UNIVALI

Escola Politécnica

Arquitetura, Programação e Organização do Processador BIP

Histórico de revisões

Revisão	Data	Responsável	Descrição
0.1	- X -	Prof. Cesar Zeferino	Primeira versão
0.2	03/2016	Prof. Cesar Zeferino	Revisão do modelo e atualização de conteúdo
0.3	10/2016	Prof. Cesar Zeferino	Revisão
0.4	05/2025	Prof. Cesar Zeferino	Revisão

Introdução

- **Objetivo**
  - Conhecer conceitos sobre arquitetura, programação e organização de computadores com base no estudo de um processador básico

- **Conteúdo**
  - Computador básico
  - Atributos arquiteturais
  - O processador BIP
  - Arquitetura, programação e organização do BIP I
  - Arquitetura, programação e organização do BIP II
  - A IDE Bipide

Bibliografia

- ZEFERINO, Cesar Albenes; RAABE, André Luis Alice; VIEIRA, Paulo Viniccius; PEREIRA, Maicon Carlos. Um Enfoque Interdisciplinar no Ensino de Arquitetura de Computadores. In: MARTINS, Carlos Augusto Paiva da Silva; NAVAUX, Philippe Olivier Alexandre; AZEVEDO, Rodolfo Jardim de; KOFUJI, Sérgio Takeo (Org.). **Arquitetura de Computadores: educação, ensino e aprendizado**. 1ed. Porto Alegre: Sociedade Brasileira de Computação (SBC), 2012, p. 165-193. Disponível em: <https://www.researchgate.net/publication/391597575_Capitulo_6_Um_Enfoque_Interdisciplinar_no_Ensino_de_Arquitetura_de_Computadores>. Acesso em: 09 maio 2025.

- VIEIRA, Paulo Viniccius; RAABE, André Luis Alice; ZEFERINO, Cesar Albenes. Bipide: ambiente de desenvolvimento integrado para a arquitetura dos processadores BIP. **Revista Brasileira de Informática na Educação**, v. 18, p. 32-43, 2010. Disponível em: <http://milanesa.ime.usp.br/rbie/index.php/rbie/article/download/1215/1111>. Acesso em: 09 maio 2025.

- PEREIRA, Maicon Carlos; VIEIRA, Paulo Viniccius; RAABE, André Luis Alice; ZEFERINO, Cesar Albenes. A basic processor for teaching digital circuits and systems design with FPGA. In: SOUTHERN CONFERENCE ON PROGRAMMABLE LOGIC (SPL), 8. , Bento Gonçalves, 2012. **Proceedings…** New York: IEEE, 2012. pp. 1-6, doi: 10.1109/SPL.2012.6211804.

Computador básico: O Processador

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

Mais abstrato

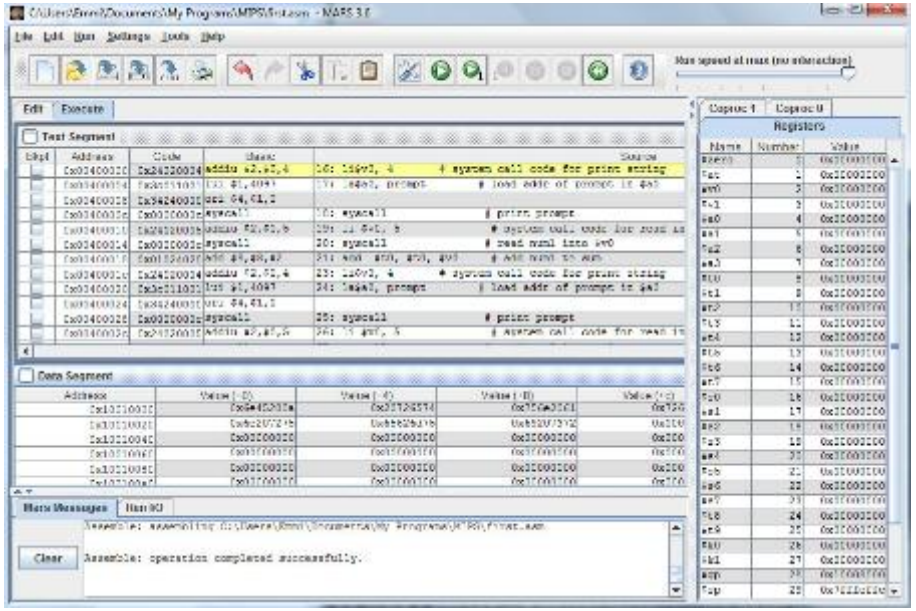
Arquitetura
(Interface do programador)

Organização
(Implementação)

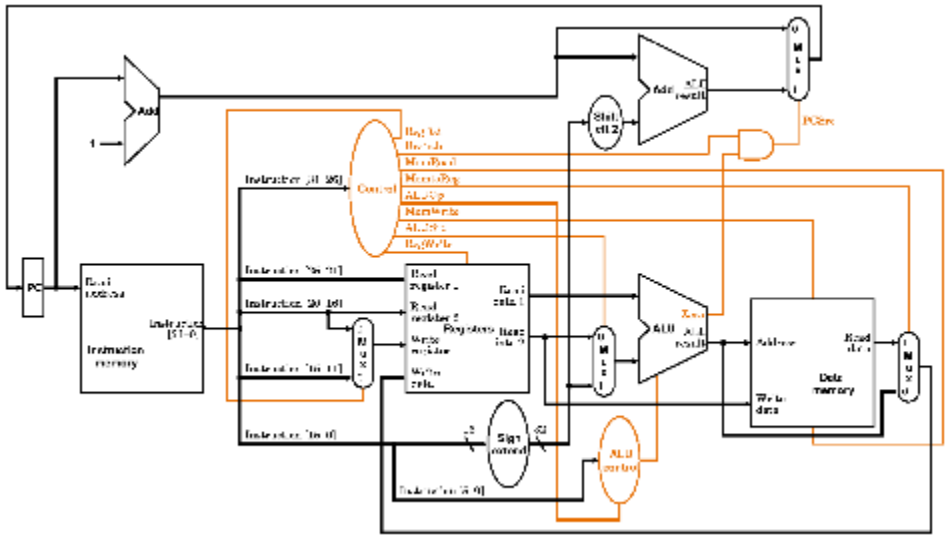
Mais detalhado

Computador básico: O Processador

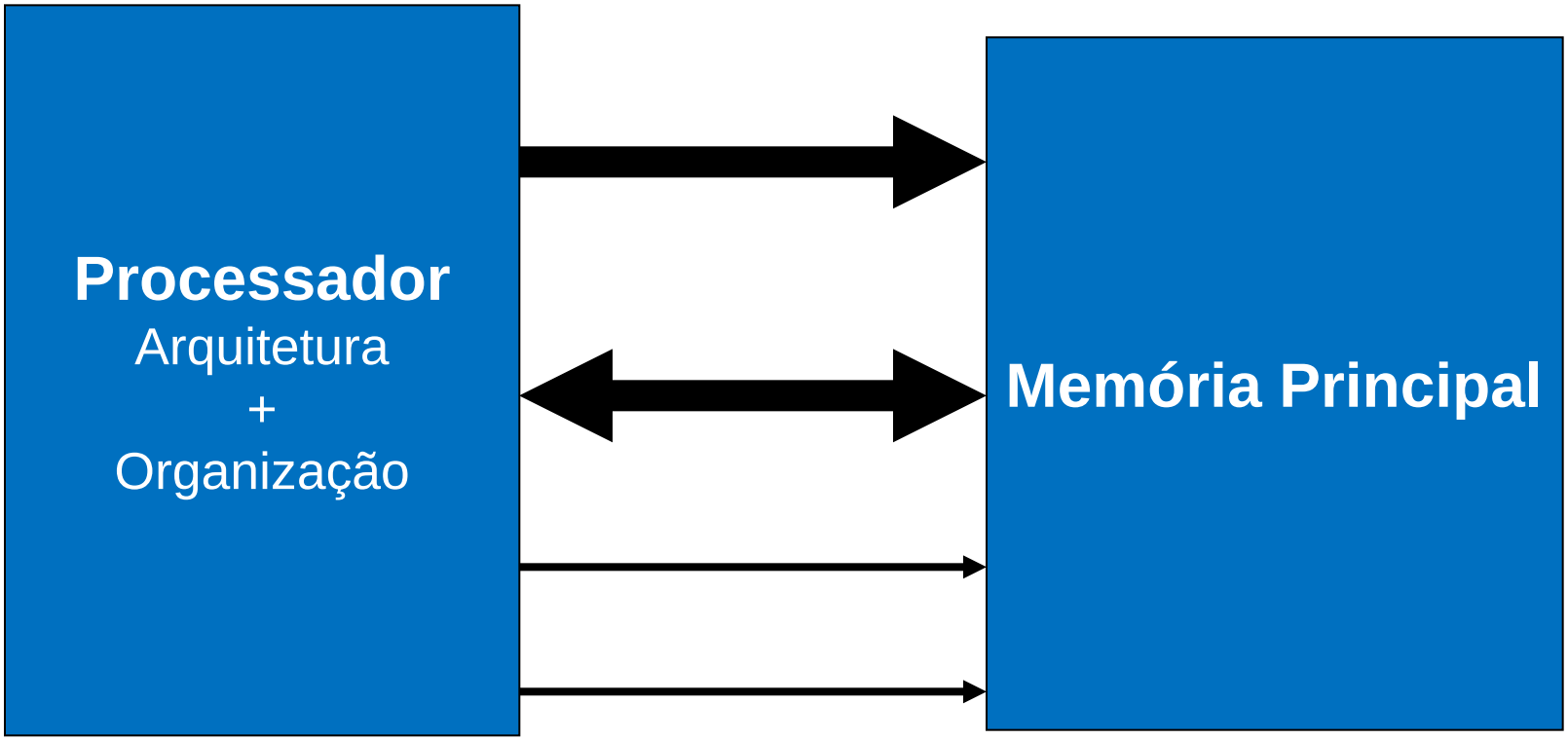
- **A arquitetura refere-se a atributos que são visíveis ao programador**
  Ex. Programação em linguagem de montagem usando uma IDE



- **A organização refere-se a atributos que não são visíveis ao programador**
  Ex. Projeto do processador usando circuitos digitais



Computador básico: O Processador



Atributos arquiteturais

Atributo	Definição
Tamanho da palavra de dados	Número de bits do dado manipulado pelo processador

Atributos arquiteturais

Atributo	Definição
Tamanho da palavra de dados	Número de bits do dado manipulado pelo processador
Tipos de dados	Tipos de dados manipulados pelo processador

Atributos arquiteturais

Atributo	Definição
Tamanho da palavra de dados	Número de bits do dado manipulado pelo processador
Tipos de dados	Tipos de dados manipulados pelo processador
Tamanho da palavra de instrução	Número de bits para representar uma instrução

