Projetando um Computador

Parte II – Arquitetura do Processador BIP

Prof. Dr. Cesar Albenes Zeferino

(zeferino@univali.br)

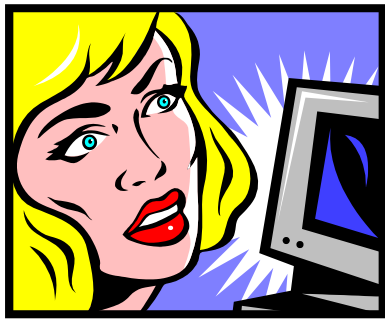
Objetivo

- Apresentar as noções do funcionamento de um computador através da descrição da arquitetura de um processador básico

Computador Básico: O Processador

- **O processador pode ser escrito em diferentes níveis de abstração (com menos ou mais de talhes)**
- **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: O Processador



Mais abstrato

ARQUITETURA

(Interface do programador)

ORGANIZAÇÃO

(Implementação)

Mais detalhado

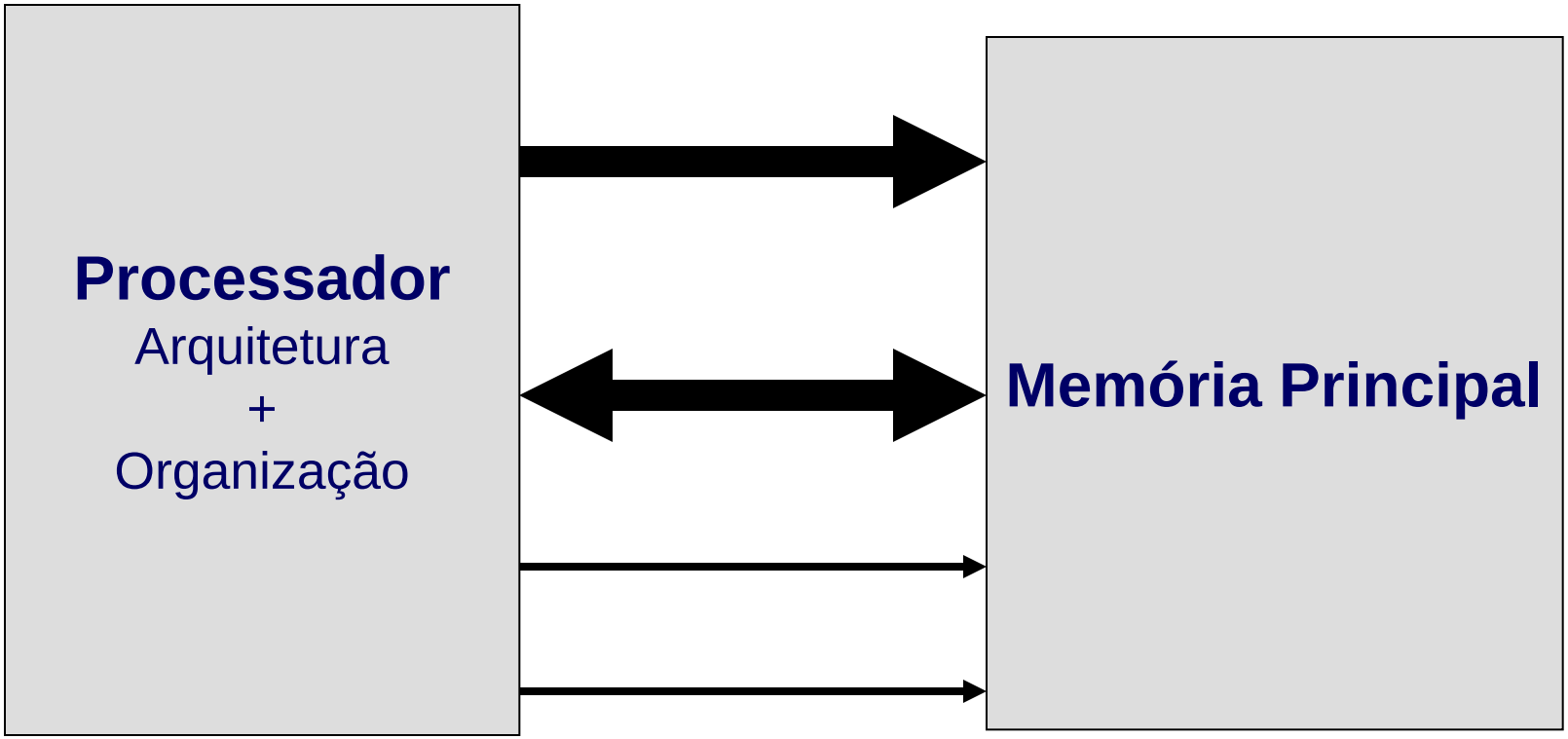
Cesar Zeferino

UNIVALI

Computador Básico: O 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: O Processador



Cesar Zeferino
UNIVALI

Atributos arquiteturais de um processador

- **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)

Atributos arquiteturais de um processador

- **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

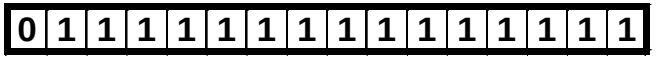
O Processador BIP

- **BIP (Basic Instruction-set Processor)**
- **Arquitetura de 16 bits**
- **Duas versões:**
  - BIP I
    - Instruções básicas para a implementação de equações baseadas em operações aritméticas de soma e subtração com número inteiros
  - BIP II
    - Acrescenta instruções de desvio

Arquitetura do BIP

- **Tamanho da palavra de dados**
  - 16 bits

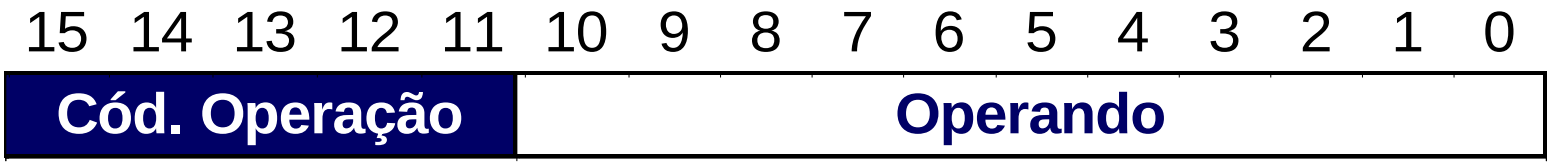
- **Tipo de dado**
  - Inteiro com sinal
    - Variando de –32768 a 32767



    - Números negativos representados em complemento 2

Arquitetura do BIP

- **Tamanho da palavra de instrução**
  - 16 bits
- **Um formato de instrução com dois campos**
  - Código da operação de 5 bits
  - Operando explícto de 11 bits que pode representar
    - O endereço de uma variável na memória (addr)
    - Uma constante imediata (imed)



