Número_de_fios}}$$

- O maior endereço é definido por

$$\text{Maior_endereço} = 2^{\text{Número_de_fios}} - 1$$

Computador básico: a memória principal

❑ Exemplo final

- ❑ Uma memória com 1024 células requer uma via de endereços que possa selecionar desde a célula de endereço 0 até a célula 1023
- ❑ Para determinar a largura da via de endereços deve-se fazer $\log_2(1024)$ ou determinar qual potência de 2 resulta em um valor maior ou igual a 1024

❑ Potências de 2

$$2^1 = 2 \quad 2^9 = 512$$

$$2^2 = 4 \quad 2^{10} = 1024$$

$$2^3 = 8 \quad 2^{11} = 2048$$

$$2^4 = 16 \quad 2^{12} = 4096$$

$$2^5 = 32 \quad 2^{13} = 8192$$

$$2^6 = 64$$

$$2^7 = 128$$

$$2^8 = 256$$

Neste caso a via de endereço será denominada ADDR[9..0] e terá 10 fios, referenciados por

ADDR[9], ADDR[8], ADDR[7], ADDR[6],
ADDR[5], ADDR[4], ADDR[3], ADDR[2],
ADDR[1], ADDR[0]



Computador básico: a memória principal

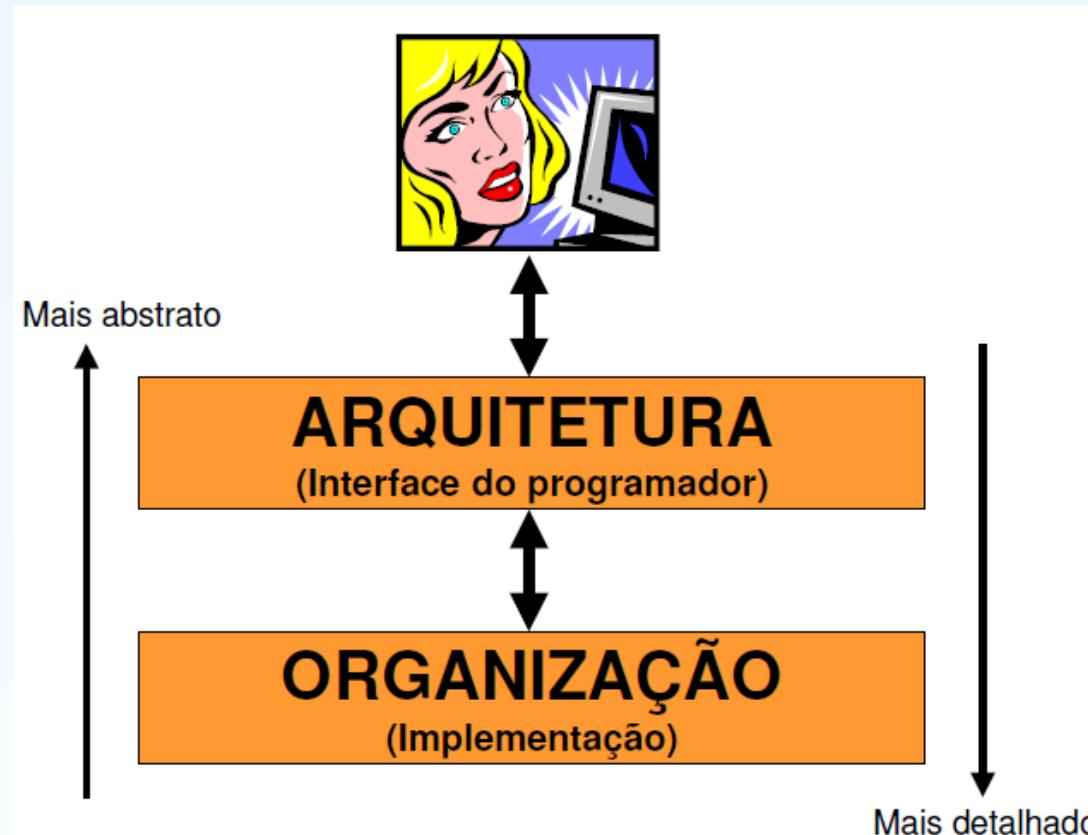
□ Exercícios

- Calcular quanto endereços podem ser representados com 5 fios (bits)
- Calcular quantos endereços podem ser representados com 7 fios
- Calcular quantos fios são necessário para construir uma via ADDR para acessar uma memória com 80 células
- Calcular quantos fios são necessário para construir uma via ADDR para acessar uma memória com 256 células
- Calcular quantos fios são necessário para construir uma via ADDR para acessar uma memória com 500 células

Computador básico: processador

- ❑ O processador pode ser escrito em diferentes níveis de abstração (com menos ou mais de detalhes)
- ❑ O primeiro nível, mais abstrato e com menos detalhes, é o nível arquitetural, que nada mais é do que a interface do programador
- ❑ O segundo nível, menos abstrato e com mais detalhes, é o nível organizacional, que constitui-se na implementação da arquitetura

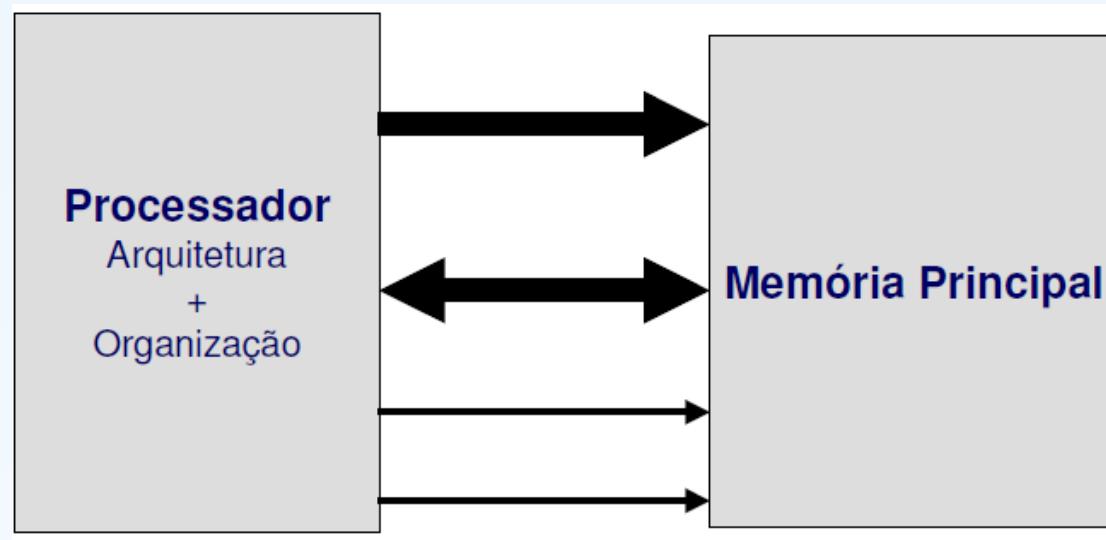
Computador básico: processador



Computador básico: processador

- ❑ A arquitetura refere-se a atributos que são visíveis ao programador do processador (programação em linguagem de montagem)
- ❑ A organização refere-se a atributos que não são visíveis ao programador, sendo foco da atenção do engenheiro de computação (projetista de hardware)

