

Infraestrutura de Hardware

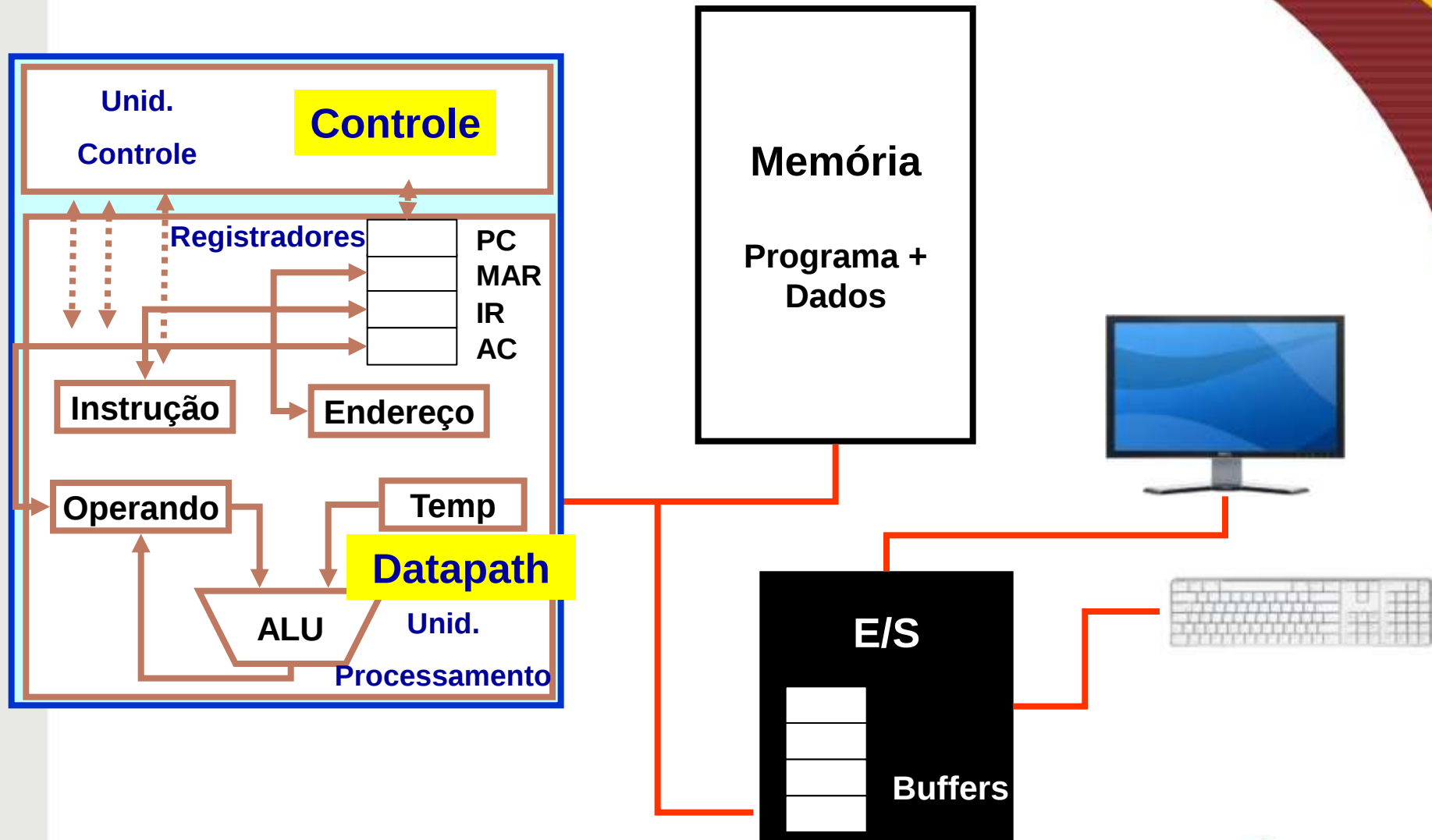
Implementação Monociclo de um Processador Simples