Atributos arquiteturais

Atributo	Definição
Tamanho da palavra de dados	Número de bits do dado manipulado pelo processador
Tipos de dados	Tipos de dados manipulados pelo processador
Tamanho da palavra de instrução	Número de bits para representar uma instrução
Formatos de instrução	Estrutura utilizada para organização das instruções - Quantidade de formatos - Largura do campo do código da operação (OpCode) - Número de operandos - Largura dos operandos

Atributos arquiteturais

Atributo	Definição
Tamanho da palavra de dados	Número de bits do dado manipulado pelo processador
Tipos de dados	Tipos de dados manipulados pelo processador
Tamanho da palavra de instrução	Número de bits para representar uma instrução
Formatos de instrução	Estrutura utilizada para organização das instruções - Quantidade de formatos - Largura do campo do código da operação (OpCode) - Número de operandos - Largura dos operandos
Modos de endereçamento	Métodos de acesso aos dados processados pelas instruções

Atributos arquiteturais

Atributo	Definição
Tamanho da palavra de dados	Número de bits do dado manipulado pelo processador
Tipos de dados	Tipos de dados manipulados pelo processador
Tamanho da palavra de instrução	Número de bits para representar uma instrução
Formatos de instrução	Estrutura utilizada para organização das instruções - Quantidade de formatos - Largura do campo do código da operação (OpCode) - Número de operandos - Largura dos operandos
Modos de endereçamento	Métodos de acesso aos dados processados pelas instruções
Registradores	Unidades de armazenamento internas da CPU

Atributos arquiteturais

Atributo	Definição
Tamanho da palavra de dados	Número de bits do dado manipulado pelo processador
Tipos de dados	Tipos de dados manipulados pelo processador
Tamanho da palavra de instrução	Número de bits para representar uma instrução
Formatos de instrução	Estrutura utilizada para organização das instruções - Quantidade de formatos - Largura do campo do código da operação (OpCode) - Número de operandos - Largura dos operandos
Modos de endereçamento	Métodos de acesso aos dados processados pelas instruções
Registradores	Unidades de armazenamento internas da CPU
Memória	Espaços de endereçamento de memória

Atributos arquiteturais

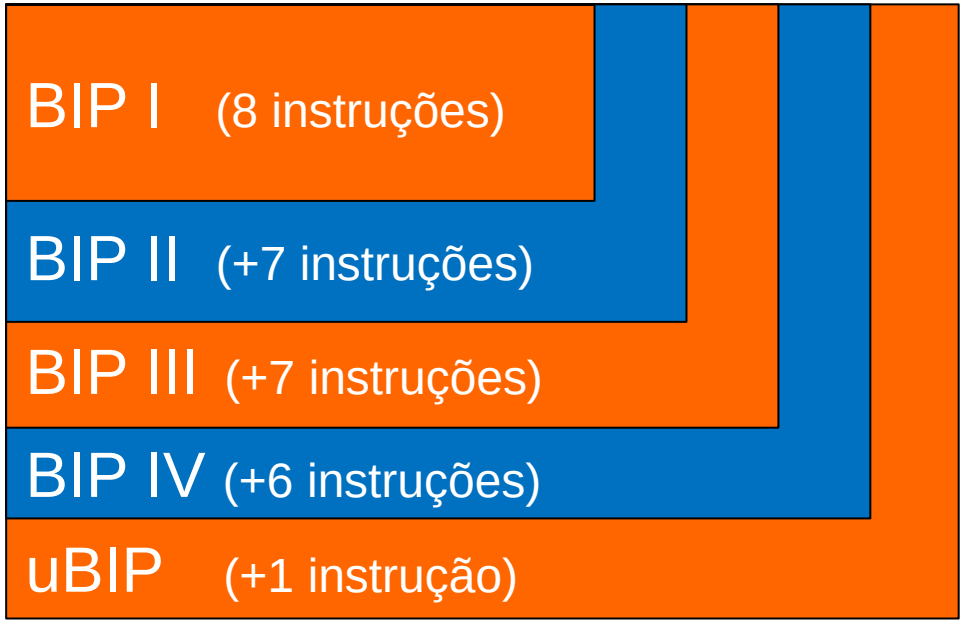
Atributo	Definição
Tamanho da palavra de dados	Número de bits do dado manipulado pelo processador
Tipos de dados	Tipos de dados manipulados pelo processador
Tamanho da palavra de instrução	Número de bits para representar uma instrução
Formatos de instrução	Estrutura utilizada para organização das instruções - Quantidade de formatos - Largura do campo do código da operação (OpCode) - Número de operandos - Largura dos operandos
Modos de endereçamento	Métodos de acesso aos dados processados pelas instruções
Registradores	Unidades de armazenamento internas da CPU
Memória	Espaços de endereçamento de memória
Conjunto de instruções	Vocabulário de instruções e códigos das operações das instruções

O Processador BIP

- BIP (Basic Instruction-set Processor)
- Arquitetura de 16 bits
- Produto de um projeto interdisciplinar conduzido pelos laboratórios LEDS e LITE desde 2006
- Diversos projetos de pesquisa desenvolvidos
- 8 TTCs e uma dissertação
- Mais de 15 artigos publicados
- Usado no CI Brasil do MCTI para treinamento de projetistas de circuitos integrados

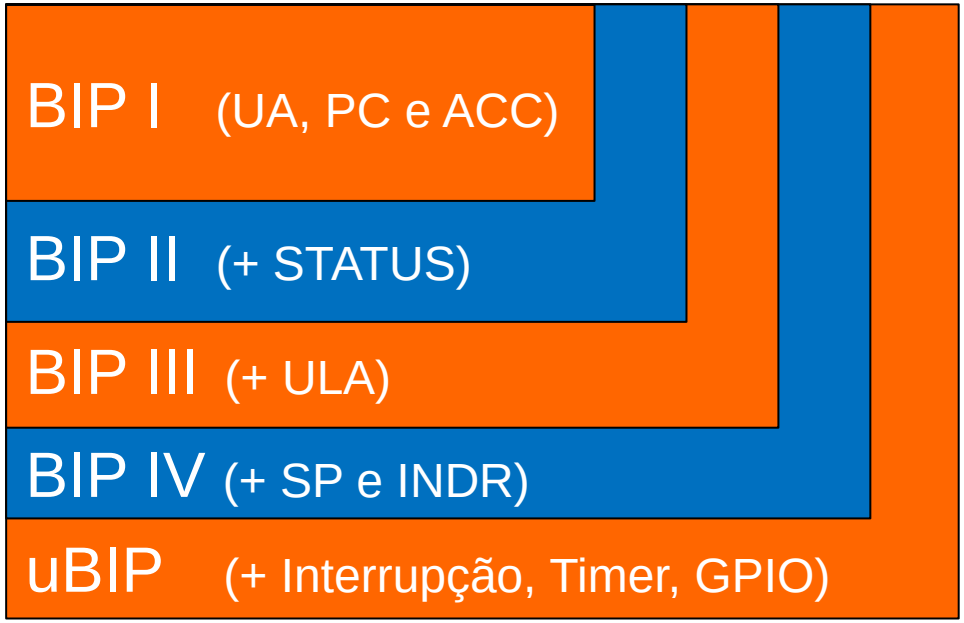
Versões

- **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
- **BIP III**: Acrescenta instruções de lógica bit a bit
- **BIP IV**: Acrescenta suporte a procedimentos e à manipulação de arrays
- **uBIP**: Acrescenta suporte a interrupções e periféricos



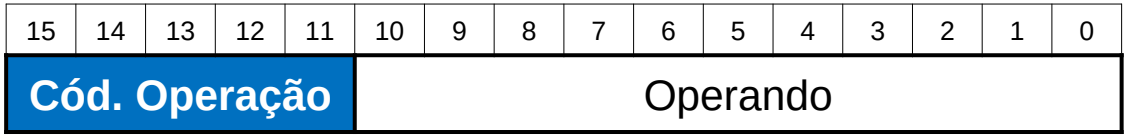
Versões

- **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
- **BIP III**: Acrescenta instruções de lógica bit a bit
- **BIP IV**: Acrescenta suporte a procedimentos e à manipulação de arrays
- **uBIP**: Acrescenta suporte a interrupções e periféricos



Suporte de hardware acrescentado em cada versão

Arquitetura do BIP I

Atributo	Definição
Tamanho da palavra de dados	16 bits
Tipos de dados	Inteiro com sinal representado com complemento de 2
Tamanho da palavra de instrução	16 bits
Formatos de instrução	Apenas um formato para todas as instruções 
Modos de endereçamento	- Imediato: o operando é uma constante de dado - Direto: o operando é um endereço de uma variável
Registradores	2 registradores: PC e ACC
Memória	2 espaços de endereçamento de memória de 2K (um para instruções e outro para dados
Conjunto de instruções	- Controle (1 instrução) - Transferência (3 instruções) - Aritmética (4 instruções)

Arquitetura do BIP I

- **Tamanho da palavra de dados: 16 bits** (65.536 valores diferentes)
- **Tipo de dado suportado: inteiro com sinal**

– 32.768

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

a

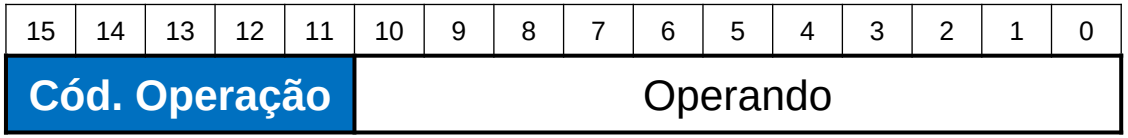
+ 32.767

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

- Número negativos são representados em complemento de 2
- O bit 15 é o bit de sinal (1: negativo, 0: positivo)

Arquitetura do BIP I

- **Tamanho da palavra de instrução: 16 bits**
- **Um formato de instrução com dois campos**



- O código de operação identifica a instrução e seus 5 bits suportam a representação de até 32 (ou seja, 2^5) instruções diferentes
- O operando (explícito) de 11 bits pode representar
  - Um endereço de memória em um espaço de 2 K (ou seja, 2^{11}) endereços diferentes
  - Uma constante de dado positiva ou negativa de 11 bits (valores no intervalo de -1024 a +1023)

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 - Accumulator)
- 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 I