Computador básico: processador



Computador básico: arquitetura de um processador

❑ Atributos arquiteturais

- ❑ Tamanho da palavra de dados
 - ❑ Número de bits do dado manipulado pelo processador
- ❑ Tipos de dados
 - ❑ Tipos de dados manipulados pelo processador: inteiro (com ou sem sinal), real,...
- ❑ Tamanho da palavra de instrução
 - ❑ Número de bits usados para representar uma instrução de programa
- ❑ Formato das instruções
 - ❑ Estrutura utilizada para organização das instruções
 - ❑ Largura (em bits) do campo do código da operação (OpCode)
 - ❑ Número de operandos
 - ❑ Largura dos operandos (em bits)

Computador básico: arquitetura de um processador

❑ Atributos arquiteturais

- ❑ Modos de endereçamento
 - ❑ Métodos de acesso aos dados processados pelas instruções
- ❑ Registradores
 - ❑ Unidades de armazenamento internas da CPU
 - ❑ Registradores de uso específico
 - ❑ Registradores de uso geral
- ❑ Conjunto de instruções
 - ❑ Vocabulário de instruções
 - ❑ Códigos das operações das instruções

Computador básico: arquitetura de um processador

□ Problema

□ Projetar a arquitetura de um processador programável que permita manipular dados inteiros positivos e negativos (-32768 a +32767) e implementar qualquer equação realize soma e/ou subtração de variáveis e constantes.

□ Escrever programas que implementem as seguintes equações

$$D = A + B - C$$

$$F = D + 2$$



Computador básico: arquitetura de um processador

□ Solução:

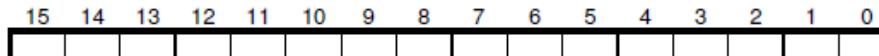
- Processador BIP (*Basic Instruction-set Processor*)
- Arquitetura de 16 bits
- Instruções básicas para a implementação de equações baseadas em operações aritméticas de soma e subtração com número inteiros

- Paulo V. Vieira, Cesar A. Zeferino, André L. A. Raabe: “*Avaliação Empírica da Proposta Interdisciplinar de Uso dos Processadores BIP*”. II Congresso Brasileiro de Informática na Educação (CBIE 2013).

Computador básico: arquitetura de um processador

☐ Arquitetura do BIP

- ☐ Tamanho da palavra de dados
 - ☐ 16 bits



☐ Tipos de dados

- ☐ Inteiro sem sinal

- ☐ Variando de 0

a

65535

0000000000000000

1111111111111111

- ☐ Inteiro com sinal

- ☐ Variando de -32768

a

32767

1000000000000000

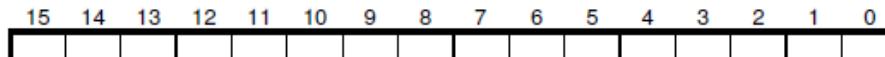
0111111111111111

- ☐ Números negativos representados em complemento 2

Computador básico: arquitetura de um processador

❑ Arquitetura do BIP (cont.)

- ❑ Tamanho da palavra de instrução
 - ❑ 16 bits



❑ Formato das instruções

- ❑ 4 bits para o código da operação
- ❑ 1 operando de 12 bits
- ❑ Dois campos
 - ❑ OpCode (4 bits): código da operação
 - ❑ addr/imed (12 bits): endereço (addr) de uma variável na memória ou constante (imed)





Computador básico: arquitetura de um processador

❑ Arquitetura do BIP (cont.)

- ❑ Modos de endereçamento
 - ❑ Todas as instruções envolvem uma operação entre um registrador do processador, denominado acumulador (ACC), e um operando indicado na instrução (variável ou constante)
 - ❑ São utilizados dois modos de endereçamento
 - ❑ **Direto**
 - ❑ o dado a ser processado é indicado pelo operando (endereço) que aponta para uma variável na memória
 - ❑ **Imediato**
 - ❑ o dado a ser manipulado é o próprio operando (imediato) da instrução



Computador básico: arquitetura de um processador

❑ Arquitetura do BIP (cont.)

- ❑ Registradores
 - ❑ Do ponto de vista arquitetural (parte visível ao programador) o BIP possuirá um único registrador importante neste momento denominado acumulador (ACC), o qual pode ser visto como uma variável temporária interna dentro do processador
 - ❑ Ele terá também outros registradores (PC, IR e STATUS), a serem estudados mais tarde

Computador básico: arquitetura de um processador

❑ Arquitetura do BIP (cont.)

- ❑ Conjunto de instruções
 - ❑ Soma
 - ❑ ADD addr soma do acumulador com uma variável (*addr)
 - ❑ ADDI imed soma do acumulador com uma constante (imed)
 - ❑ Subtração
 - ❑ SUB addr subtração do acumulador com uma variável (*addr)
 - ❑ SUBI imed subtração do acumulador com uma constante (imed)
 - ❑ Carga no acumulador (load)
 - ❑ LD addr carga de uma variável (*addr) no acumulador
 - ❑ LDI imed carga de uma constante (imed) no acumulador
 - ❑ Armazenamento na memória (store)
 - ❑ STO addr cópia do acumulador para uma variável (*addr)

*addr indica o conteúdo da célula apontada pelo endereço (addr)

Computador básico: arquitetura de um processador

❑ Arquitetura do BIP (cont.)

❑ Conjunto de instruções

Instruções baseadas no modo de endereçamento direto

ADD end	$ACC \leftarrow ACC + (*end)$	O acumulador (ACC) recebe a soma do seu conteúdo com o de uma variável em memória indicada pelo endereço (end)
SUB end	$ACC \leftarrow ACC - (*end)$	O acumulador (ACC) recebe a subtração do seu conteúdo com o de uma variável em memória indicada pelo endereço (end)
LD end	$ACC \leftarrow (*end)$	O acumulador (ACC) recebe o conteúdo de uma variável em memória indicada pelo endereço (end)
STO end	$*end \leftarrow ACC$	A variável em memória indicada pelo endereço (end) recebe o conteúdo do acumulador (ACC).

Instruções baseadas no modo de endereçamento imediato

ADDI imed	$ACC \leftarrow ACC + imed$	O acumulador (ACC) recebe a soma do seu conteúdo com o valor de uma constante embutida na instrução (imed)
SUBI imed	$ACC \leftarrow ACC - imed$	O acumulador (ACC) recebe a subtração do seu conteúdo com o valor de uma constante embutida na instrução (imed)
LDI imed	$ACC \leftarrow imed$	O acumulador (ACC) recebe o valor de uma constante embutida na instrução (imed)

Computador básico: arquitetura de um processador

□ Arquitetura do BIP (cont.)