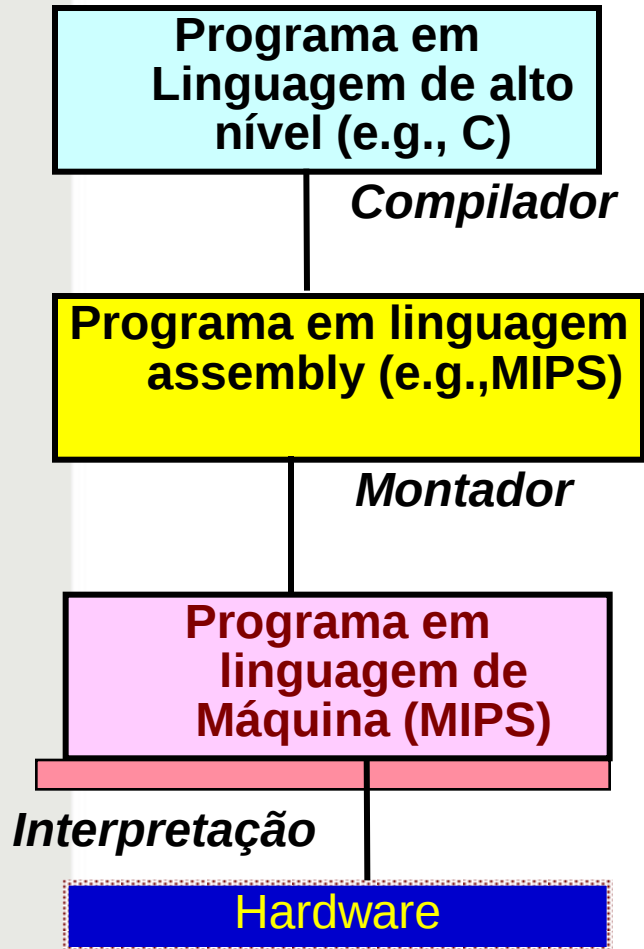
UNIVERSIDADE
FEDERAL
DE PERNAMBUCO

Cln.ufpe.br

Componentes de um Computador



Etapas para o Hardware Entender Instruções



```
temp = v[k];  
v[k] = v[k+1];  
v[k+1] = temp;
```

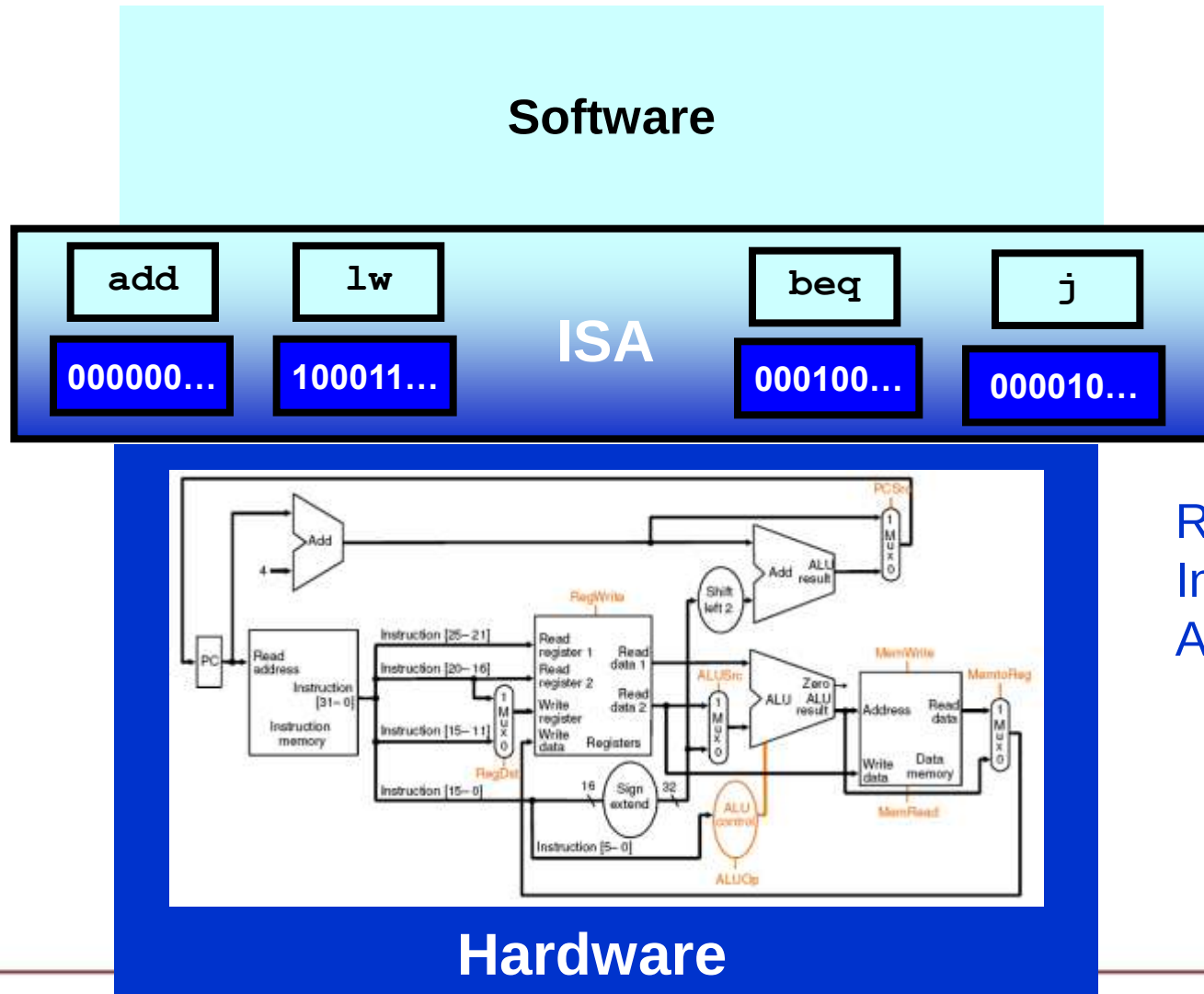
```
lw $t0, 0($2)  
lw $t1, 4($2)  
sw $t1, 0($2)  
sw $t0, 4($2)
```

```
0000 1001 1100 0110 1010 1111 0101 1000  
1010 1111 0101 1000 0000 1001 1100 0110  
1100 0110 1010 1111 0101 1000 0000 1001  
0101 1000 0000 1001 1100 0110 1010 1111
```