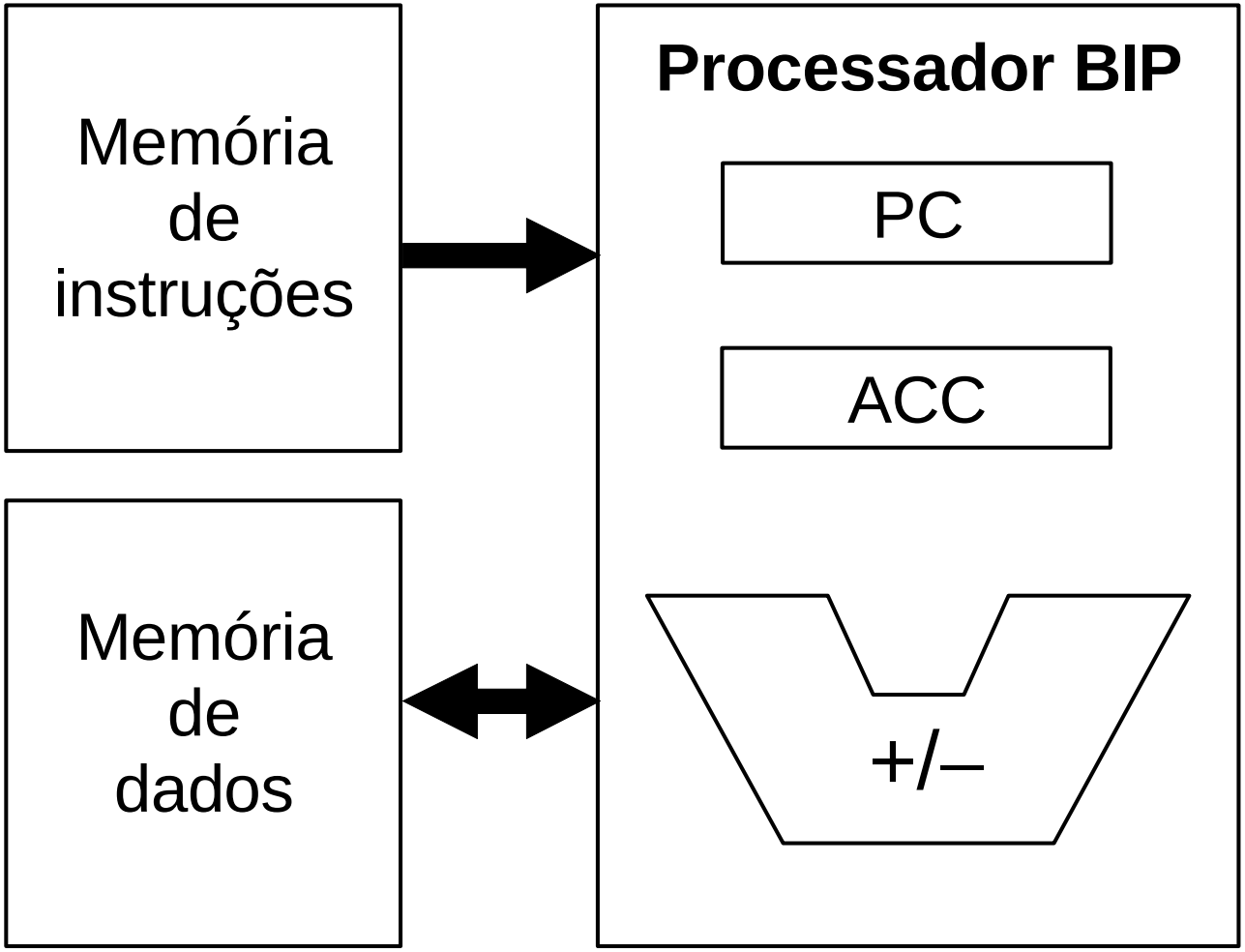
- **Registradores**
  - PC (Program Counter)
    - Registrador de propósito específico
    - Indica o endereço na memória da instrução a ser executada
    - É incrementado a cada instrução executada (PC++) para que o programa avance para a instrução seguinte
    - Possui largura de 11 bits (tamanho do endereço das memórias)
  - ACC (Accumulator)
    - Registrador de propósito geral
    - Serve para armazenamento temporário de dados no processador
    - É um operando implícito nas instruções
    - Possui largura de 16 bits (tamanho da palavra de dados)

Arquitetura do BIP I

- **Memória**
  - 2 espaços de endereçamento de memória de 2K (ou seja, 2^{11}) endereços cada
  - Memória de instruções: armazena o código do programa
  - Memória de dados: armazena as variáveis do programa

Arquitetura do BIP I

- **Modelo simplificado do processador**



- **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 I

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

Conjunto de instruções

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

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

Legenda:

← atribuição

( ) conteúdo da memória de dados

A notação **(operando) ← ACC** significa que o endereço da memória de dados apontado pelo operando receberá uma cópia do conteúdo do acumulador

- **Quais afirmativas sobre a arquitetura do BIP são verdadeiras**
  - (a) O BIP é um processador de 32 bits
  - (b) A largura da palavra de instrução do BIP é de 11 bits
  - (c) Todas as instruções do BIP utilizam um mesmo formato de representação
  - (d) Nas instruções aritméticas, o registrador ACC é um operando implícito
  - (e) Nas instruções de acesso à memória, o operando explícito da instrução aponta para a posição da memória a ser acessada

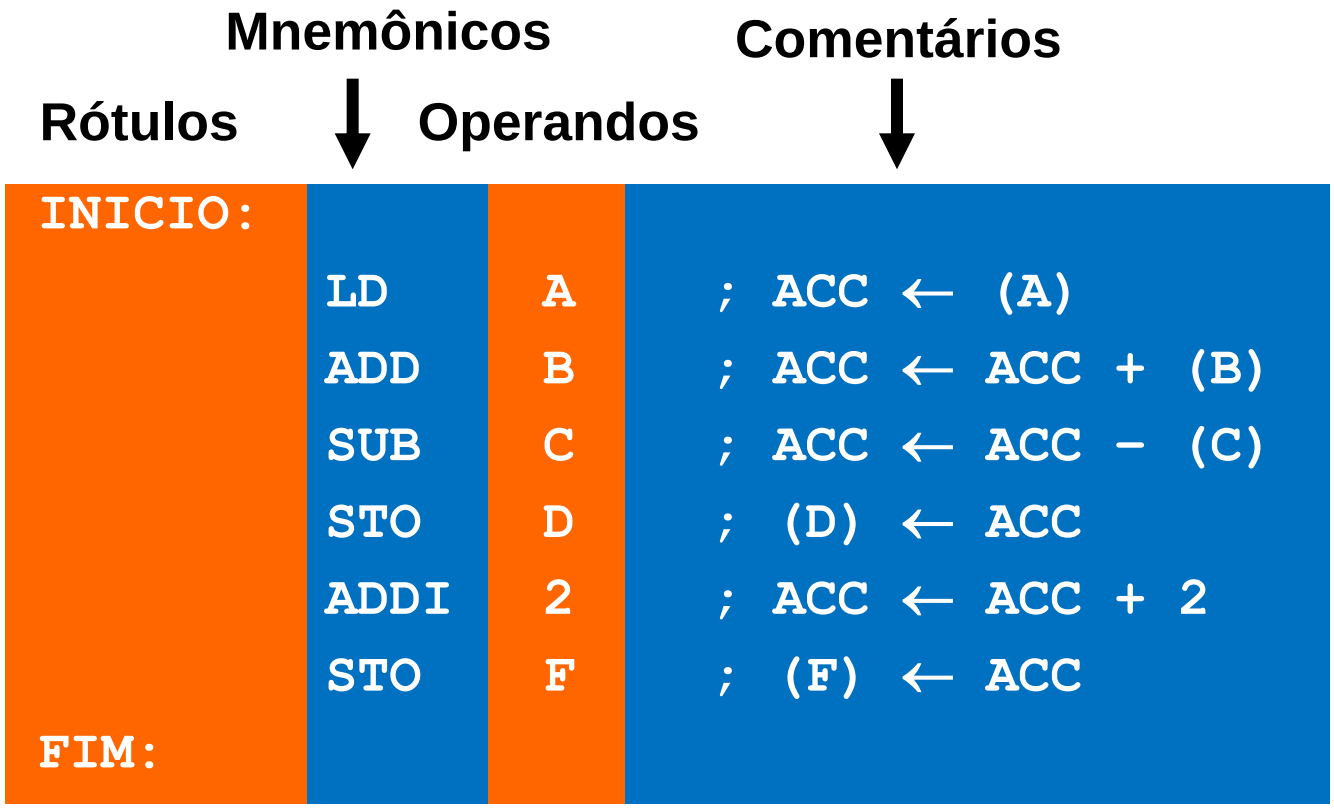
- **Quais afirmativas sobre a arquitetura do BIP são verdadeiras**
  - (a) O BIP é um processador de 32 bits
  - (b) A largura da palavra de instrução do BIP é de 11 bits
  - ✓ (c) Todas as instruções do BIP utilizam um mesmo formato de representação
  - ✓ (d) Nas instruções aritméticas, o registrador ACC é um operando implícito
  - ✓ (e) Nas instruções de acesso à memória, o operando explícito da instrução aponta para a posição da memória a ser acessada

Comentário: O BIP é um processador com palavras de dados e de instrução de 16 bits, todas as instruções são representadas utilizando o mesmo formato de instrução, o qual contem o código da operação e um operando explícito. Nas instruções de acesso à memória, esse operando é um ponteiro para uma posição da memória que será envolvida em uma transferência com o acumulador, que é o operando implícito das instruções.

- **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.