- Conjunto de instruções
 - Códigos da instruções (em binário)

□ NOP	0000 ₍₂₎	= 0	(10)
□ STO	0001 ₍₂₎	= 1	(10)
□ LD	0010 ₍₂₎	= 2	(10)
□ LDI	0011 ₍₂₎	= 3	(10)
□ ADD	0100 ₍₂₎	= 4	(10)
□ ADDI	0101 ₍₂₎	= 5	(10)
□ SUB	0110 ₍₂₎	= 6	(10)
□ SUBI	0111 ₍₂₎	= 7	(10)

A instrução NOP faz com que o processador não realize nenhuma operação, servindo para inserir pausas na execução do programa.

Computador básico: arquitetura de um processador

Implementando o trecho abaixo no BIP

$$D = A + B - C$$

$$F = D + 2$$

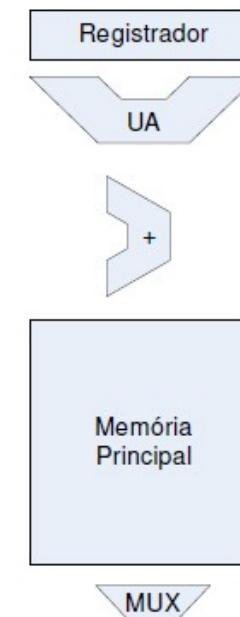
LD A	; ACC \leftarrow A
ADD B	; ACC \leftarrow ACC + B
SUB C	; ACC \leftarrow ACC - C
STO D	; D \leftarrow ACC
ADDI 2	; ACC \leftarrow ACC + 2
STO F	; F \leftarrow ACC

Computador básico: organização de um processador

- ❑ A organização é a implementação da arquitetura
- ❑ Tipicamente, representamos a organização, por um diagrama de blocos constituído por unidades funcionais interligadas para linhas e barras que representam fios usados para transportas sinais elétricos

Computador básico: organização de um processador

- ❑ Organização do BIP
 - ❑ Componentes básicos
 - ❑ Registradores
 - ❑ Unidade Aritmética (UA)
 - ❑ Somador
 - ❑ Memória Principal
 - ❑ Fios
 - ❑ Multiplexadores (MUX)



Computador básico: organização de um processador

❑ Organização do BIP (cont.)

- ❑ Registradores
 - ❑ PC (*Program Counter* ou Contador de Programa)
 - ❑ Indica o endereço da instrução a ser executada
 - ❑ IR (*Instruction Register* ou Registrador de Instrução)
 - ❑ Armazena a instrução a ser executada
 - ❑ ACC (Acumulador)
 - ❑ Armazena dados
- ❑ Unidade Aritmética (UA)
 - ❑ Realiza as operações de soma e subtração
- ❑ Somador
 - ❑ Usado para incrementar o valor do PC

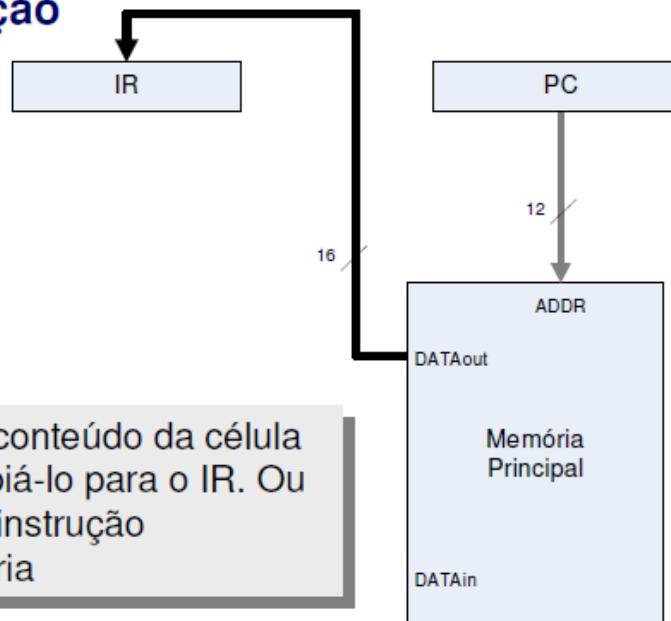
Computador básico: organização de um processador

❑ Organização do BIP (cont.)

- ❑ Memória principal
 - ❑ Armazena as instruções de programa e os dados
- ❑ Fios
 - ❑ Interconectam os componentes
- ❑ Multiplexadores (MUX)
 - ❑ São seletores de sinal para fios compartilhados

Computador básico: organização de um processador

❑ Busca da instrução

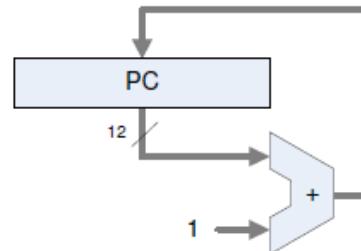


Consiste em buscar o conteúdo da célula indicada pelo PC e copiá-lo para o IR. Ou seja, o IR recebe uma instrução armazenada na memória

Observa-se que a via de endereços (cinza) tem 12 bits de largura e a de dados (preta) tem 16 bits,

Computador básico: organização de um processador

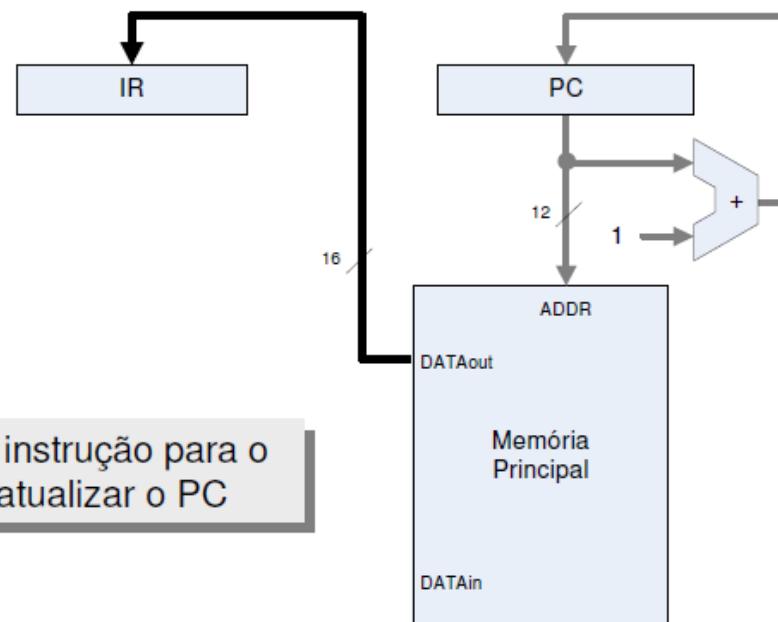
☐ Incremento do PC



Consiste em incrementar o valor do PC em uma unidade, fazendo-o indicar a próxima instrução a ser executada