Perguntas que Devem ser Respondidas ao Final do Curso

- Como um programa escrito em uma linguagem de alto nível é entendido e executado pelo HW?
- Qual é a interface entre SW e HW e como o SW instrui o HW a executar o que foi planejado?
- O que determina o desempenho de um programa e como ele pode ser melhorado?
- Que técnicas um projetista de HW pode utilizar para melhorar o desempenho?

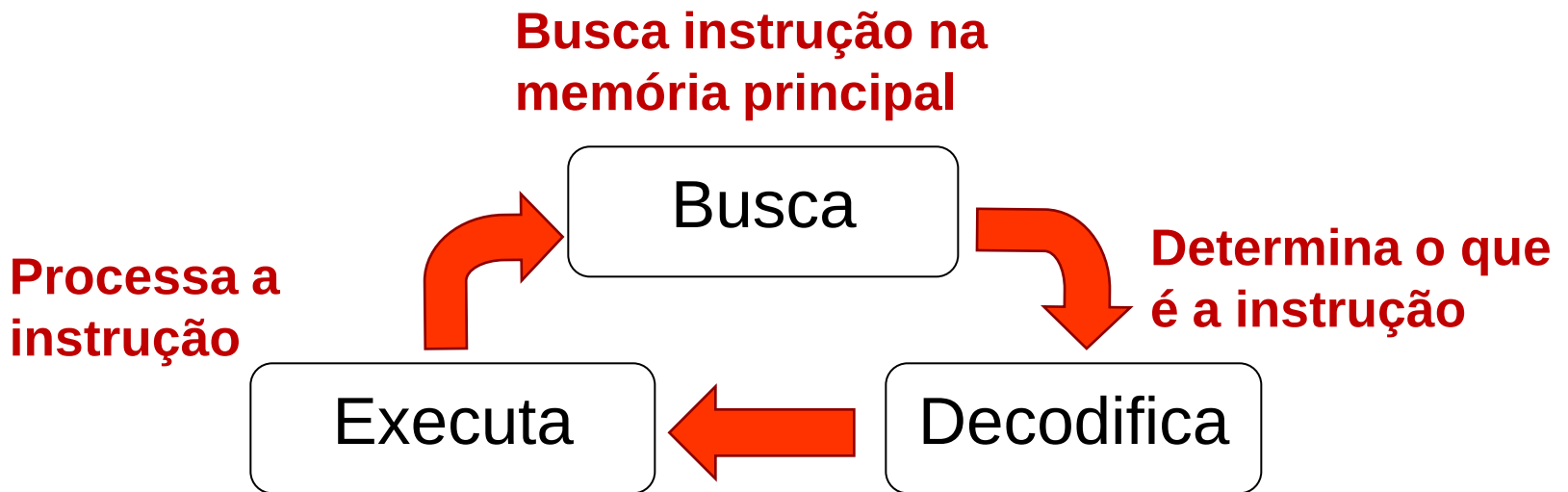
Interface HW/SW: ISA



Repertório de
Instruções da
Arquitetura

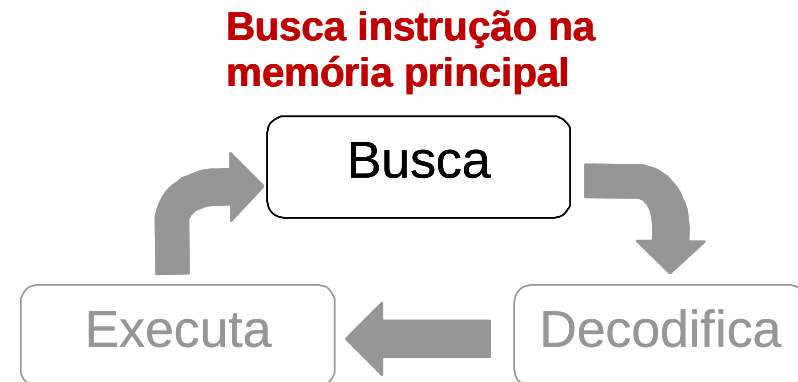
Visão Simplificada de Processamento de Instrução

- CPU faz continuamente 3 ações:



Buscando Instrução

- CPU deve saber qual é a instrução que deve ser buscada
Instrução na sequência?
Instrução especificada em um desvio?
- CPU deve calcular endereço da próxima instrução a ser executada



Decodificando Instrução

- CPU deve determinar o que é a instrução

Diferentes formatos de instrução

Como se identifica diferentes formatos?

Como se identifica diferentes instruções com mesmo formato?