Exemplo



- **Um programa na linguagem de montagem (extensão .asm de assembly) é organizado em dois segmentos (seções):**
  - **`.data`**: no qual são declaradas as variáveis
  - **`.text`**: no qual é descrito o programa

- **Exemplo (exemplo.asm)**

```
.data
     A : 0     ; A declarado com valor inicial 0
     B : 2     ; B declarado com valor inicial 2
.text
     LDI  1    ; A atualizado com valor 1
     STO  A
     LD   B    ; B incrementado com o valor de A
     ADD  A    ; ou seja, ao final deste bloco
     STO  B    ; B = 2 + 1 = 3
     HLT  0
```

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 e nos exemplos representam endereços da memória de dados

(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

- **Qual será o conteúdo do ACC e das posições (A) e (B) da memória após a execução da cada uma das instruções abaixo?**

Mnemônicos	Operandos	Comentários	ACC	(A)	(B)
LDI	0	; ACC = 0		5	7
ADDI	1	; ACC = ACC + 1			
ADD	B	; ACC = ACC + (B)			
STO	A	; (A) = ACC			

- **Qual será o conteúdo do ACC e das posições (A) e (B) da memória após a execução da cada uma das instruções abaixo?**

Mnemônicos	Operandos	Comentários	ACC	(A)	(B)
LDI	0	; ACC = 0	**0**	5	7
ADDI	1	; ACC = ACC + 1			
ADD	B	; ACC = ACC + (B)			
STO	A	; (A) = ACC			

- **Qual será o conteúdo do ACC e das posições (A) e (B) da memória após a execução da cada uma das instruções abaixo?**

Mnemônicos	Operandos	Comentários	ACC	(A)	(B)
LDI	0	; ACC = 0	0	5	7
ADDI	1	; ACC = ACC + 1	**1**	5	7
ADD	B	; ACC = ACC + (B)			
STO	A	; (A) = ACC			

- **Qual será o conteúdo do ACC e das posições (A) e (B) da memória após a execução da cada uma das instruções abaixo?**

Mnemônicos	Operandos	Comentários	ACC	(A)	(B)
LDI	0	; ACC = 0	0	5	7
ADDI	1	; ACC = ACC + 1	1	5	7
ADD	B	; ACC = ACC + (B)	**8**	5	7
STO	A	; (A) = ACC			

- **Qual será o conteúdo do ACC e das posições (A) e (B) da memória após a execução da cada uma das instruções abaixo?**

Mnemônicos	Operandos	Comentários	ACC	(A)	(B)
LDI	0	; ACC = 0	0	5	7
ADDI	1	; ACC = ACC + 1	1	5	7
ADD	B	; ACC = ACC + (B)	8	5	7
STO	A	; (A) = ACC	8	**8**	7

- **Qual será o conteúdo do ACC e das posições (A) e (B) da memória após a execução da cada uma das instruções abaixo?**

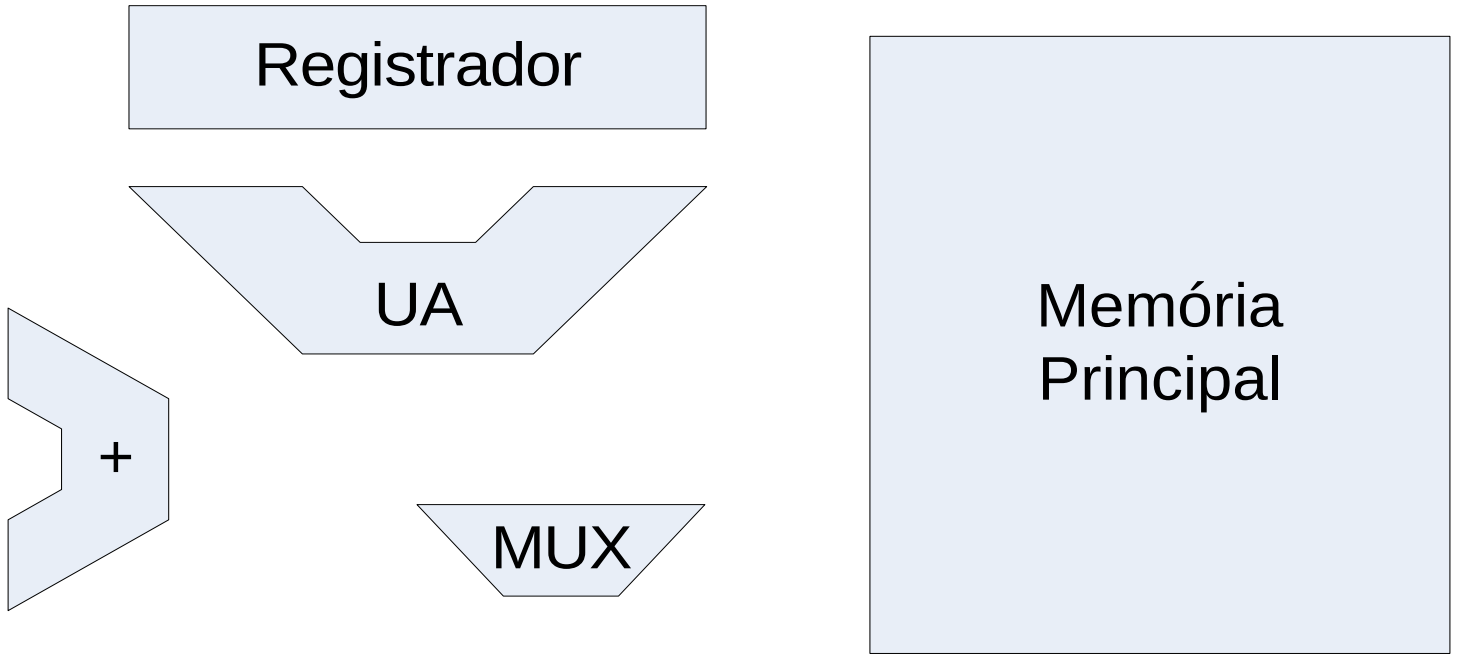
Mnemônicos	Operandos	Comentários	ACC	(A)	(B)
LDI	0	; ACC = 0	0	5	7
ADDI	1	; ACC = ACC + 1	1		
ADD	B	; ACC = ACC + (B)	8		
STO	A	; (A) = ACC		8	

Lista de Exercícios

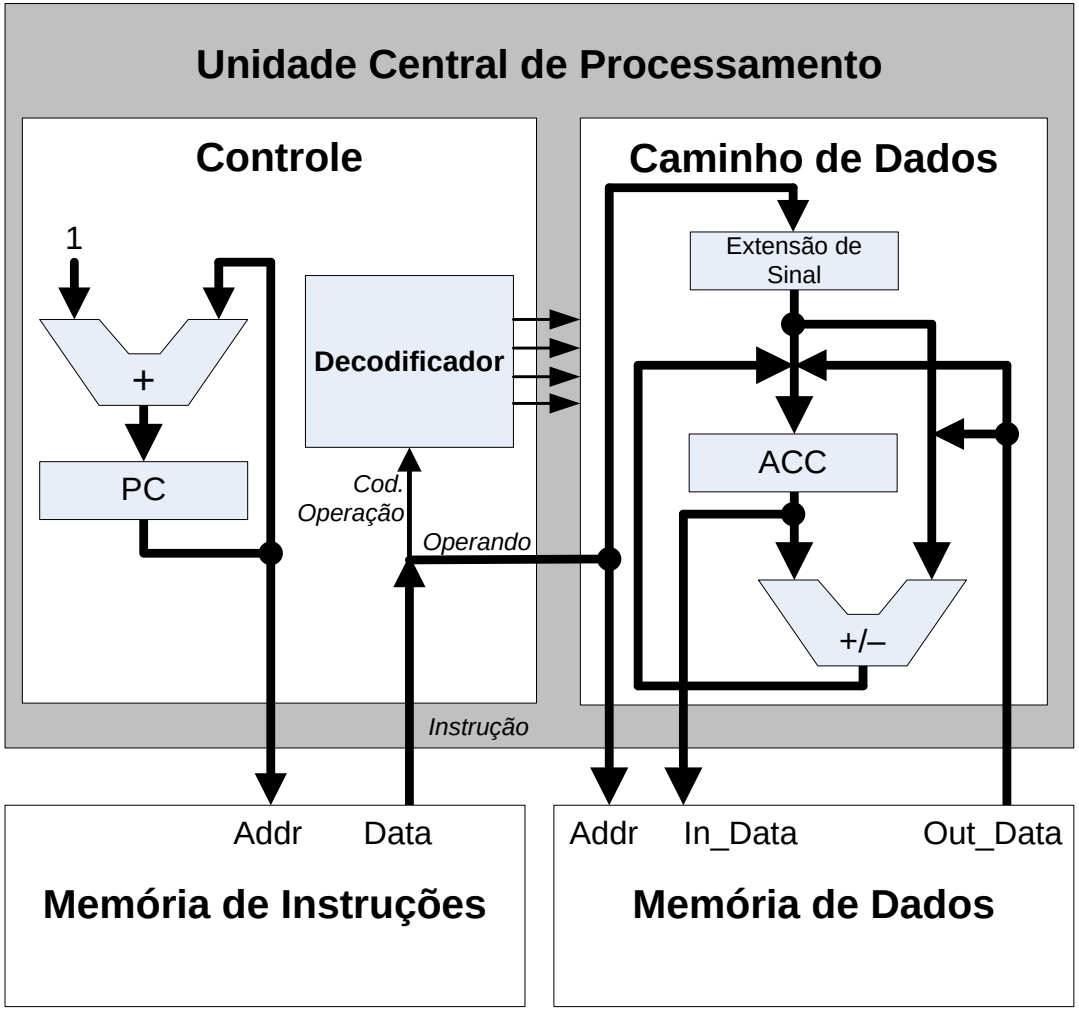
- **Ver arquivo: Listas de Exercícios – Programação do BIP I**

Organização do BIP I

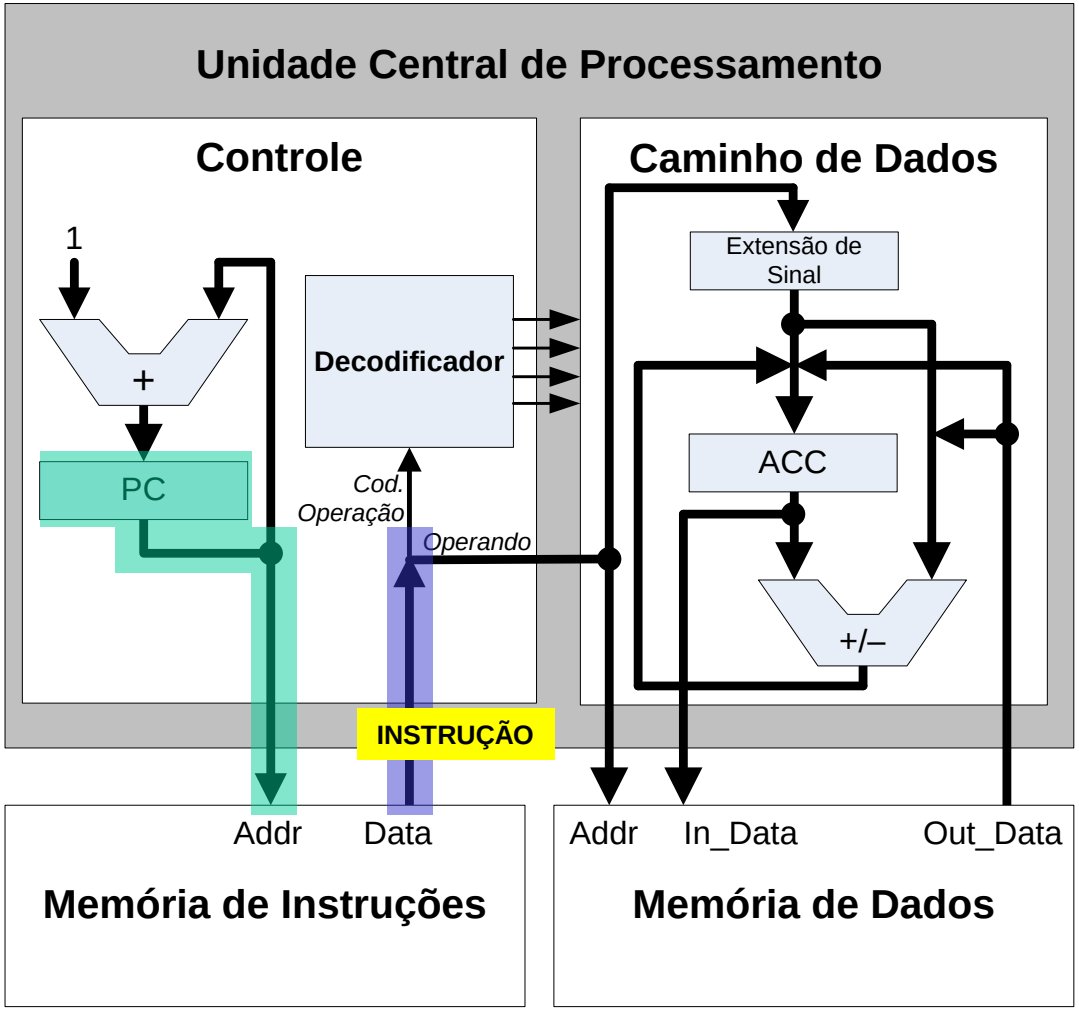
- A organização é a implementação da arquitetura
- Representada por um diagrama de blocos constituído por unidades funcionais interligadas para linhas e barras que representam fios usados para transportas sinais elétricos
- Blocos