Arquitetura do BIP

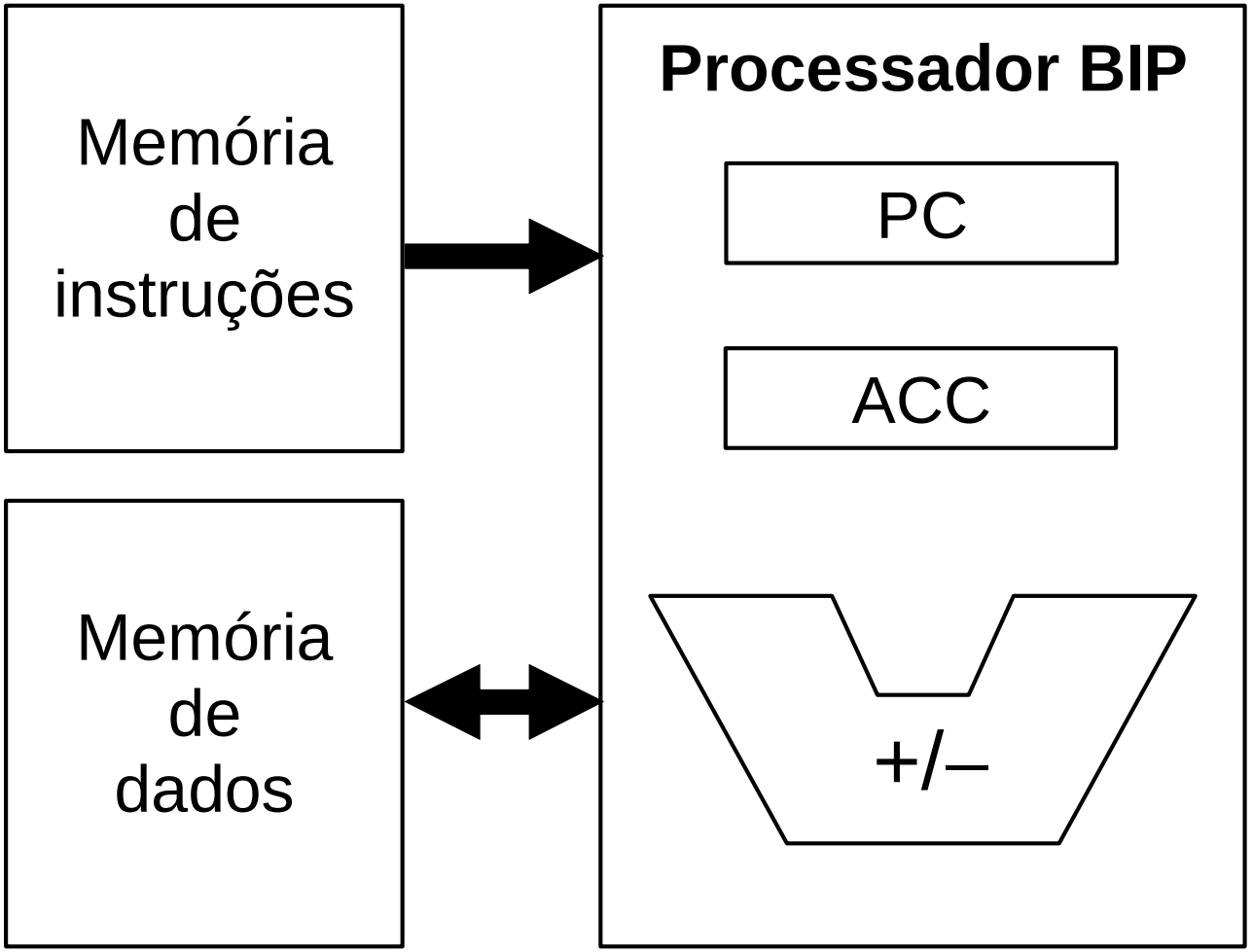
- **Modos de endereçamento**
  - Todas as instruções envolvem uma operação entre o operando explícito da instrução e um operando implícito
  - O operando implícito é um registrador de uso geral do processador denominado acumulador (ACC)
  - Tipos de modos de endereçamento
    - **Direto**
      - O dado a ser processado é uma variável na memória apontada pelo operando da instrução (endereço)
    - **Imediato**
      - o dado a ser manipulado é o próprio operando (imediato) da instrução

Arquitetura do BIP

- **Registradores**
  - PC (Program Counter)
    - Registrador de propósito específico
    - Indica o endereço na memória da instrução a ser executada
    - É incrementado a cada instrução executada (PC++) para que o programa avance para a instrução seguinte
  - ACC (Accumulator)
    - Registrador de propósito geral
    - Serve para armazenamento temporário de dados no processador
    - É um operando implícito nas instruções

Arquitetura do BIP

- **Modelo simplificado do processador**



Arquitetura do BIP

- **Conjunto de instruções**
  - Três classes de instrução
    - Controle
      - Controle da execução do processador
    - Aritmética
      - Soma e subtração
    - Transferência
      - Transferência de dados entre processador e memória e carga do acumulador
  - Cada instrução possui um mnemônico (apelido) para facilitar a sua referência
    - ex. ADD ao invés de 00100

Arquitetura do BIP

Conjunto de instruções

Instruções de controle

HLT	Halt	paralisa a execução do programa

Instruções de aritmética

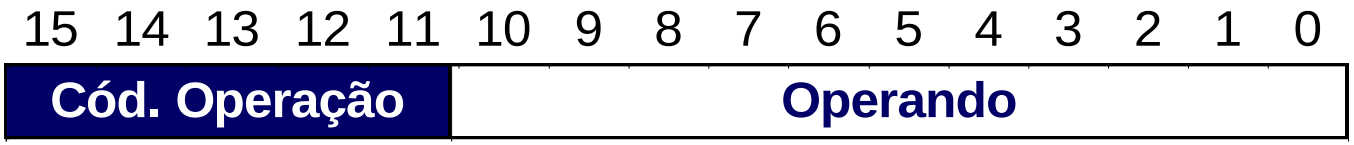
ADD	Add	Soma uma variável ao ACC
ADDI	Add Immediate	Soma uma constante ao ACC
SUB	Subtract	Subtrai uma variável do ACC
SUBI	Sub. Immediate	Subtrai uma constante do ACC

Instruções de transferência

STO	Store	Copia o conteúdo do ACC para uma variável
LD	Load	Copia o conteúdo de uma variável para o ACC
LDI	Load Immediate	Carrega uma constante no ACC