Como se obtém os operandos?

op	rs	rt	rd	sa	funct	R format
op	rs	rt	immediate			I format
op	jump target					J format



Executando Instrução

- CPU deve seguir o que a instrução manda

Diferentes tipos de instruções

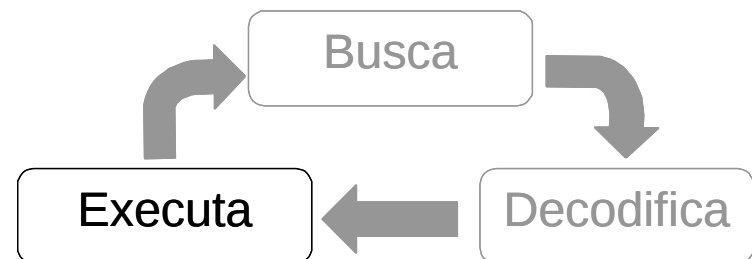
- Aritméticas/Lógicas
- Acesso à memória
- Desvios

Cálculos com dados e endereços

Acesso a diferentes componentes

- Registradores, ALU, Memória ...

Coordenação da interação entre diferentes componentes



Partes de uma CPU

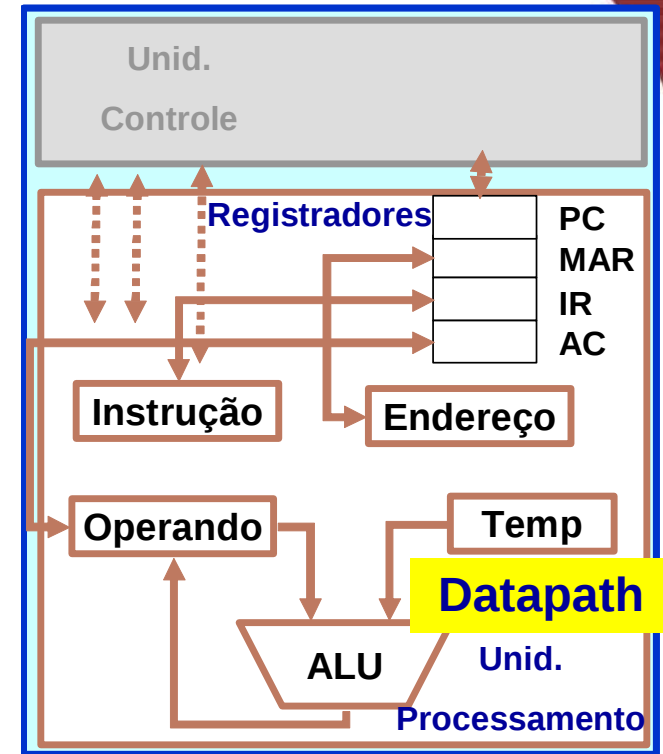
■ Datapath

Unidade de Processamento da instrução

Caminho onde os dados/endereços trafegam

Composto por unidades funcionais, que operam sobre dados/endereços, e suas interconexões

- Processamento: ALUs, Somadores, Multiplexadores ...
- Armazenamento: Registradores e Memória



Partes de uma CPU

■ Controle

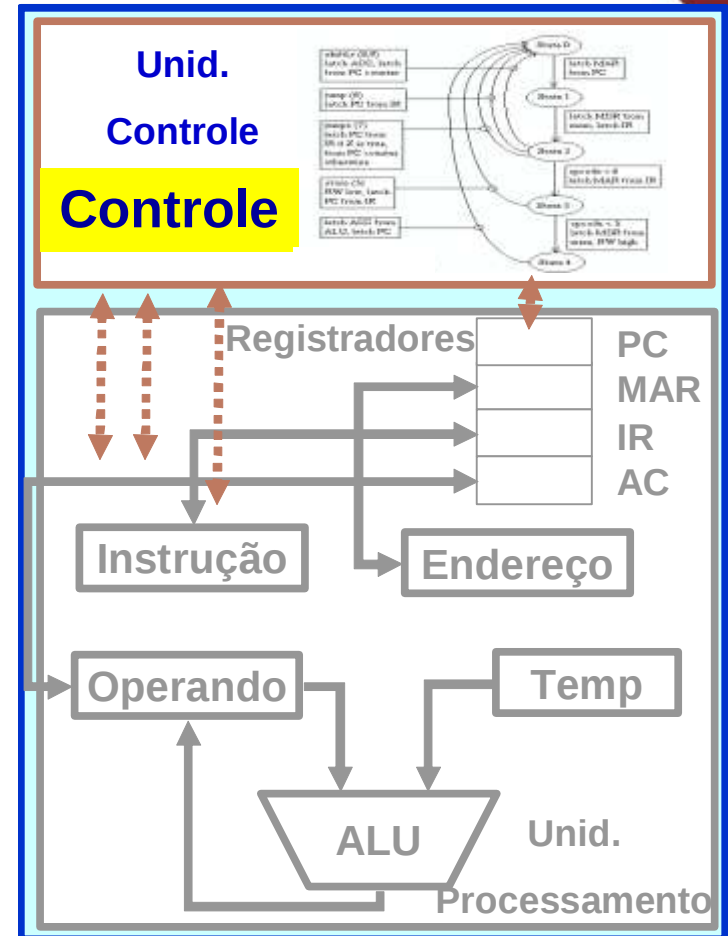
Unidade de Controle

Decodifica a instrução

Coordena as etapas de processamento da instrução

Ativa somente as unidades funcionais necessárias para execução da instrução

- Decisão baseada na instrução



Arquitetura de uma CPU

- Projeto de arquitetura de uma CPU requer a definição de:
 - Conjunto de registradores
 - Tipos de dados
 - Repertório de instruções
 - Formato de instruções