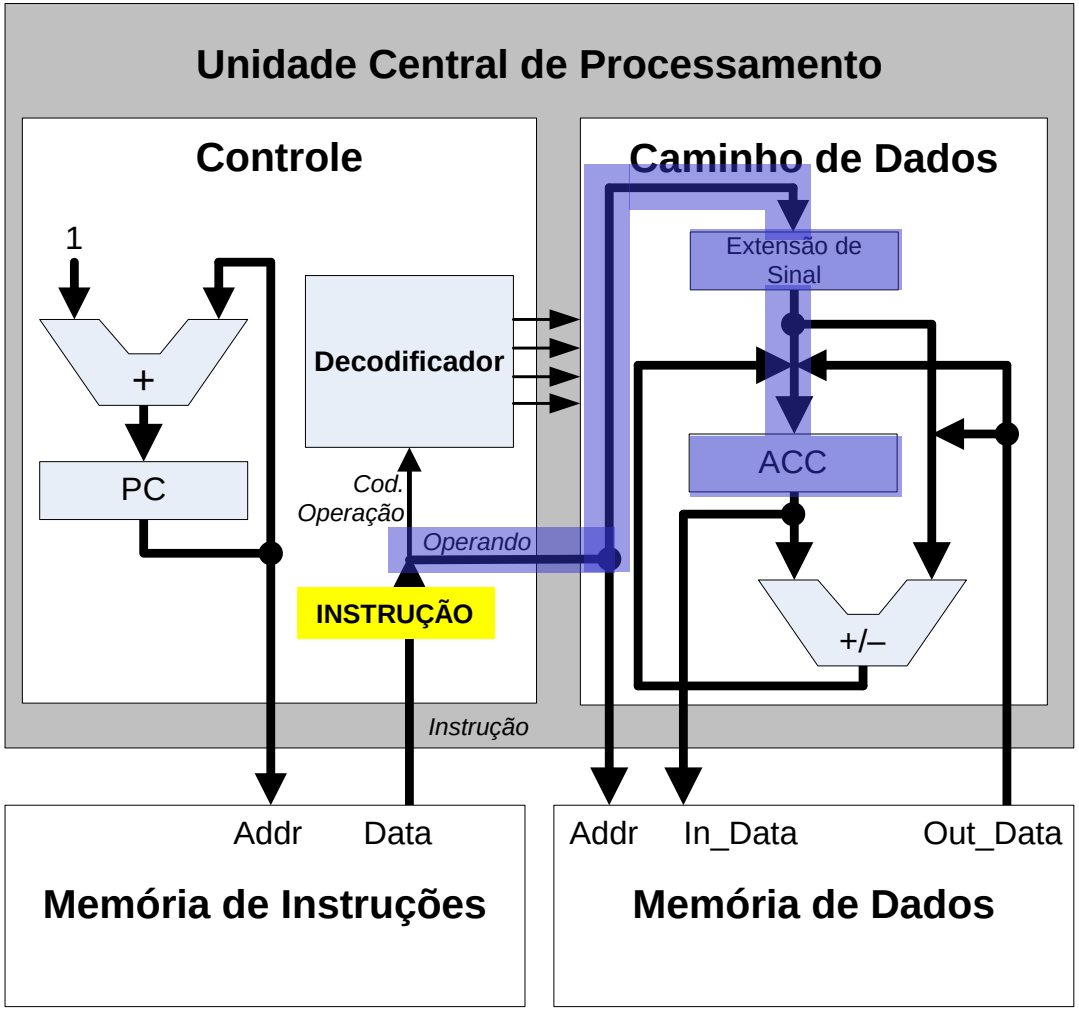
Organização do BIP I



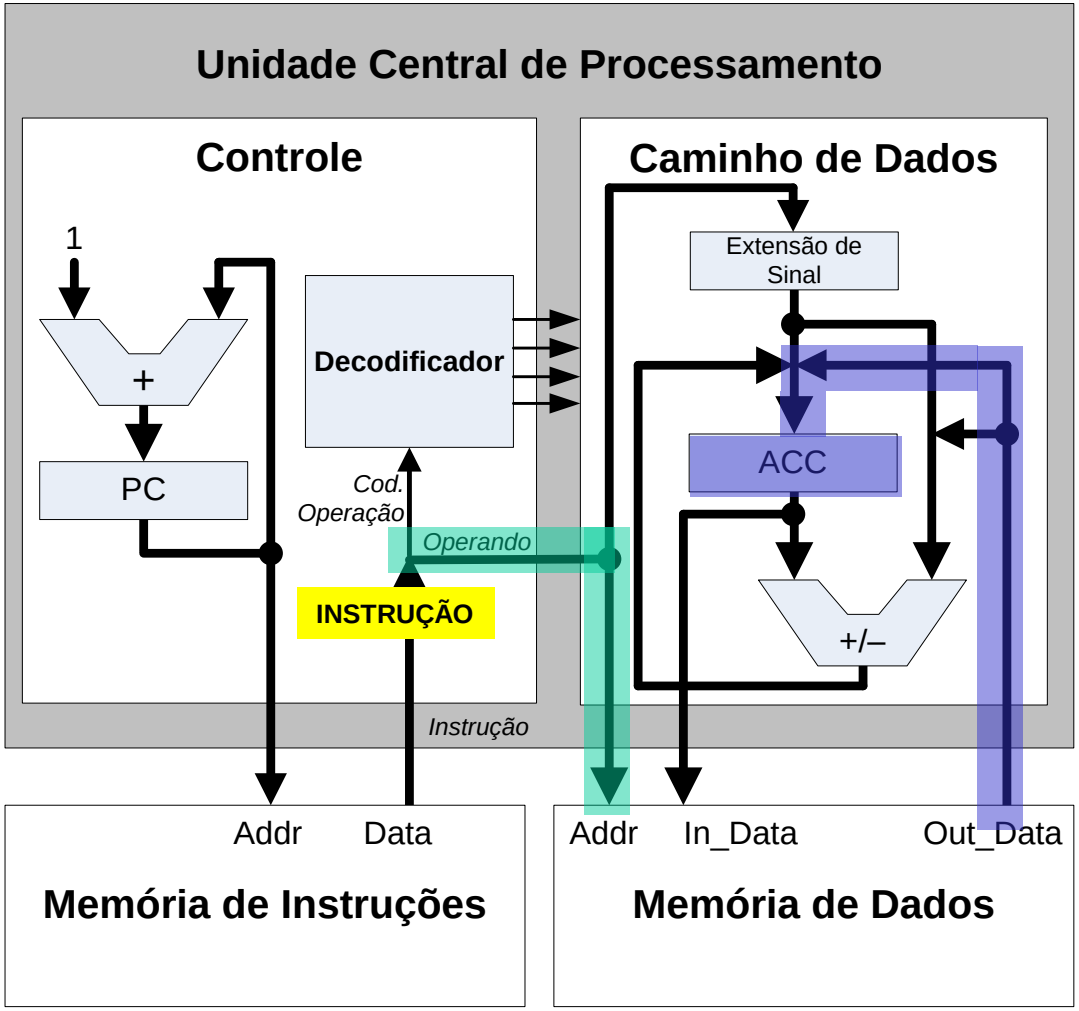
Busca de uma instrução



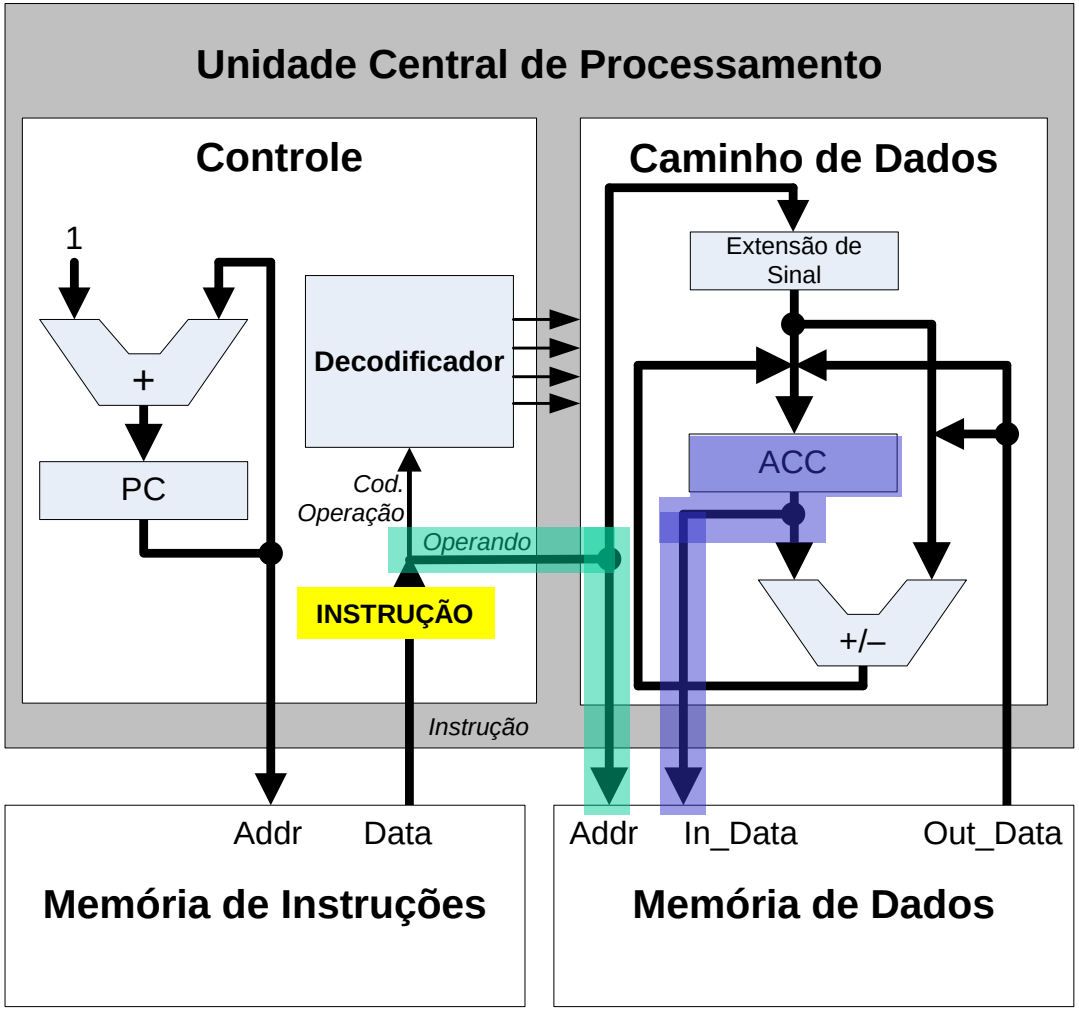
Execução da instrução LDI