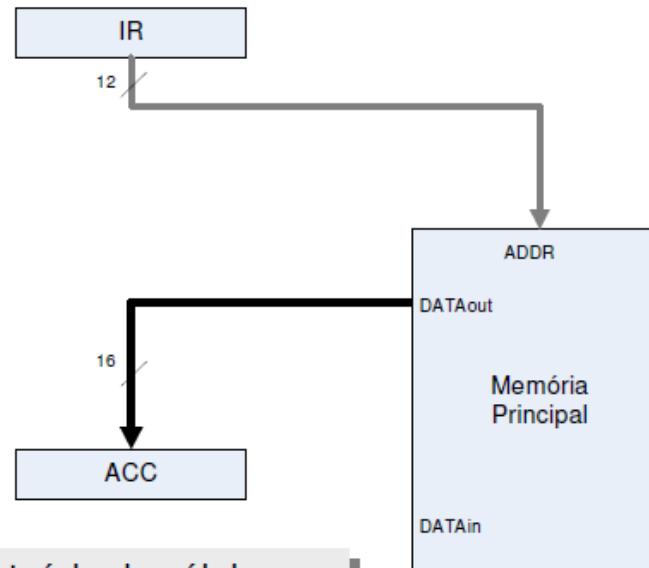
Computador básico: organização de um processador

❑ Busca da instrução e incremento do PC

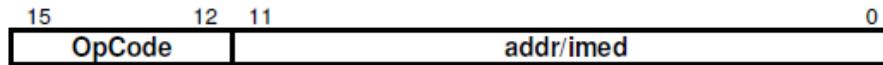


Computador básico: organização de um processador

❑ Operação LD

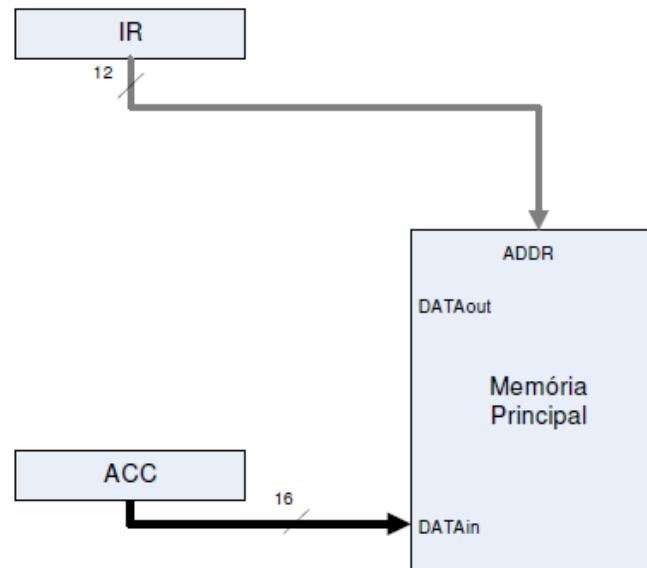


Copia para o ACC o conteúdo da célula indicada pelos doze bits mais à direita da instrução armazenada no IR, ou seja, pelo campo **addr**

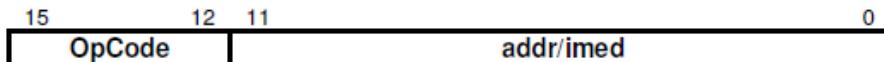


Computador básico: organização de um processador

□ Operação STO



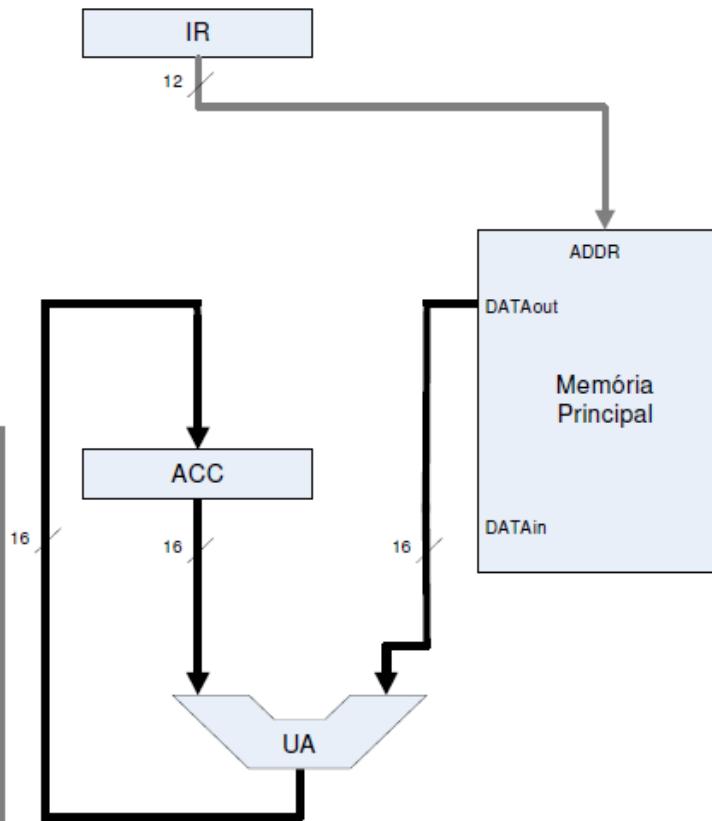
Copia o conteúdo do ACC para a célula indicada pelo campo **addr** da instrução armazenada no IR



Computador básico: organização de um processador

❑ Operações ADD/SUB

Soma (ou subtraí) o conteúdo do ACC com o conteúdo da célula indicada pelo campo **addr** da instrução mantida no IR, armazenando resultado da operação no ACC



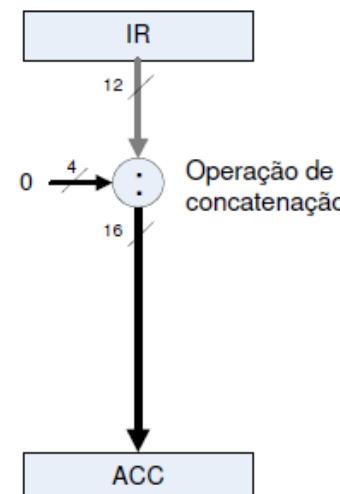
Computador básico: organização de um processador

❑ Operação LDI

Carrega no ACC os doze bits mais à direita da instrução armazenada no IR, ou seja, o campo **imed**.