Projetando uma CPU Simples

- Iremos ver como se pode projetar e implementar uma CPU simples
- CPU com um repertório de instruções composto por um subconjunto de instruções do MIPS
 - Aritméticos, lógicos, armazenamento e desvio
- Conjunto de registradores: mesmo do MIPS
- Operandos são do tipo **inteiro**
- Toda instrução deve ser executada em **um ciclo de clock** (Implementação **Monociclo**)
 - Irreal nos dias de hoje
 - Ciclo de clock deve longo o suficiente para que a instrução mais lenta respeite esta restrição

Instruções

Repertório

Instrução	Descrição
ADD rd, rs, rt	$rd \leftarrow rs + rt$
SUB rd, rs, rt	$rd \leftarrow rs - rt$
AND rd, rs, rt	$rd \leftarrow rs \text{ and } rt$ (bit a bit)
OR rd, rs, rt	$rd \leftarrow rs \text{ or } rt$ (bit a bit)
SLT rd, rs, rt	Se $rs < rt$, $rd \leftarrow 1$, senão $rd \leftarrow 0$
LW rt, desl(rs)	Carrega palavra de mem. em registrador rt
SW rt, desl(rs)	Armazena conteúdo de registrador rt em mem.
BEQ rs, rt, end	Desvio para end, se $rs == rt$