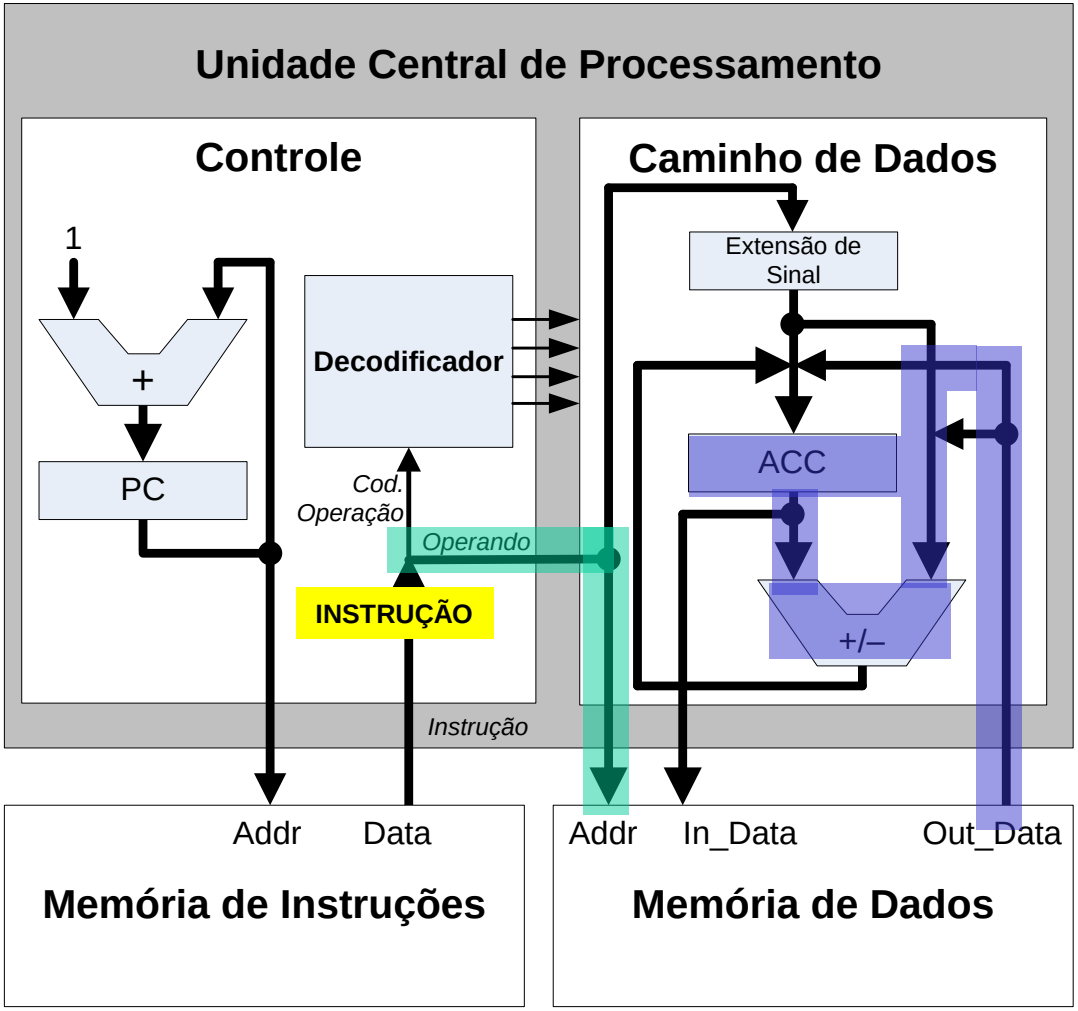
Execução da instrução LD



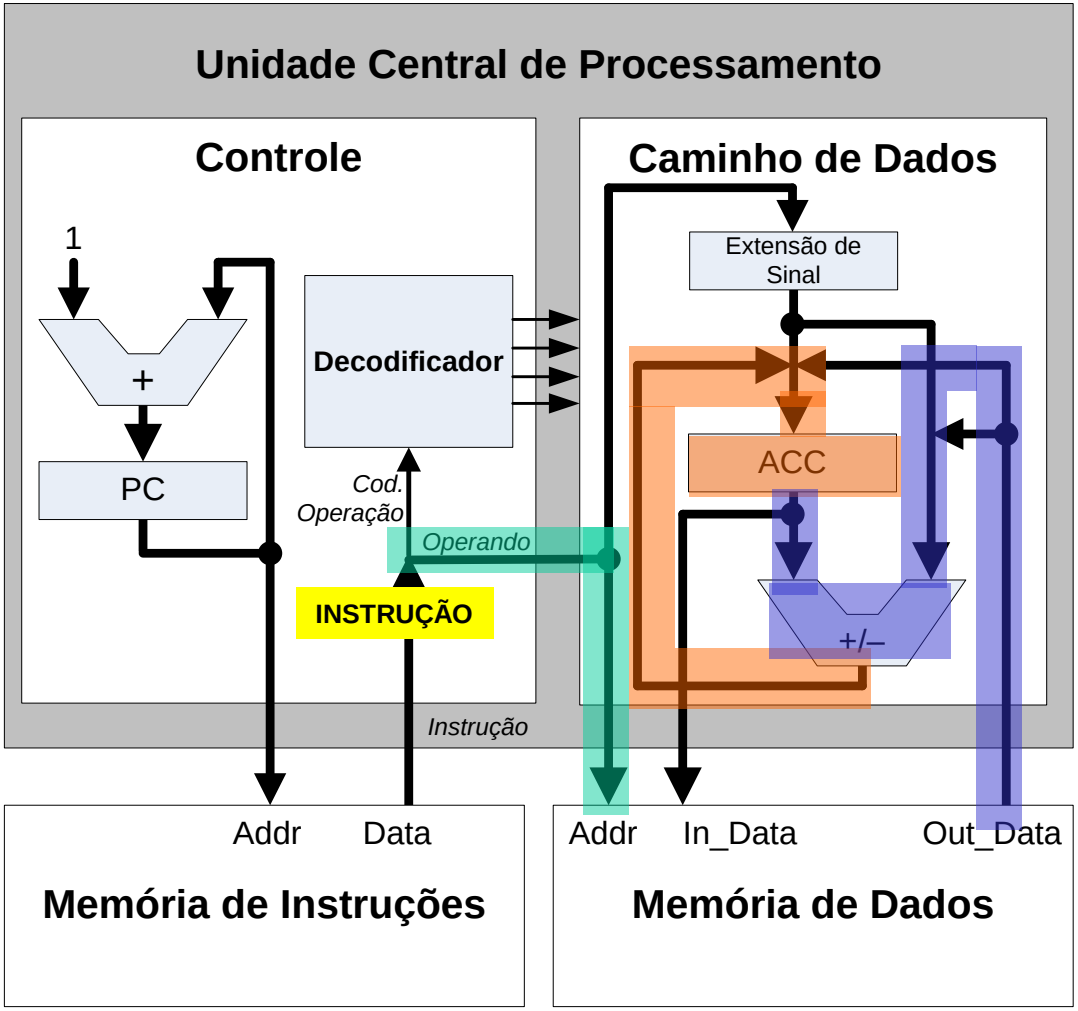
Execução da instrução STO



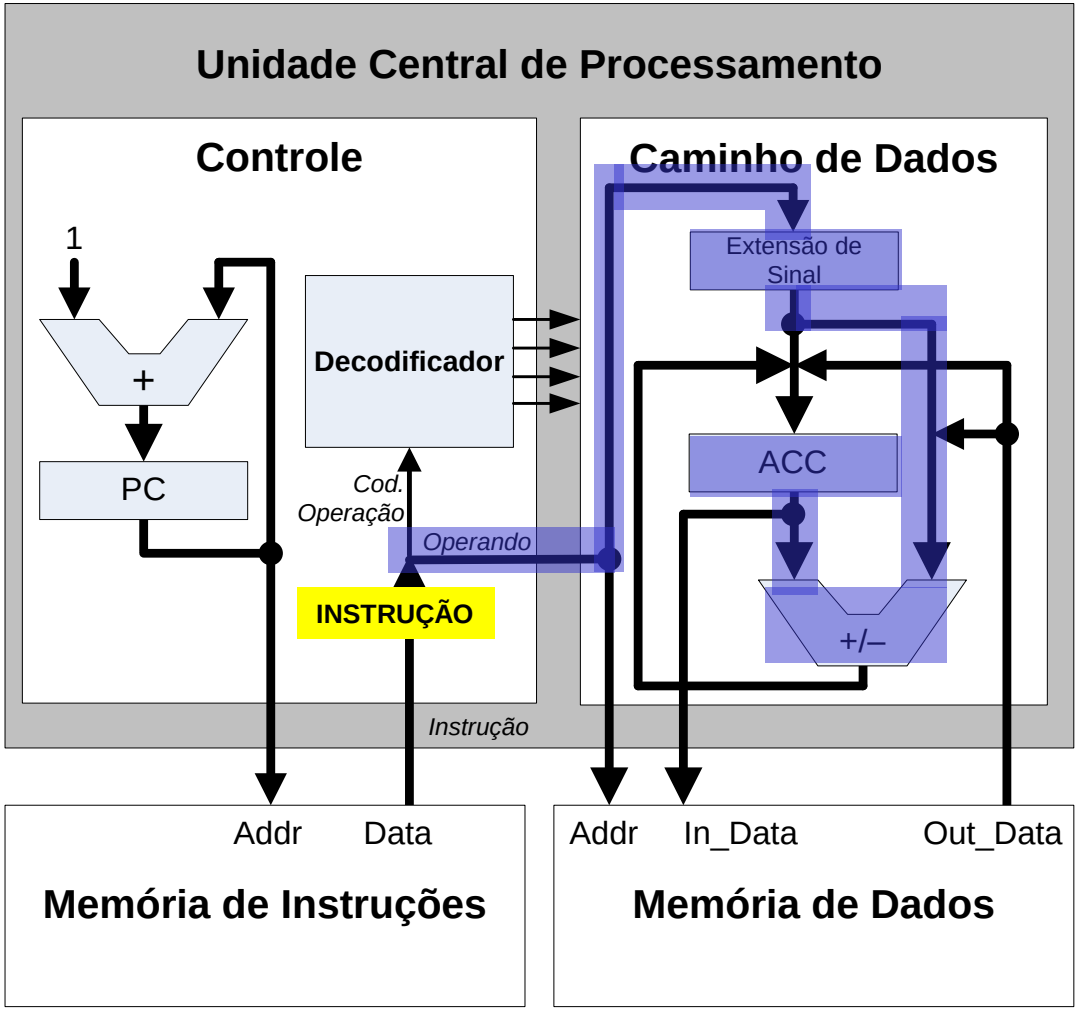
Execução da instrução ADD (execução da operação)