OBS:

Como o acumulador possui 16 bits, é preciso concatenar 4 bits em 0 com os 12 bits do campo **imed**, formando uma palavra de 16 bits

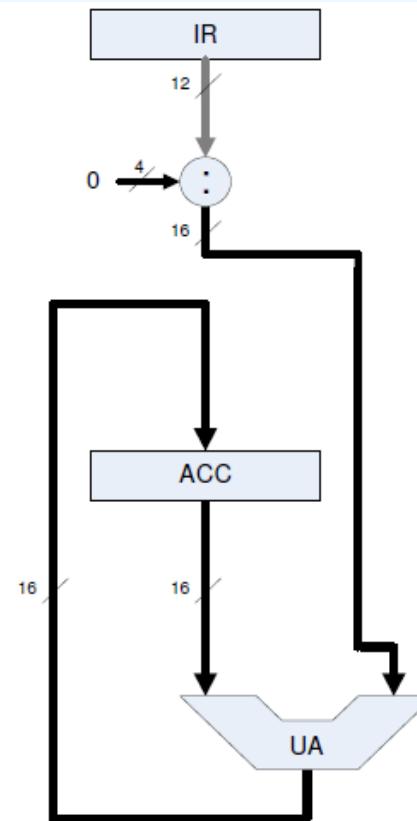


15	12	11	0
OpCode	addr/imed		

Computador básico: organização de um processador

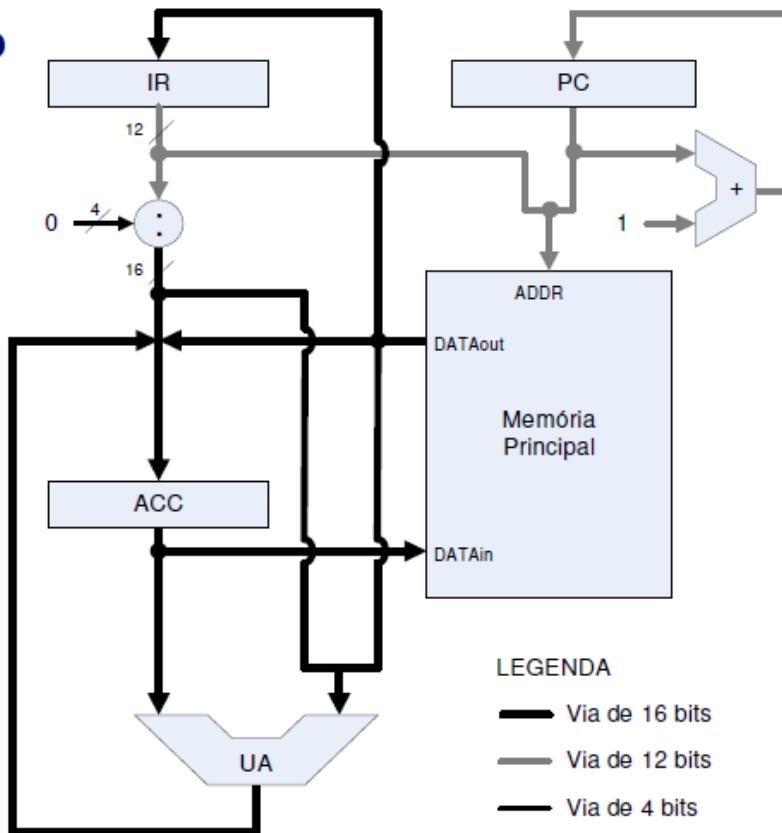
□ Operações ADDI/SUBI

Soma (ou subtrai) o conteúdo do ACC com o conteúdo do campo **imed** da instrução mantida no IR (com quatro bits em 0 à esquerda), armazenando resultado da operação no ACC



Computador básico: organização de um processador

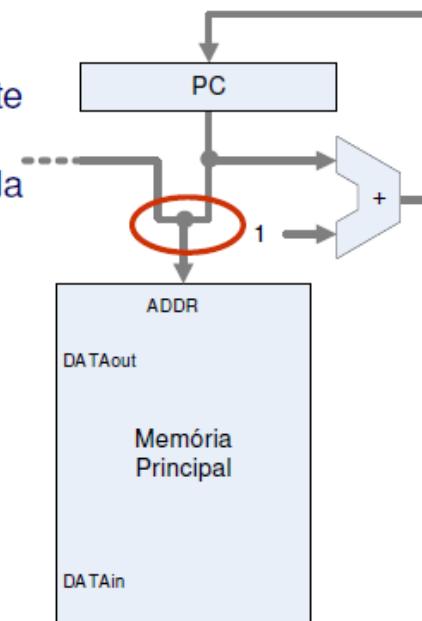
❑ Organização simplificada do BIP



Computador básico: organização de um processador

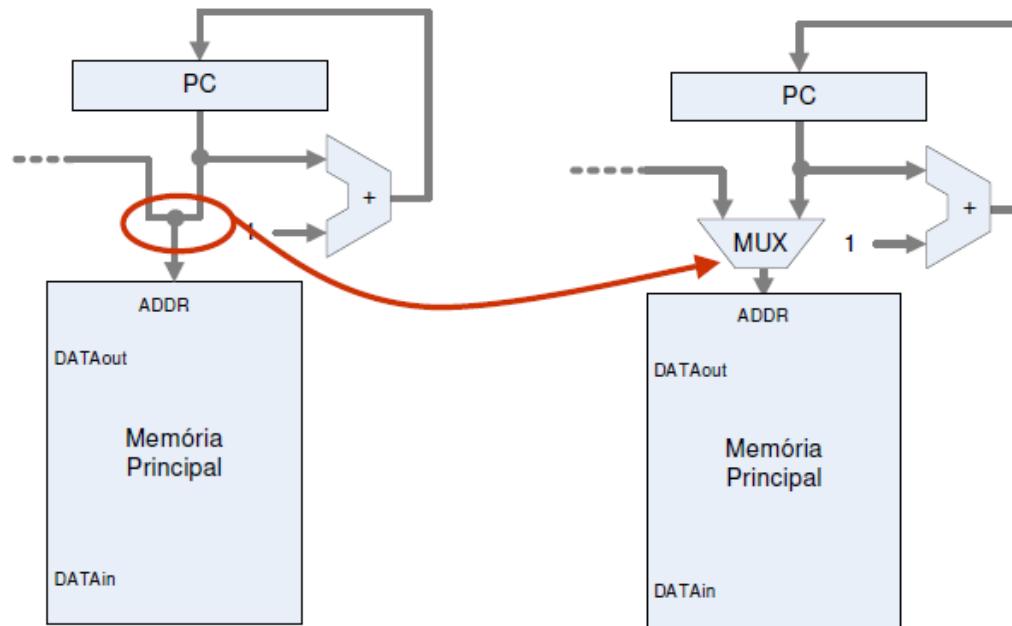
□ Limitação da organização simplificada do BIP

- Na prática, não é possível fazer com que duas vias compartilhem uma mesma entrada de um componente, conforme é destacado abaixo.
- É preciso utilizar um componente que realize a seleção de qual via deve ser conectada à entrada a cada momento.
- Esse seletor é chamado de multiplexador (ou MUX)
- VER PRÓXIMA PÁGINA