Formato

Aritméticos/Lógicos

op	rs	rt	rd	sa	funct
----	----	----	----	----	-------

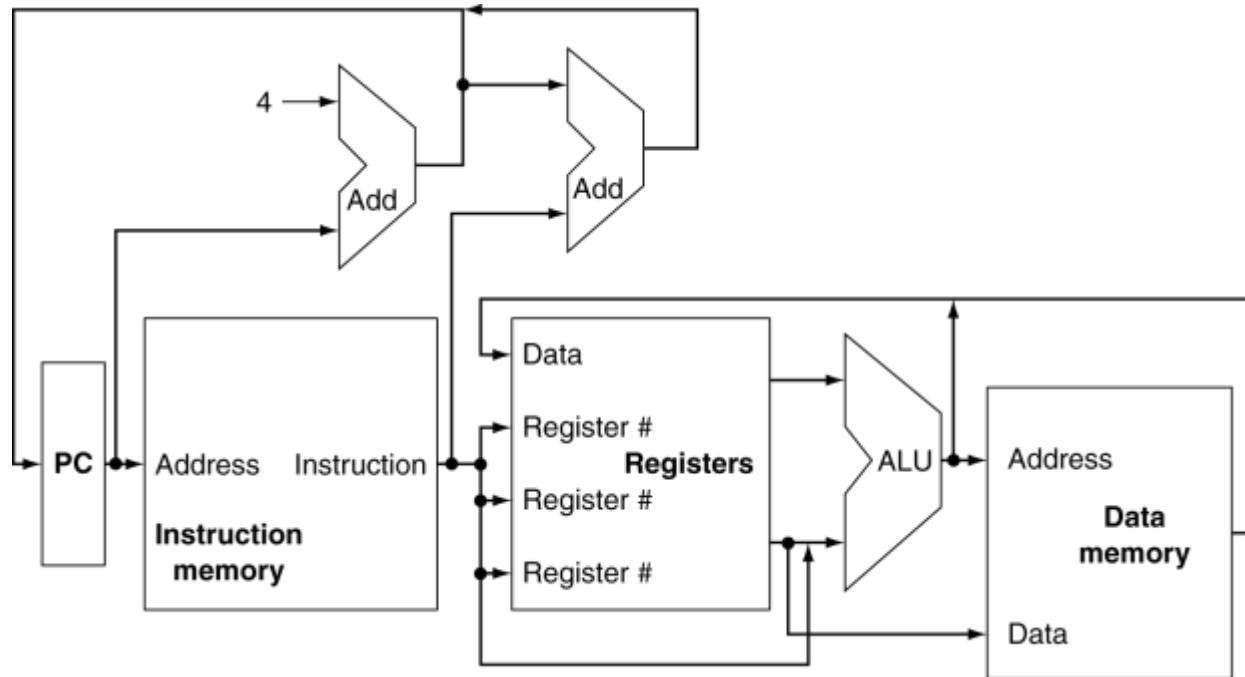
Armazenamento

op	rs	rt	deslocamento
----	----	----	--------------

Desvio

op	rs	rt	endereço
----	----	----	----------

MIPS: Visão Abstrata

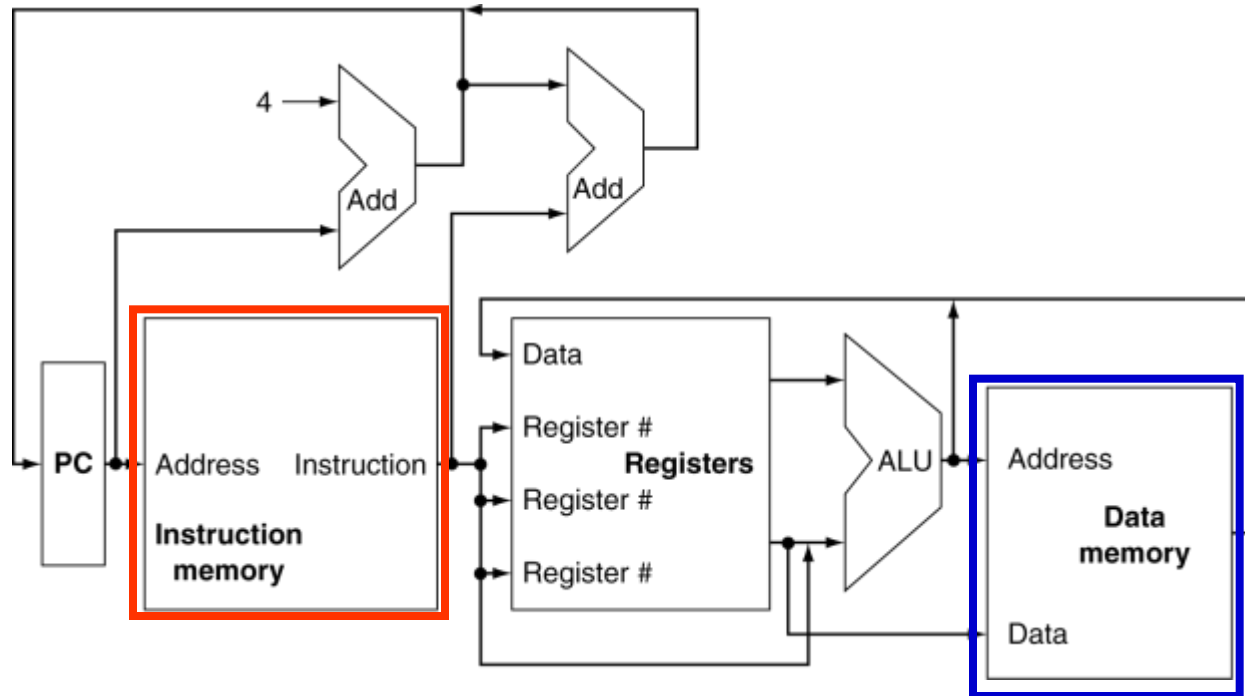


- Na CPU simples, 2 etapas sempre executadas independente da instrução

Endereço do PC é enviado a memória para buscar instrução

Ler um ou dois registradores, especificados na instrução

Por Que Duas Memórias?



- Na CPU simples, uma instrução deve ser executada em um ciclo
Nenhuma unidade funcional pode ser utilizada mais de uma vez no mesmo ciclo
Duplica-se a memória: **instruções** e **dados**

Categorias de Elementos da CPU

- A CPU é composta por 2 categorias de elementos:

Combinacional

Sequencial

- Combinacional

Operam sobre dados

Saída é uma função da entrada

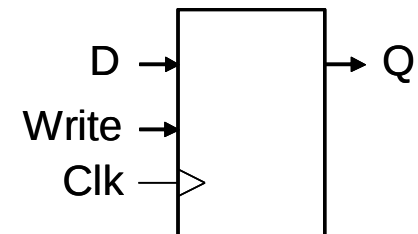
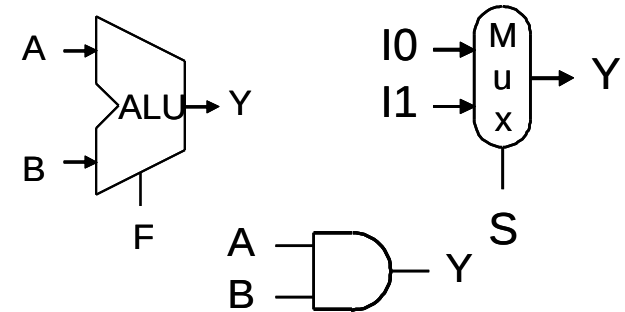
ALUs, multiplexadores...