Arquitetura do BIP

Conjunto de instruções



Cód. operação	Instrução	Operação
00000	**HLT**	Paralisa a execução
00001	**STO** endereço	(endereço) ← ACC
00010	**LD** endereço	ACC ← (endereço)
00011	**LDI** constante	ACC ← constante
00100	**ADD** endereço	ACC ← ACC + (endereço)
00101	**ADDI** constante	ACC ← ACC + constante
00110	**SUB** endereço	ACC ← ACC – (endereço)
00111	**SUBI** constante	ACC ← ACC – constante
01000 - 11111	Reservados para as futuras gerações	

Legenda:

← atribuição

( ) conteúdo

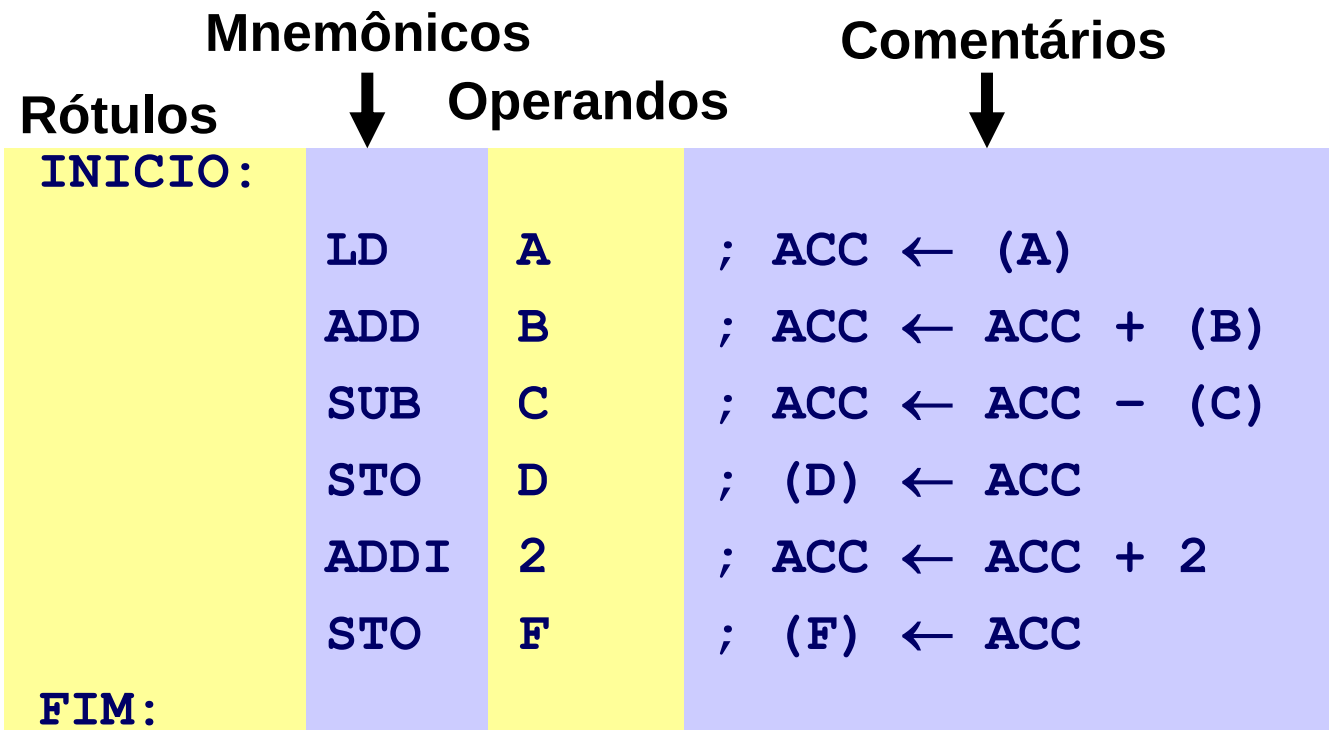
Programação do BIP

- **Estrutura de código na linguagem de montagem**
  - Não identado
  - Apenas uma instrução por linha
  - Não possui marcador de fim de linha
  - Código organizado em quatro colunas (tabuladas)
    - Rótulos
    - Mnemônicos
    - Operando(s)
    - Comentários opcionais (precedidos do caracter “;”)
  - Uso intensivo de comentários

Um rótulo (*label*) é uma abstração que permite referenciar de forma facilitada um endereço do programa, tipicamente usada em instruções de desvio.

Programação do BIP

Exemplo



Programação do BIP

Abstração	Código em C	Código na ling. de montagem
Atribuição de uma constante	`A = 10;`	`LDI 10 ; ACC ← 10` `STO A ; (A) ← ACC`
Atribuição de uma variável	`A = B;`	`LD B ; ACC ← (B)` `STO A ; (A) ← ACC`

Comando com uma operação aritmética	`A = A + 1;`	`LD A ; ACC ← (A)`
		`ADDI 1 ; ACC ← ACC + 1`
		`STO A ; (A) ← ACC`

Comando com múltiplas operações aritméticas	`A = A + B - 3;`	`LD A ; ACC ← (A)`
		`ADD B ; ACC ← ACC + (B)`
		`SUBI 3 ; ACC ← ACC - 3`
		`STO A ; (A) ← ACC`

Notas

(1) A e B são nomes de variáveis armazenadas em memória

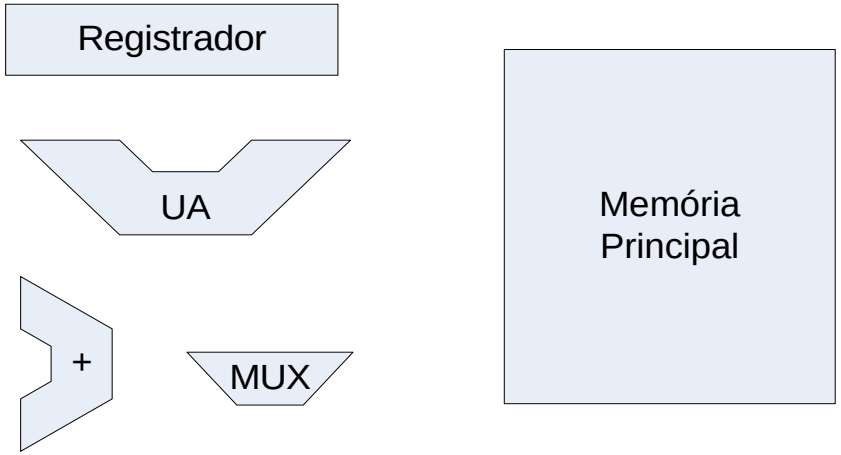
(2) Para atualizar uma variável não é necessário copiar seu conteúdo para o acumulador, salvo se a instrução utilizar esse conteúdo como fonte da operação