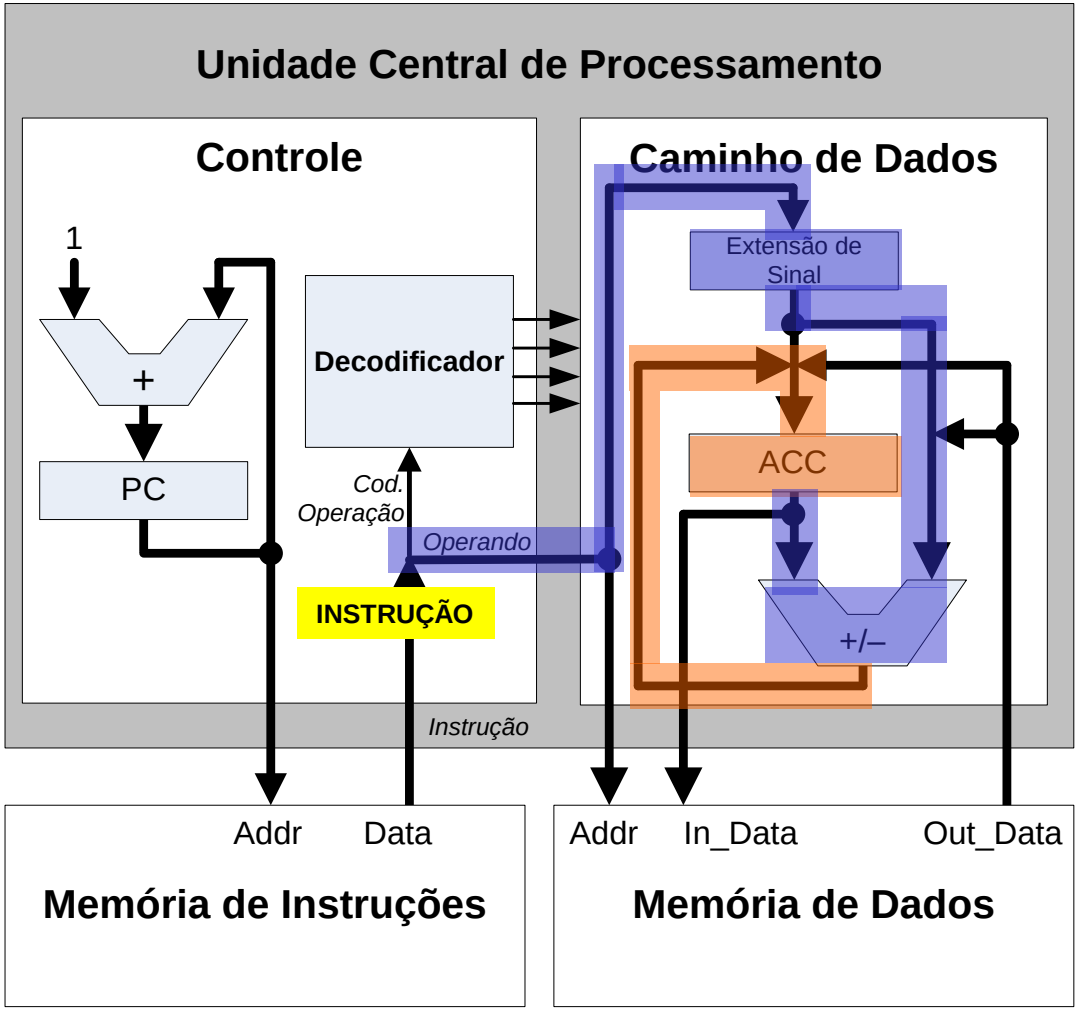
Execução da instrução ADD (escrita do resultado)



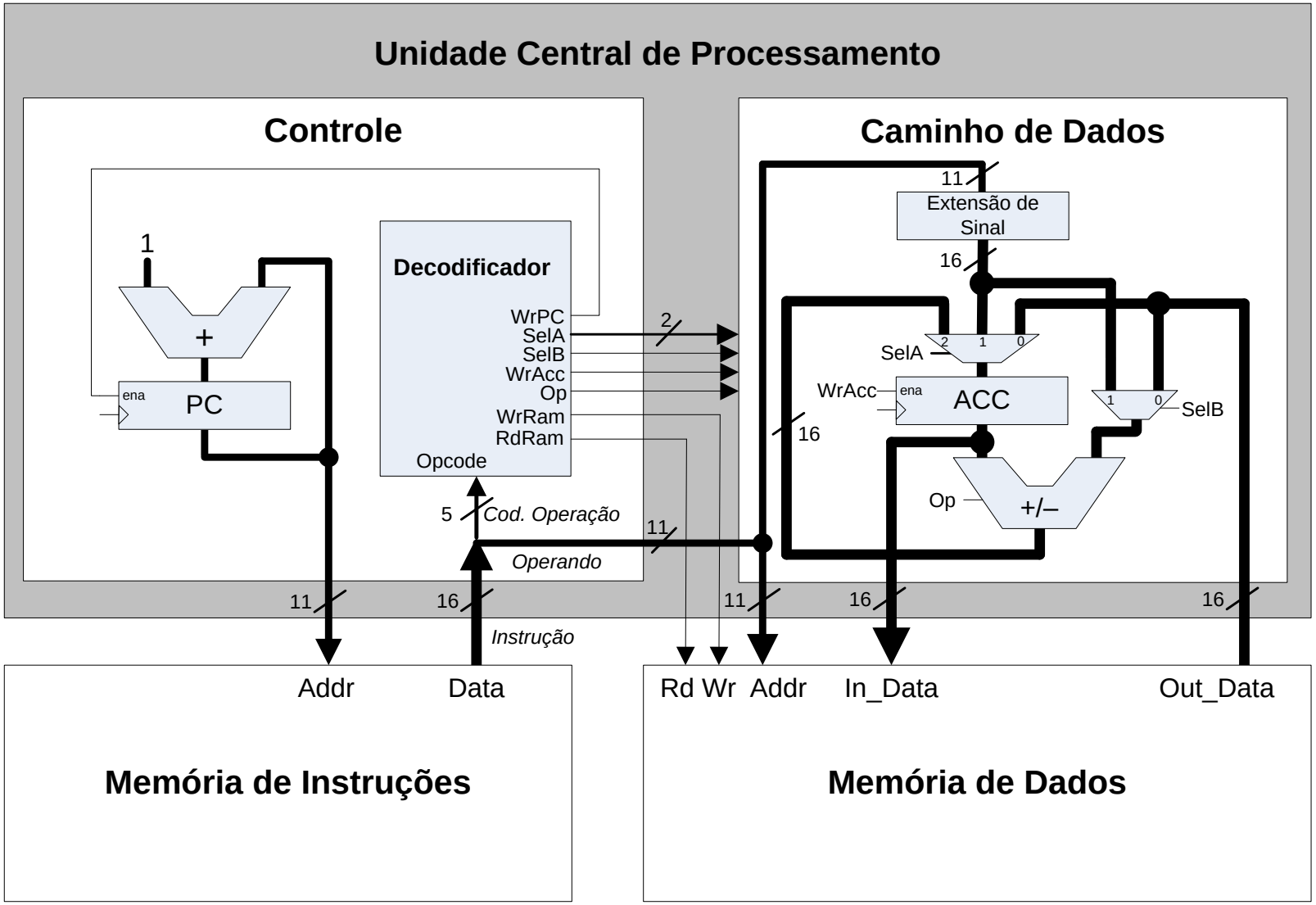
Execução da instrução ADDI (execução da operação)



Execução da instrução ADDI (escrita do resultado)