Computador básico: organização de um processador

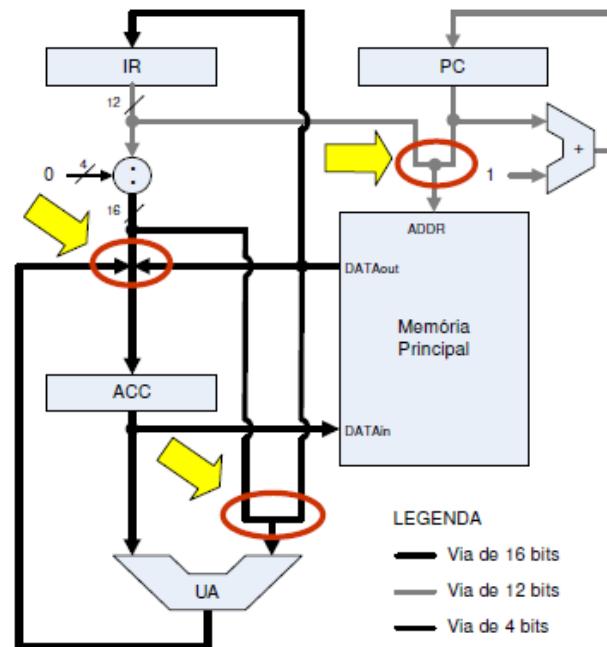
□ Limitação da organização simplificada do BIP



Computador básico: organização de um processador

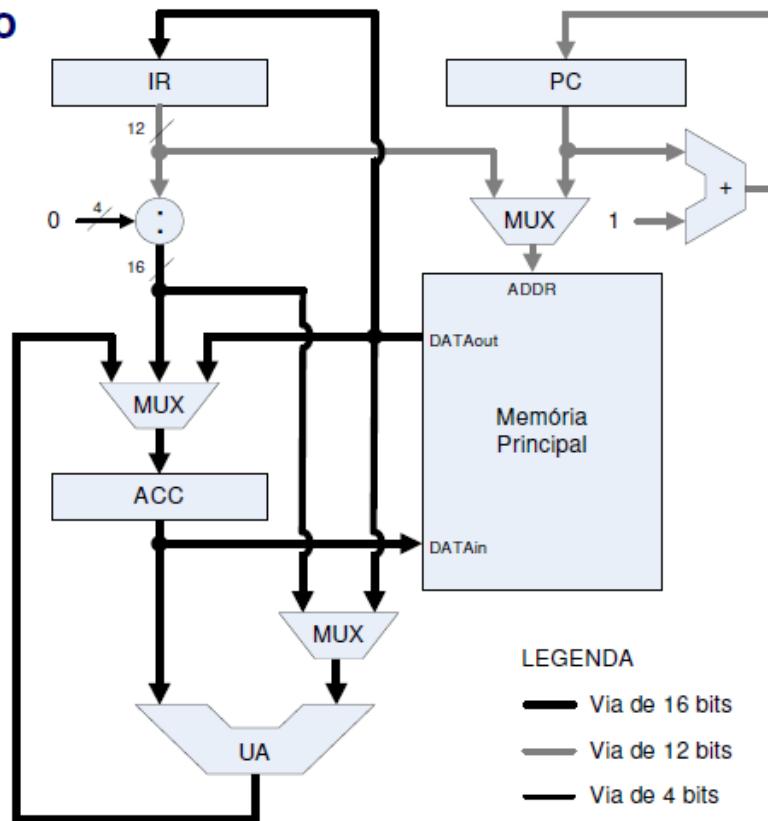
□ Limitação da organização simplificada do BIP

- Analisando-se a organização simplificada do BIP, percebe-se a necessidade de três multiplexadores (em destaque):



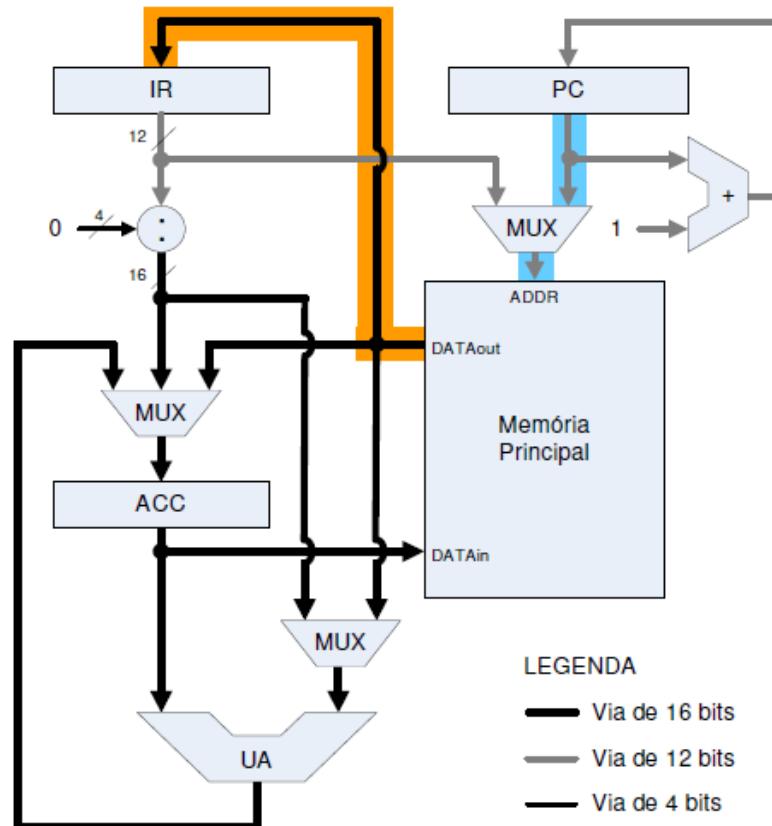
Computador básico: organização de um processador

❑ Organização completa do BIP



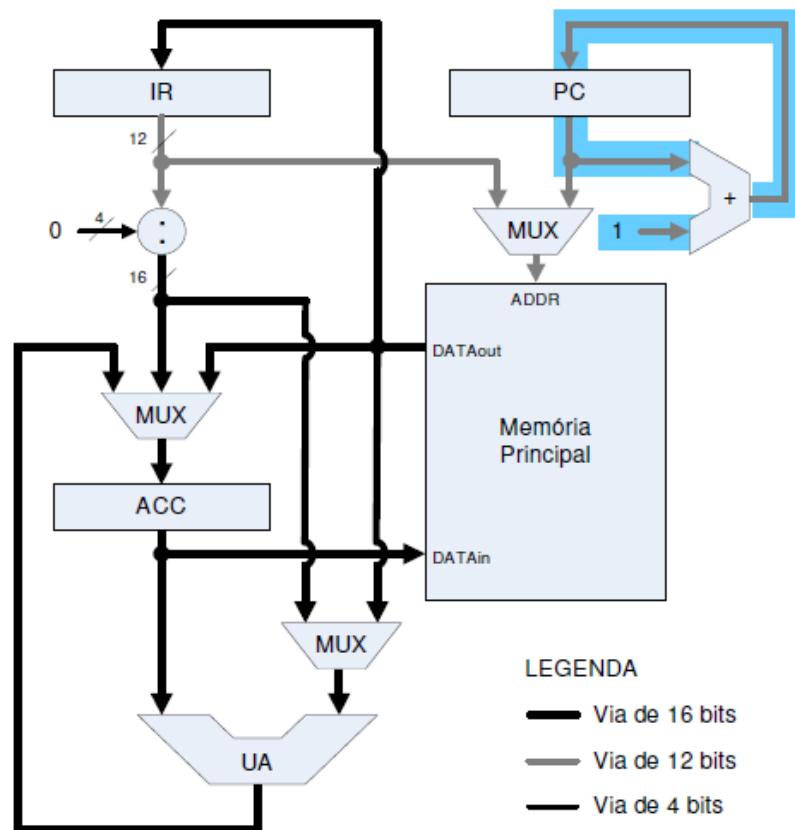
Computador básico: organização de um processador