Exercícios

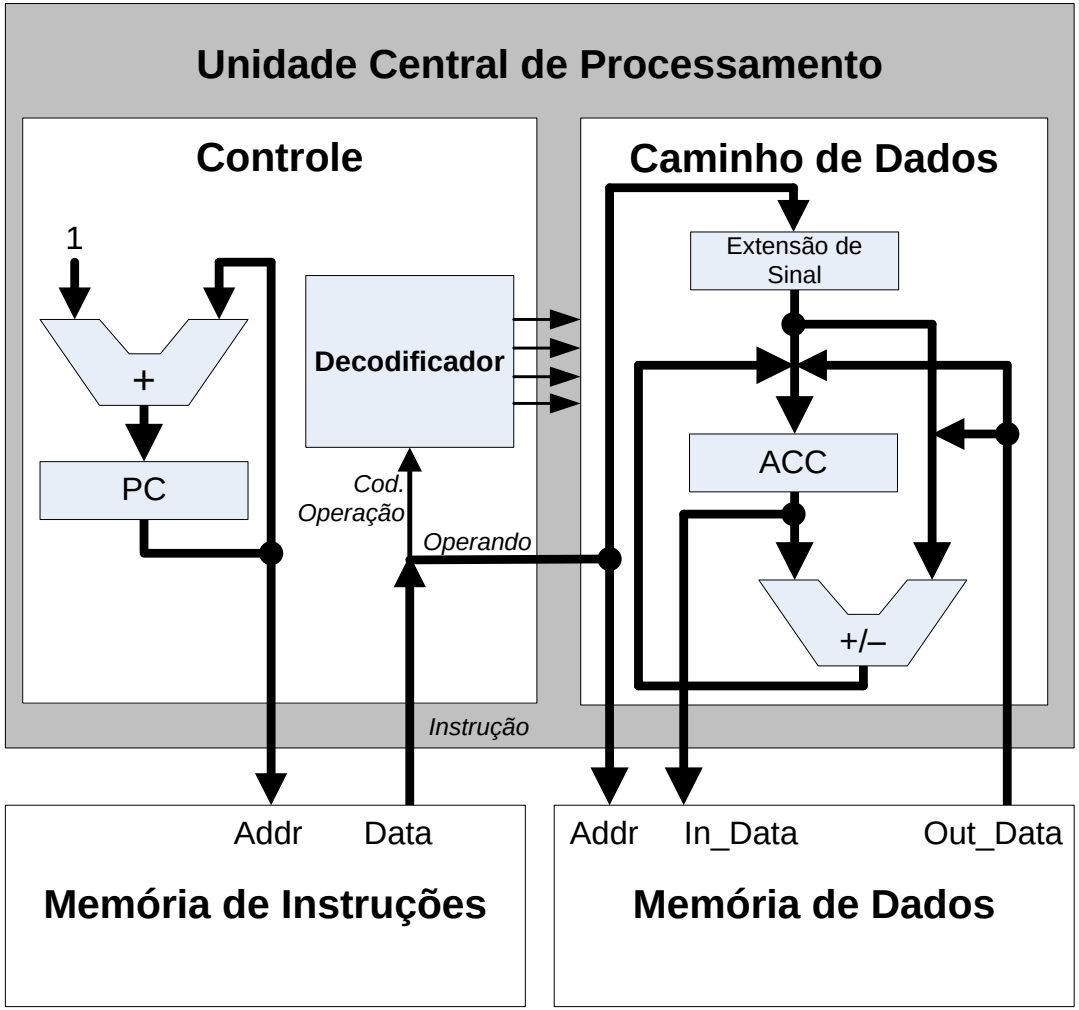
- Ver listas de exercícios

Organização do BIP I

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

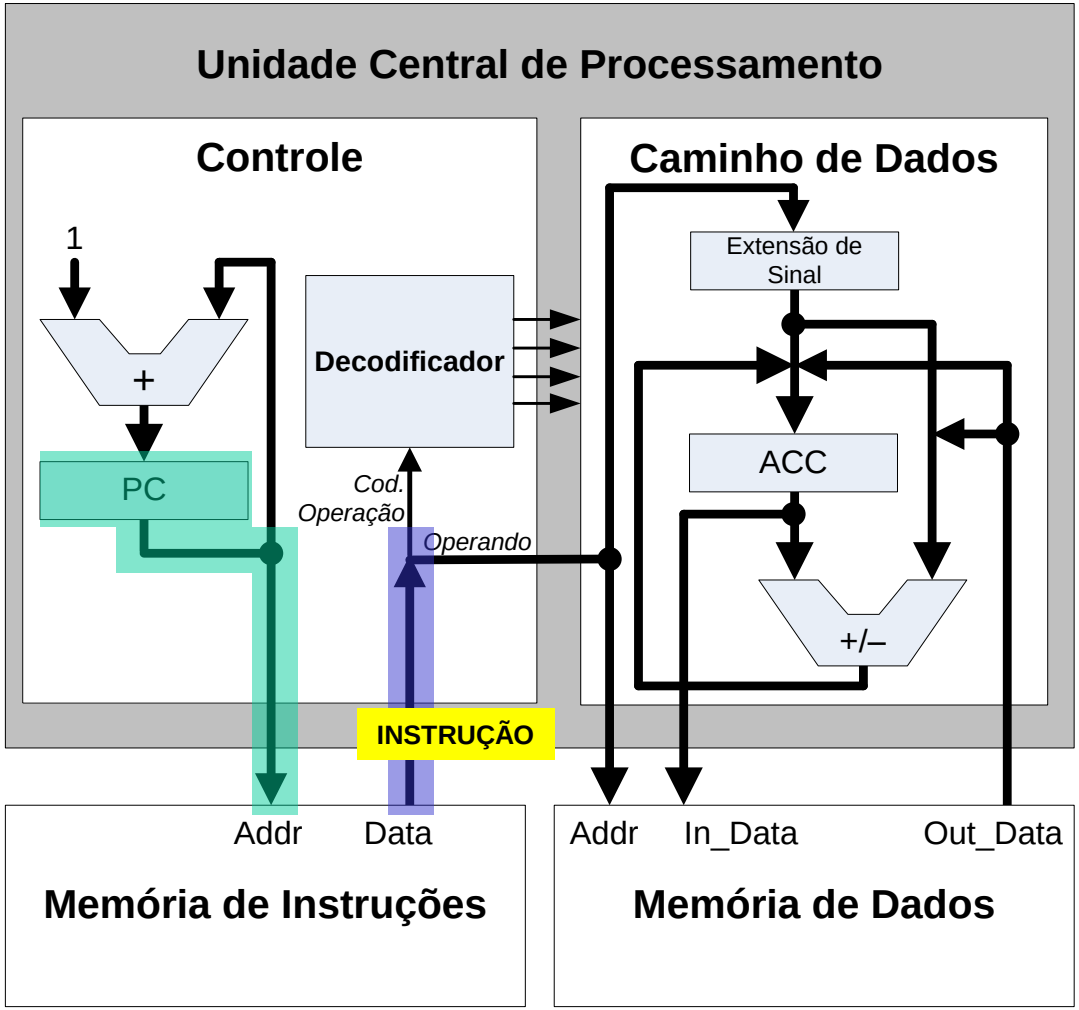


Organização do BIP I



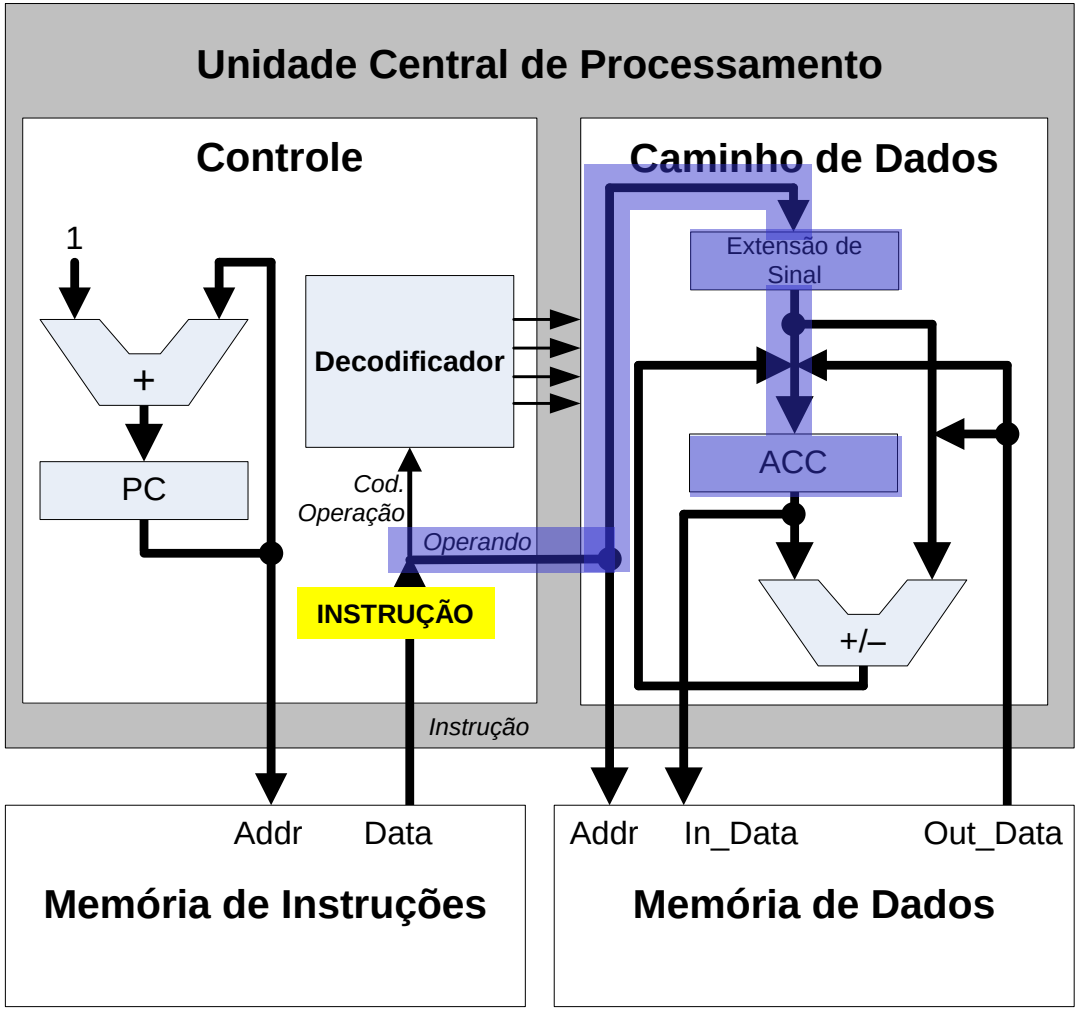
Organização do BIP I

Busca de uma instrução



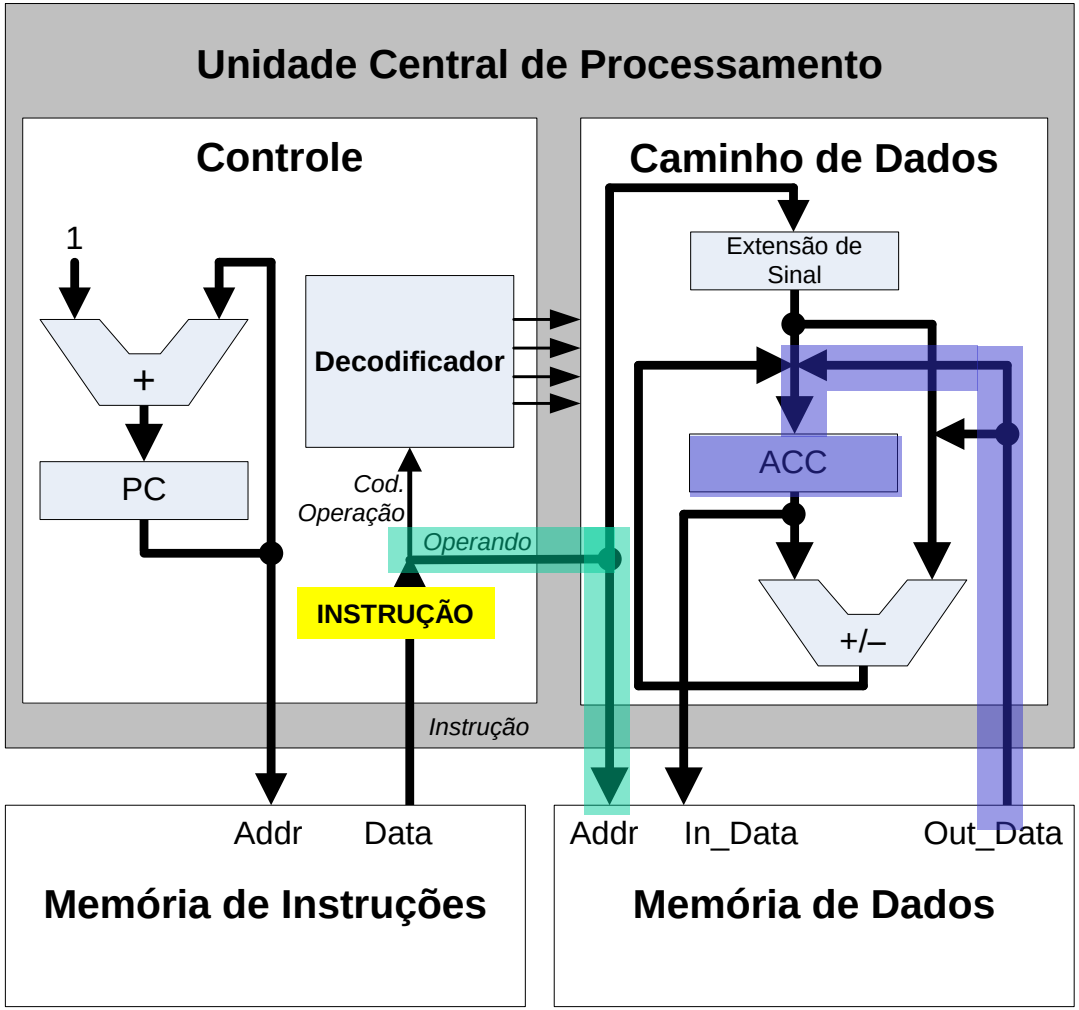
Organização do BIP I

Execução da instrução LDI



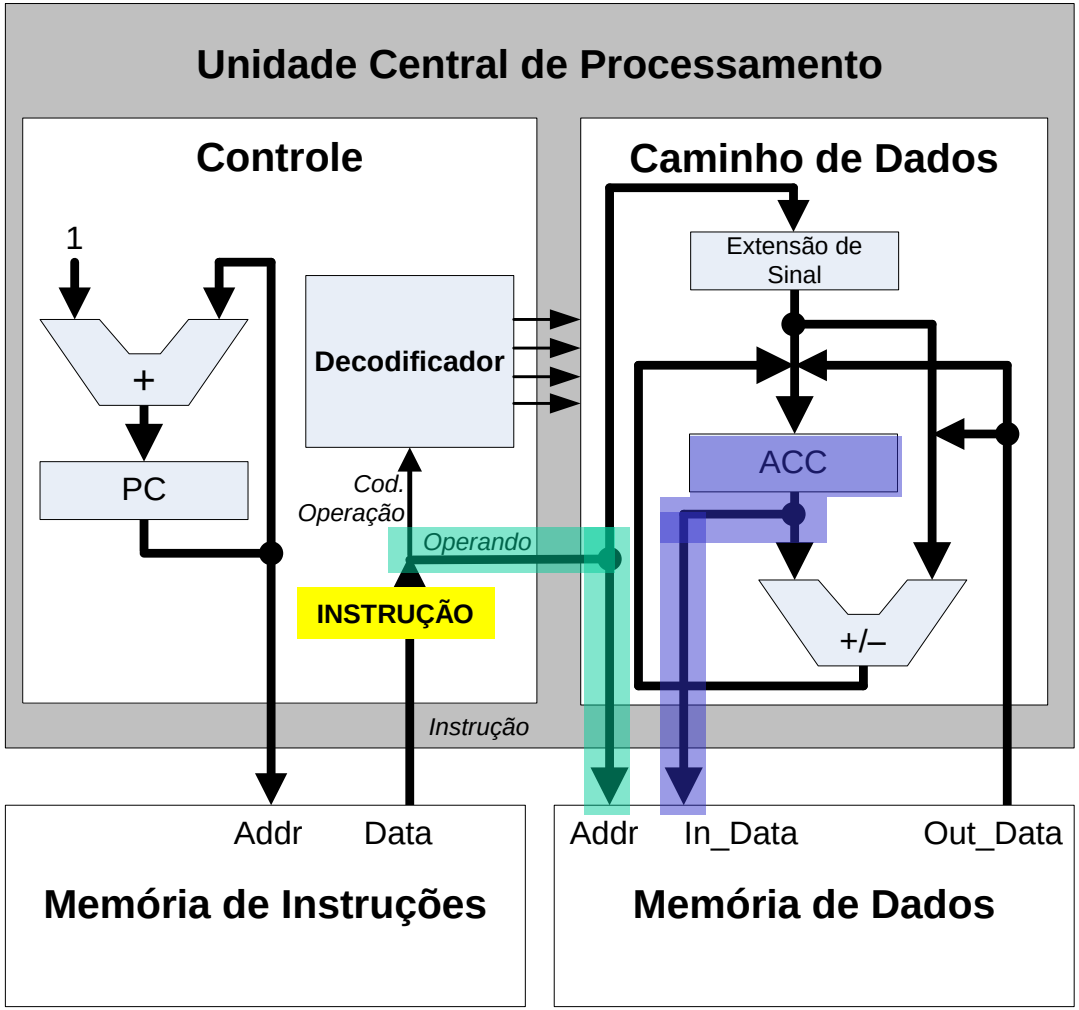
Organização do BIP I

Execução da instrução LD



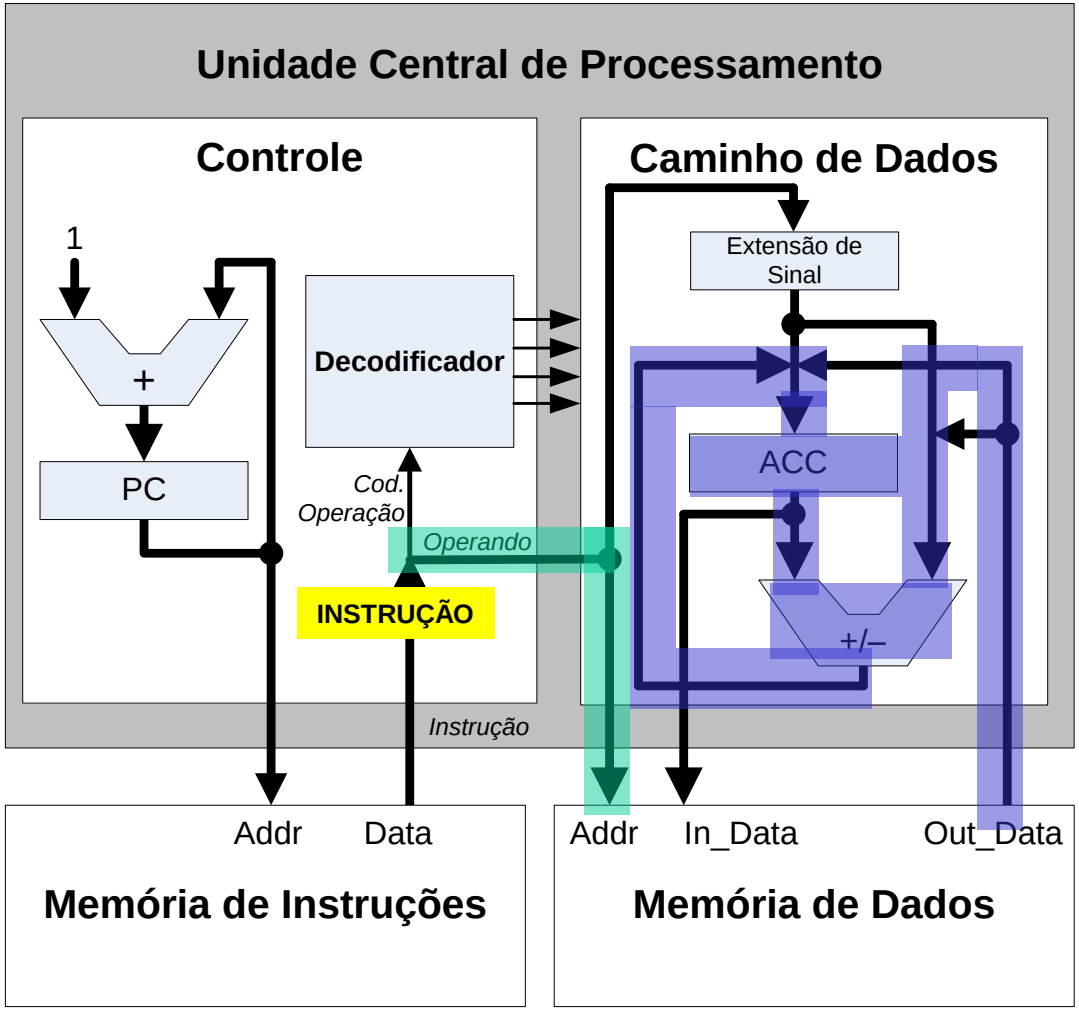
Organização do BIP I

Execução da instrução STO



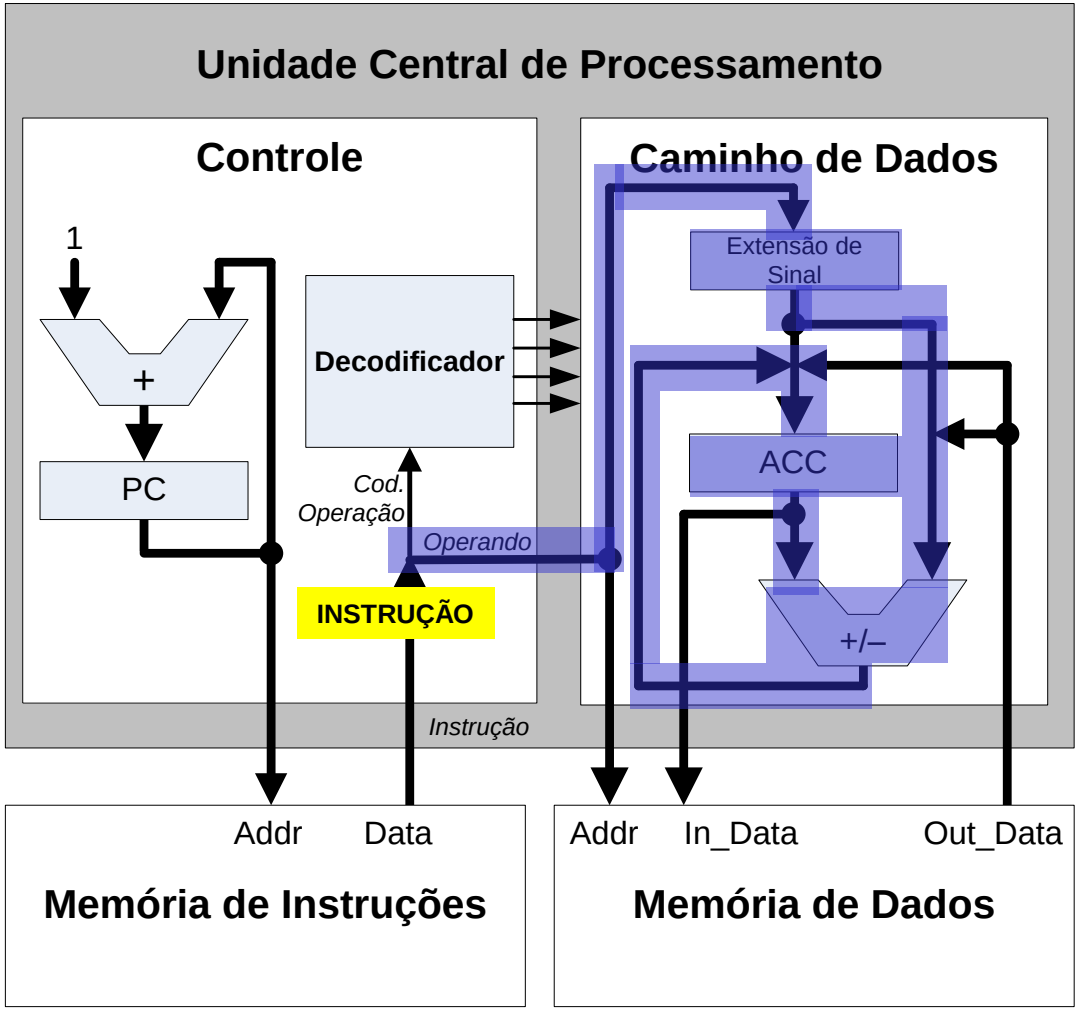
Organização do BIP I

Execução da instrução ADD



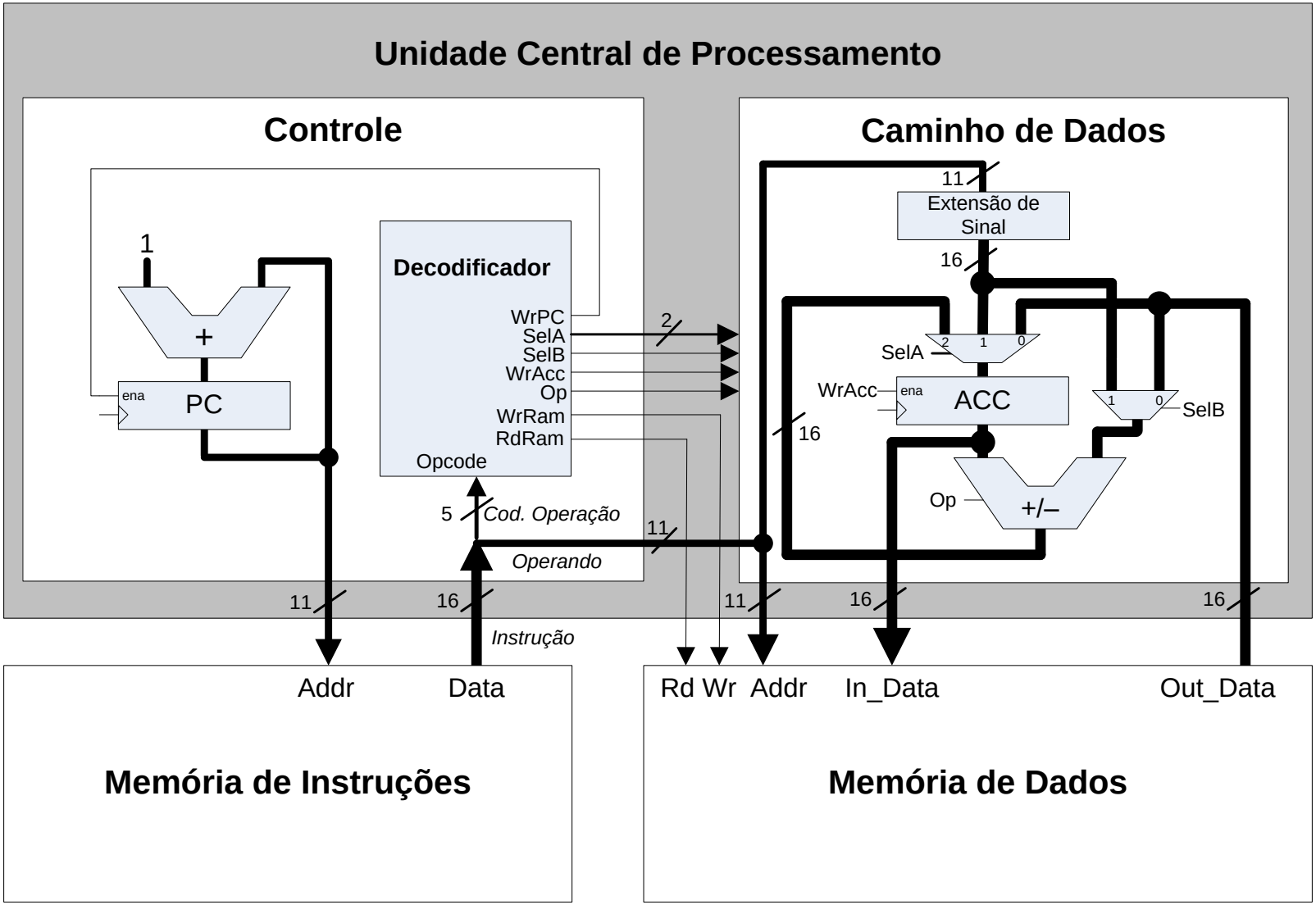
Organização do BIP I

Execução da instrução ADDI



Organização do BIP I

Organização detalhada



Arquitetura estendida do BIP – BIP II