Organização detalhada



Arquitetura do 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

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 do BIP II

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

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

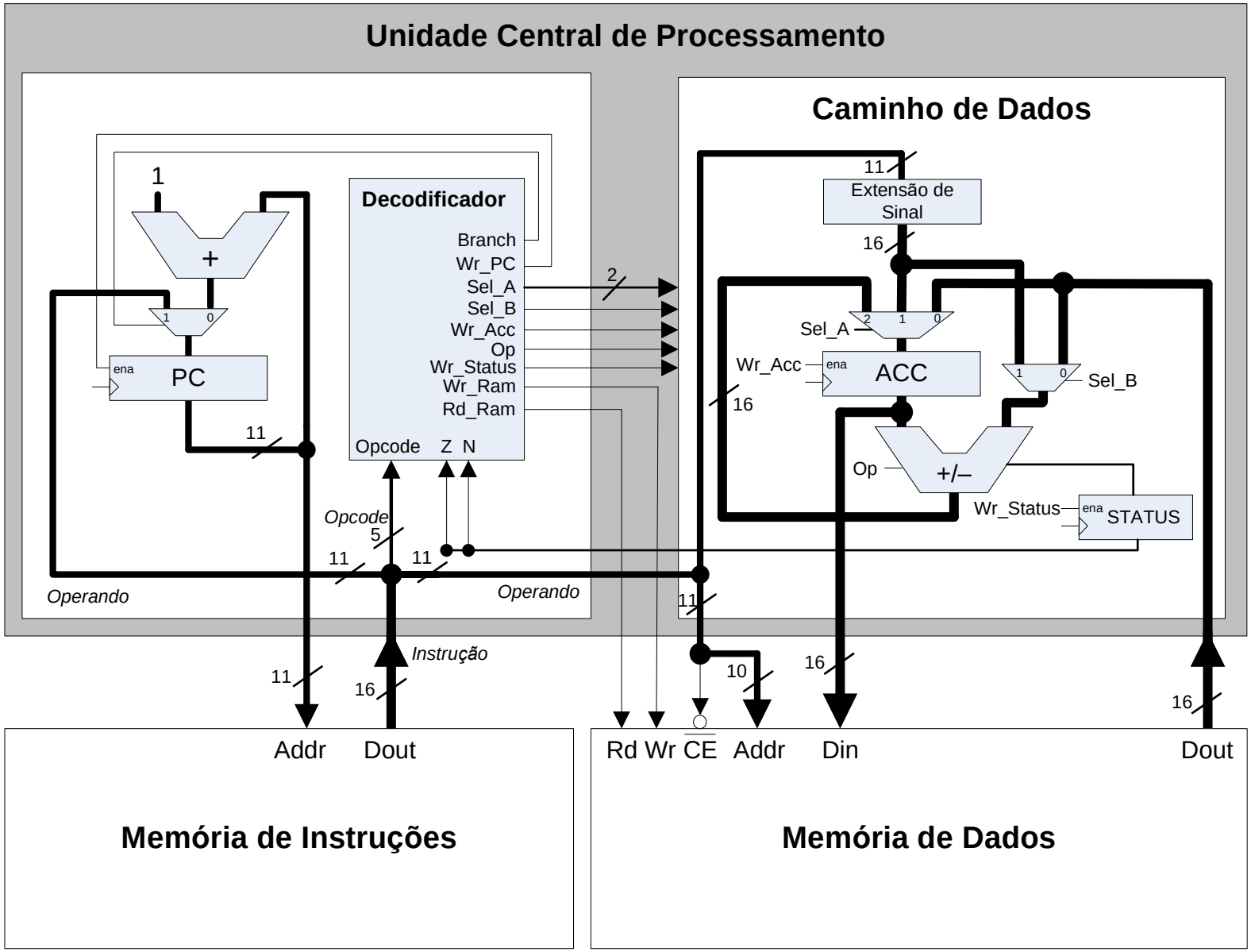
Abstração	Código em C	Código na ling. de montagem

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`

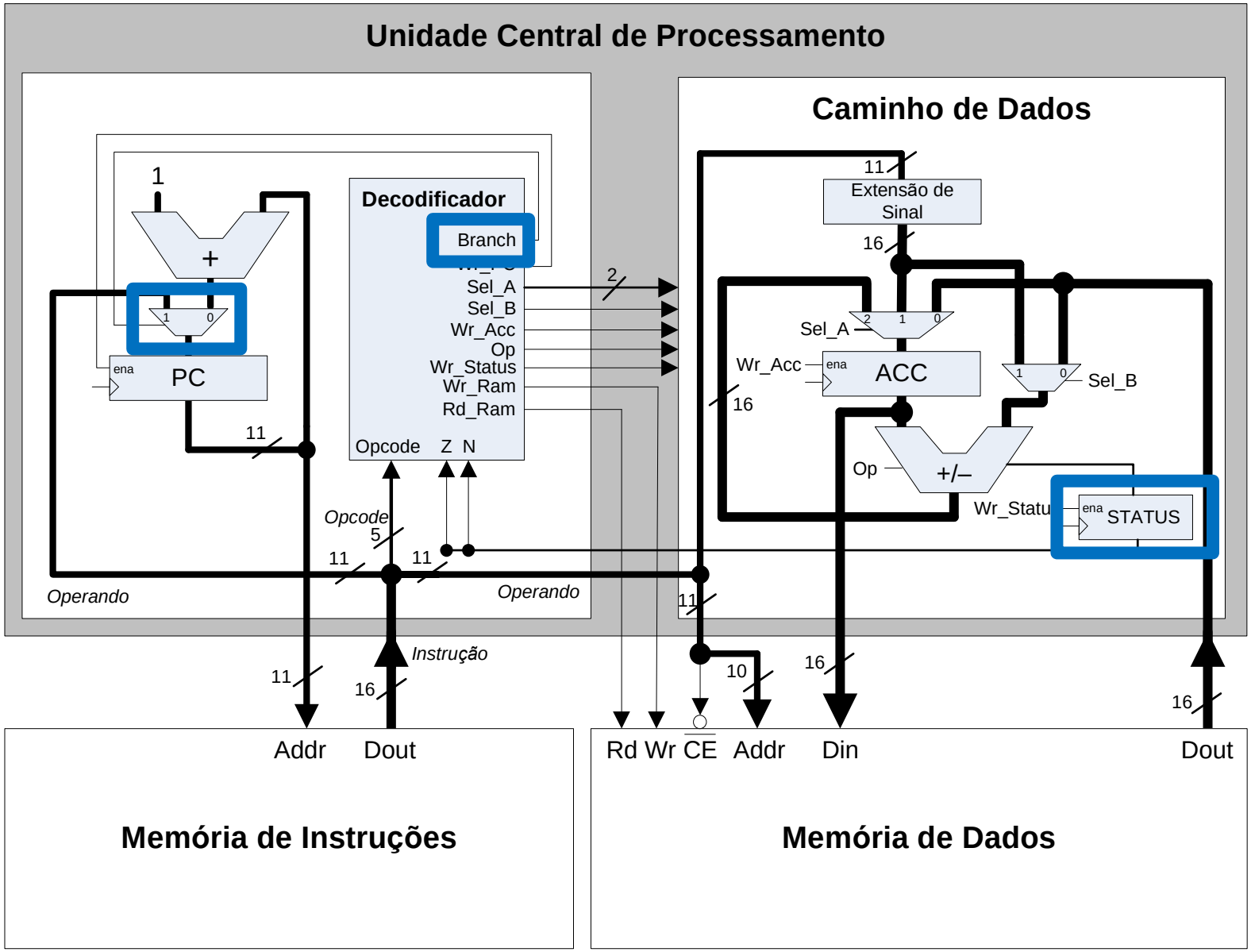
Organização do BIP II

- **Acrescenta o suporta de hardware necessários às instruções de desvio**
  - Registrador STATUS
  - Multiplexador para carga de imediato no PC
  - Lógica de decodificação

Organização do BIP II



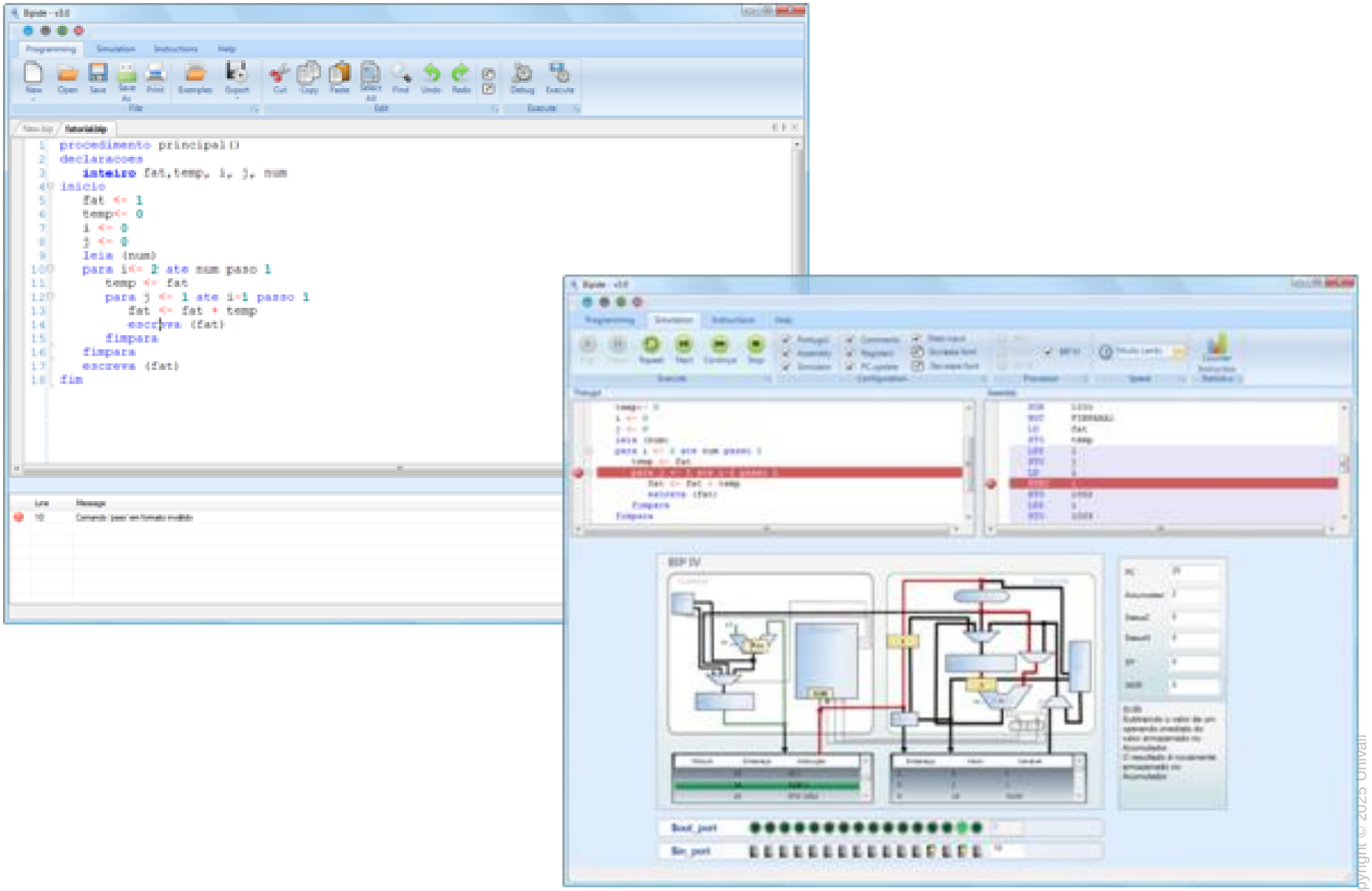
Organização do BIP II



A IDE Bipide

- **Ambiente de desenvolvimento e simulação da arquitetura e da organização dos processadores BIP**
- **Desenvolvido no TTC em Ciência da Computação de Paulo Viniccius Vieira**
  - Melhor trabalho de Conclusão de curso em Informática na Educação, SBC - Simpósio Brasileiro de Informática na Educação – SBIE 2009
- **Aprimorado em outros três TTCs**
  - Nereu Pires de Oliveira Jr. (2013)
  - Paula Mannes (2013)
  - Paulo Roberto Machado Rech (2011)

A IDE Bipide



A IDE Bipide