□ Busca da instrução



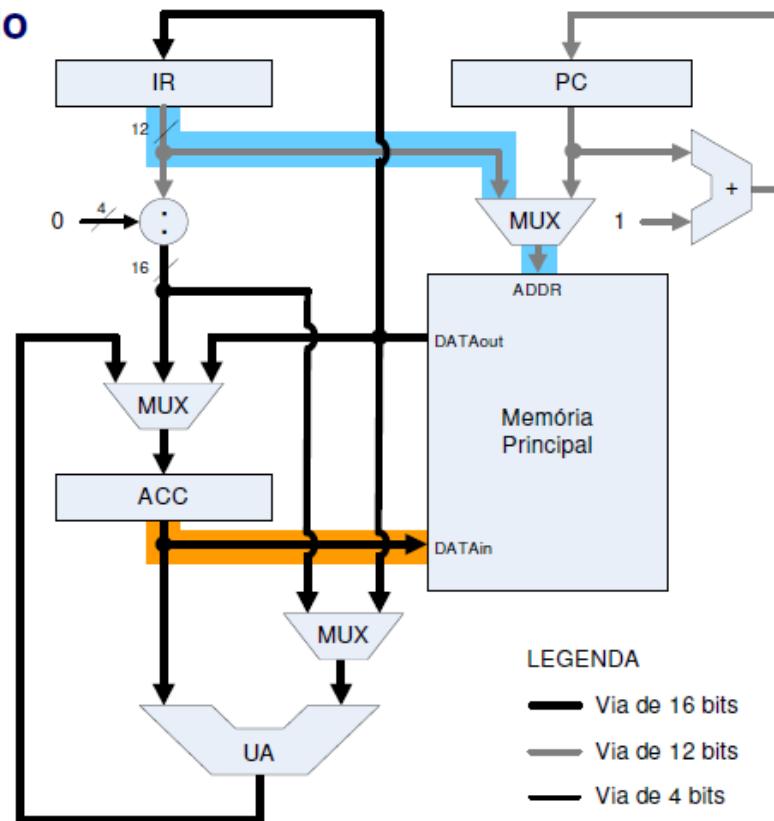
Computador básico: organização de um processador

❑ Incremento do PC



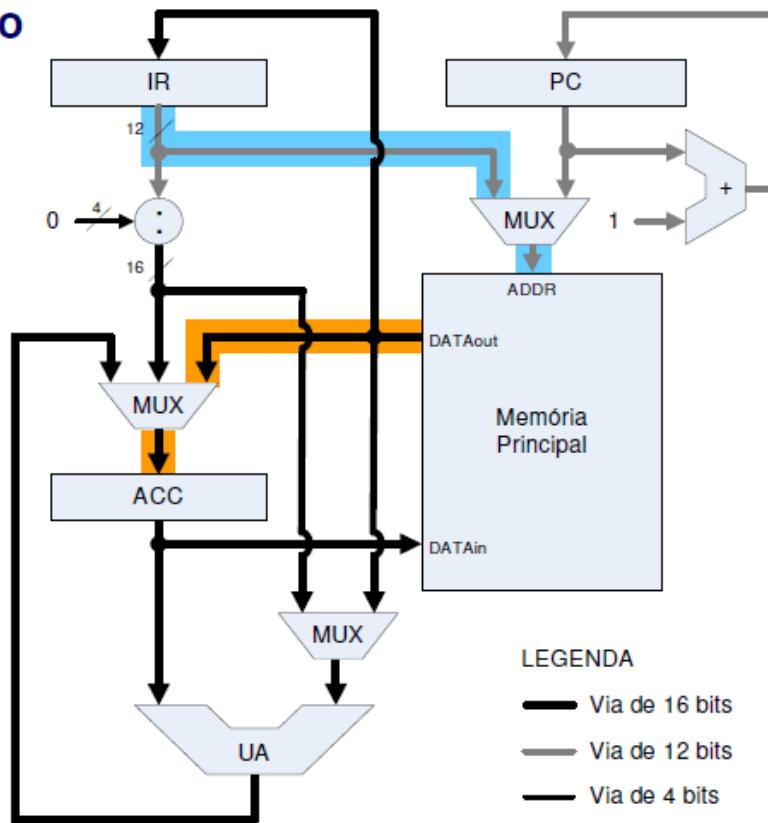
Computador básico: organização de um processador

Execução do STO



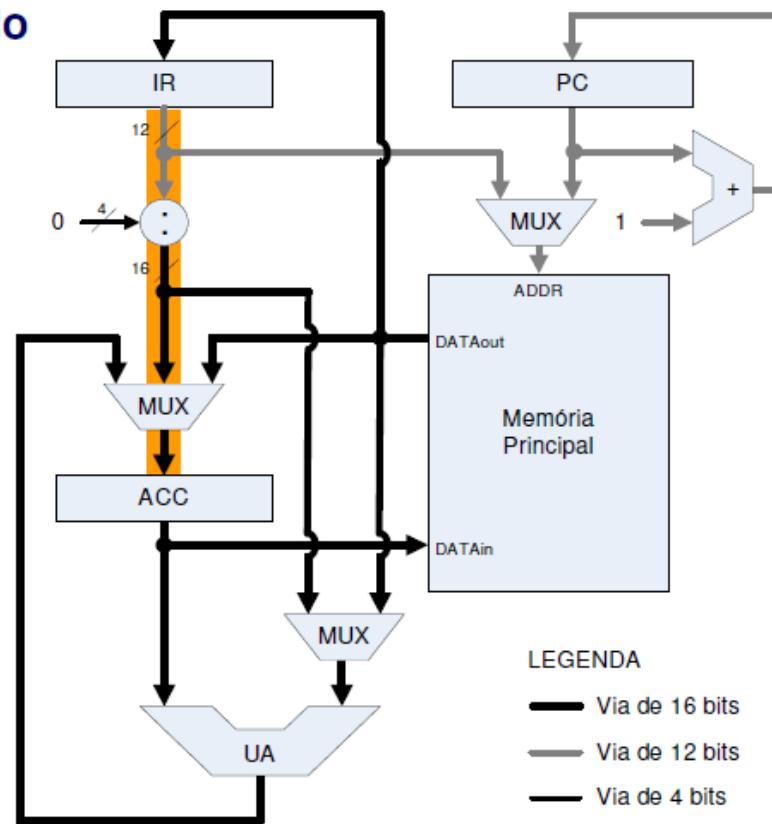
Computador básico: organização de um processador