- **Inclui um registrador com dois bits de estado**
  - STATUS.Z (Zero)
    - Resultado da última operação aritmética = 0
  - STATUS.N (Negative)
    - Resultado da última operação aritmética < 0
- **Inclui duas classes adicionais de instrução**
  - Desvio incondicional
    - Muda o fluxo de execução do programa, atualizando o PC com o endereço de instrução especificado
  - Desvio condicional
    - Se a condição especificada for verdadeira, desvia para o endereço de instrução especificado

Arquitetura estendida do BIP – BIP II

Conjunto de instruções estendido

Instrução de desvio incondicional

- JMP Jump — Desvia incondicionalmente

Instruções de desvio condicional

- Devem ser precedidas por uma operação SUB ou SUBI sobre os operandos a serem comparados

- BEQ Branch on Equal — Desvia se igual
- BNE Branch on Not Equal — Desvia se não igual
- BGT Branch on Greater Than — Desvia se maior que
- BGE Branch on Greater or Equal — Desvia se maior ou igual que
- BLT Branch on Less Than — Desvia se menor que
- BLE Branch on Less or Equal — Desvia se menor ou igual que

- Exemplo

```
LD    A
SUBI  2
BLE   addr    ; PC = addr se (A) <= 2
```

Arquitetura estendida do BIP – BIP II

Cód. operação	Instrução	Operação
01000	**BEQ** endereço	Se (STATUS.Z=1) então PC ← endereço Se não PC ← PC + 1
01001	**BNE** endereço	Se (STATUS.Z=0) então PC ← endereço Se não PC ← PC + 1
01010	**BGT** endereço	Se (STATUS.Z=0) e (STATUS.N=0) então PC ← endereço Se não PC ← PC + 1
01011	**BGE** endereço	Se (STATUS.N=0) então PC ← endereço Se não PC ← PC + 1
01100	**BLT** endereço	Se (STATUS.N=1) então PC ← endereço Se não PC ← PC + 1
01101	**BLE** endereço	Se (STATUS.Z=1) ou (STATUS.N=1) então PC ← endereço Se não PC ← PC + 1