- Sequencial

Possui um estado

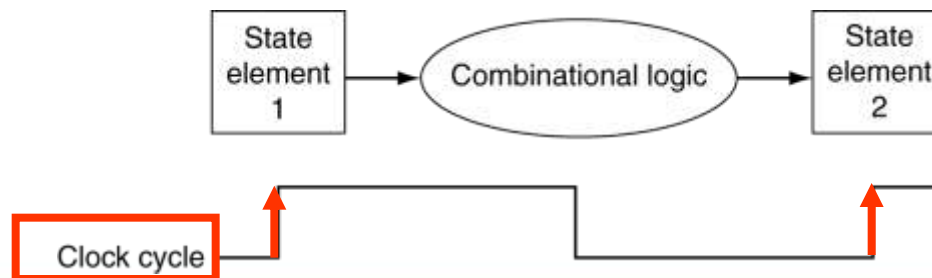
Armazena informações

Registradores, memória...

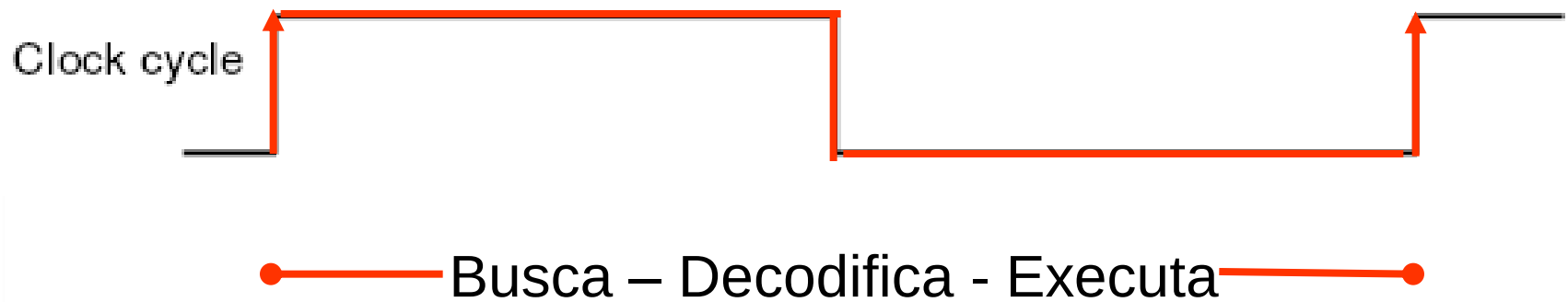
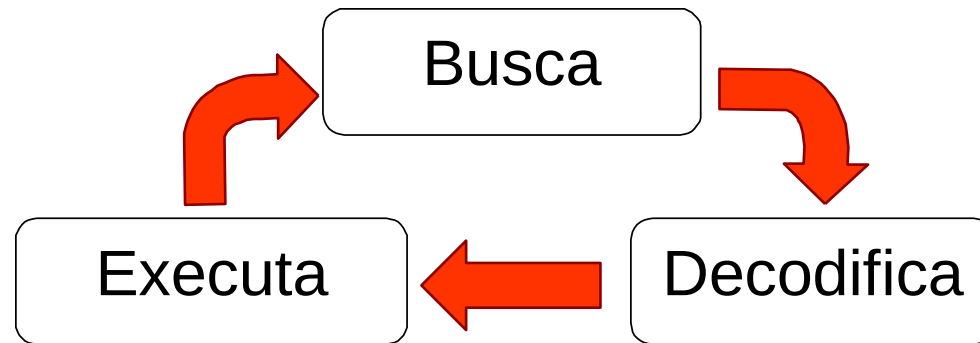


Relógio (Clock)

- Lógica combinacional opera sobre dados durante ciclos de clock
Entre subidas ou descidas do clock (clock edges)
- Execução típica na CPU
Lê conteúdo de elementos sequenciais → envia valores para lógica combinacional → escreve resultados em um ou mais elementos sequenciais
- Mudança de estado (escritas) ocorre nos clock edges
Se não ocorrer, deve haver um sinal explícito para habilitar escrita



Implementação Monociclo



Construindo a Unidade de Processamento (Datapath)

- Precisamos analisar o que cada etapa do ciclo de execução de uma instrução precisa