Execução do LD



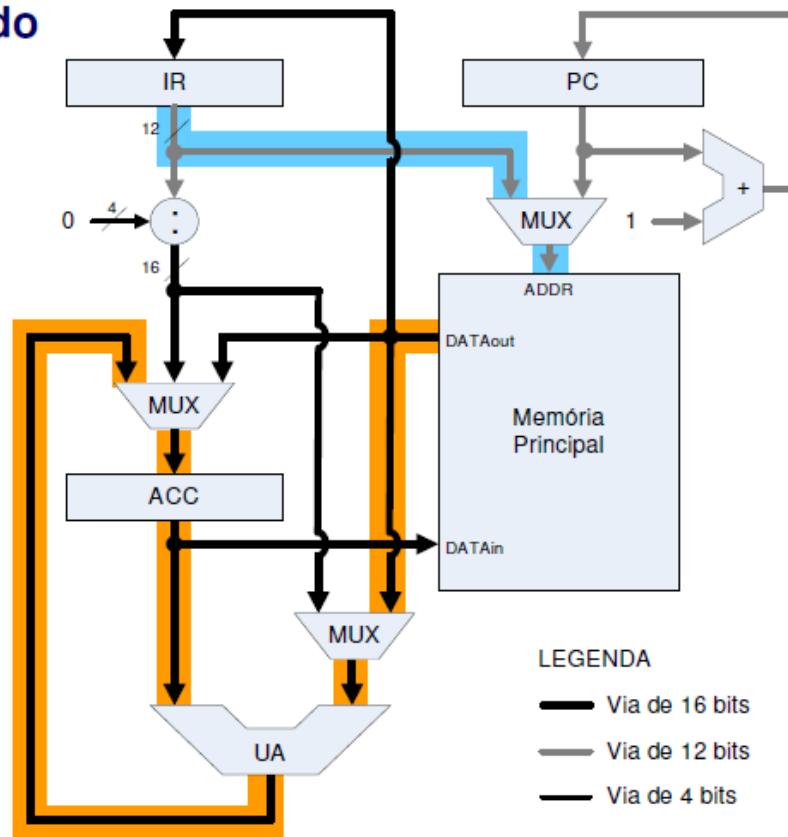
Computador básico: organização de um processador

Execução do LDI



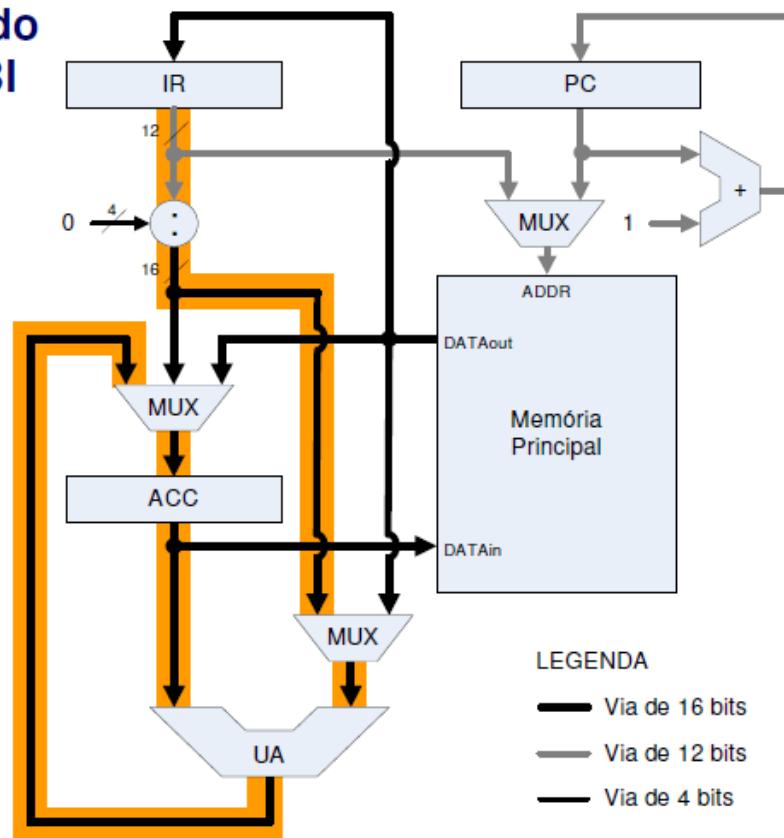
Computador básico: organização de um processador

Execução do ADD / SUB



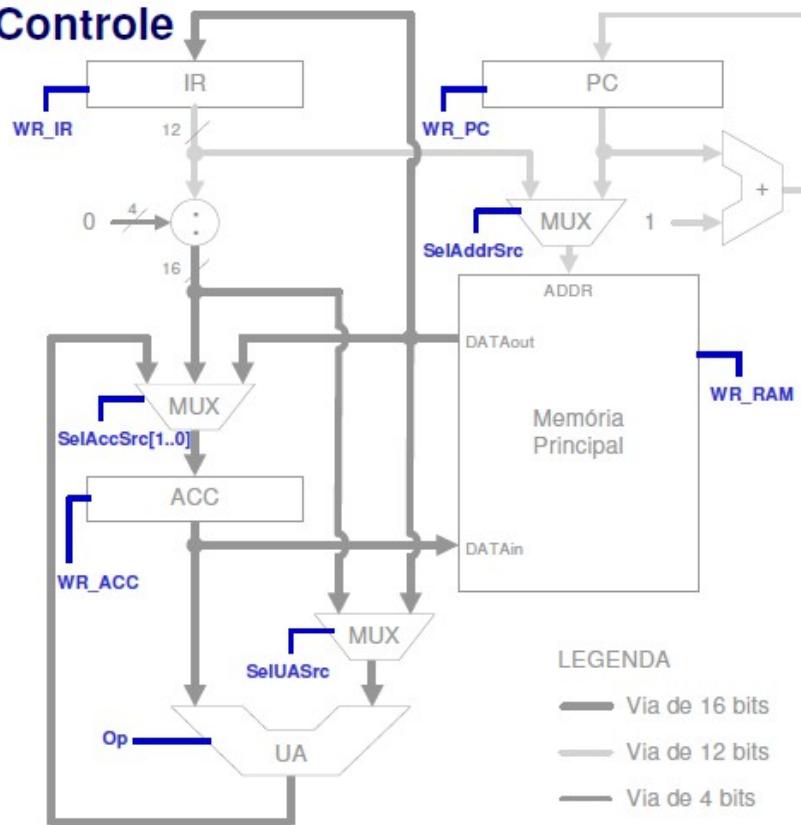
Computador básico: organização de um processador

Execução do ADDI / SUBI



Computador básico: organização de um processador

Sinais de Controle





Computador básico: organização de um processador

Sinais de Controle



FIM AULA 06