01110	**JMP** endereço	PC ← endereço
01111 - 11111	Reservados para as futuras gerações	

Programação do BIP II

Abstração	Código em C	Código na ling. de montagem

Teste de condição tipo if-then	`if (A==B) {`
	`  // Bloco 1`
	`}`
	`// Bloco 2`
		`    LD   A      ; ACC ← A`
		`    SUB  B      ; ACC ← ACC - B`
		`    BNE  L1`
		`    ...         ; Bloco 1`
		`L1: ...         ; Bloco 2`

Teste de condição tipo if-then-else	`if (A==B) {`
	`  // Bloco 1`
	`} else {`
	`  // Bloco 2`
	`}`
	`// Bloco 3`
		`    LD   A      ; ACC ← A`
		`    SUB  B      ; ACC ← ACC - B`
		`    BNE  L1`
		`    ...         ; Bloco 1`
		`    JMP  L2`
		`L1: ...         ; Bloco 2`
		`L2: ...         ; Bloco 3`

Programação do BIP II

Abstração	Código em C	Código na ling. de montagem

Laço de repetição do tipo while	`i = 0;`
	`while (i<10) {`
	`  // Bloco 1`
	`  i++;`
	`}`
	`// Bloco 2`
		`  LDI  0  ; ACC ← 0`
		`  STO  I  ; I ← ACC`
		`L1: SUBI 10  ; ACC ← ACC - 10`
		`  BGE  L2`
		`  ...  ; Bloco 1`
		`  LD  I  ; ACC ← I`
		`  ADDI 1  ; ACC ← ACC + 1`
		`  STO  I  ; I ← ACC`
		`  JMP  L1`
		`L2: ...  ; Bloco 2`

Laço de repetição do tipo for	`for(i=0;i<10;i++){`
	`  // Bloco 1`
	`}`
	`// Bloco 2`
		`Igual ao laço while`

Programação do BIP II

Abstração	Código em C	Código na ling. de montagem

Laço de repetição do tipo do-while	`i = 0;`
	`do {`
	`  // Bloco 1`
	`  i++;`
	`} while (i<10)`
	`// Bloco 2`
		`    LDI  0      ; ACC ← 0`
		`    STO  I      ; I   ← ACC`
		`L1: ...        ; Bloco 1`
		`    LD   I      ; ACC ← I`
		`    ADDI 1      ; ACC ← ACC + 1`
		`    STO  I      ; I   ← ACC`
		`    SUBI 10     ; ACC ← ACC - 10`
		`    BLT  L1`
		`    ...         ; Bloco 2`