Busca e execução

- Precisamos analisar o que cada instrução do repertório precisa

`add, sub, and, or, slt, lw, sw, beq`

Componentes Básicos: Busca de Instrução

- Memória

Entrada: endereço , Saída: instrução

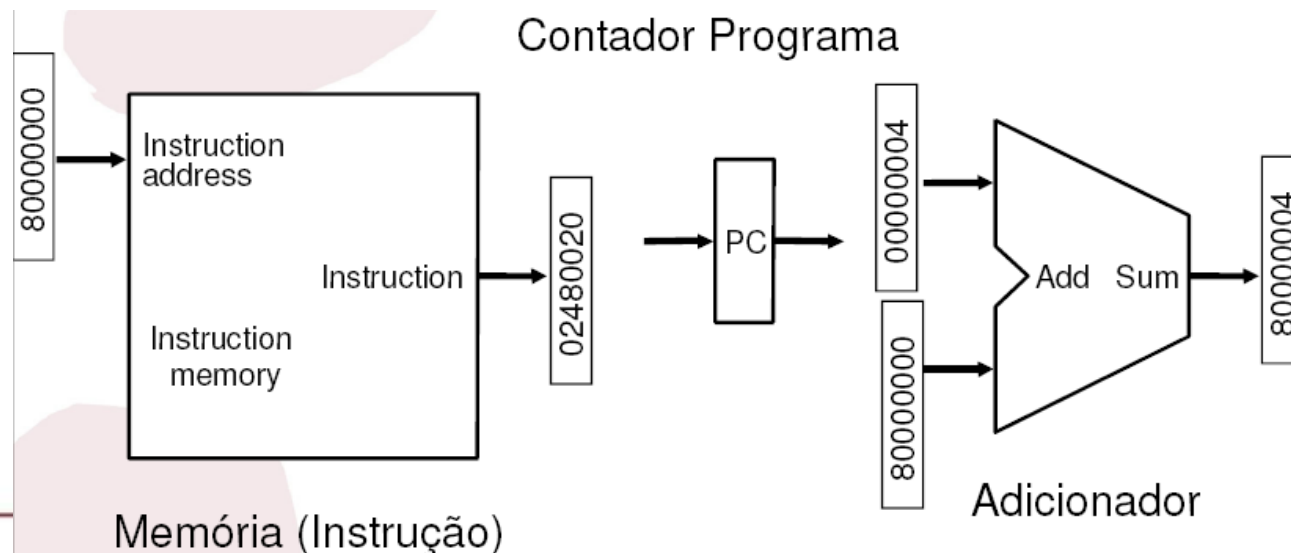
- Registrador PC de 32 bits

Contém endereço da instrução a ser executada

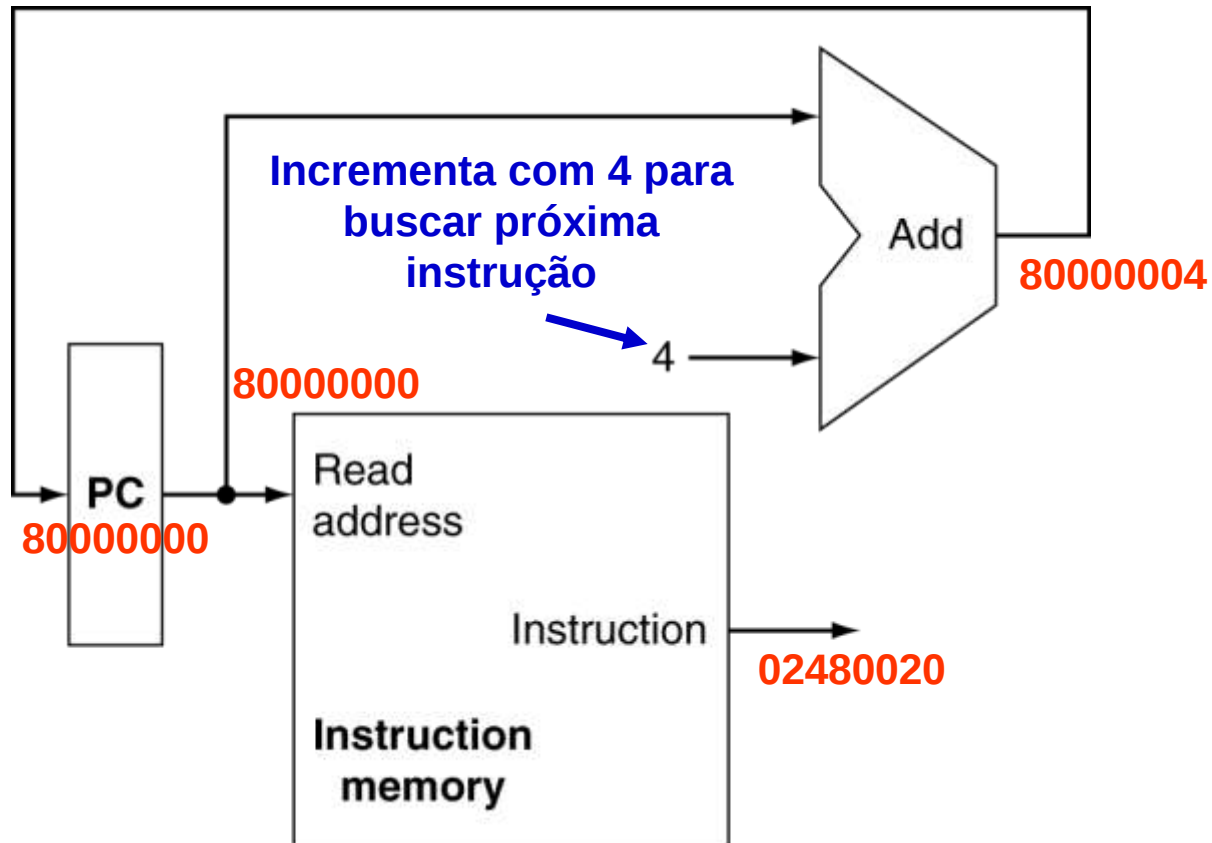
Atualizado todo ciclo de clock

- Somador

Soma PC com valor constante

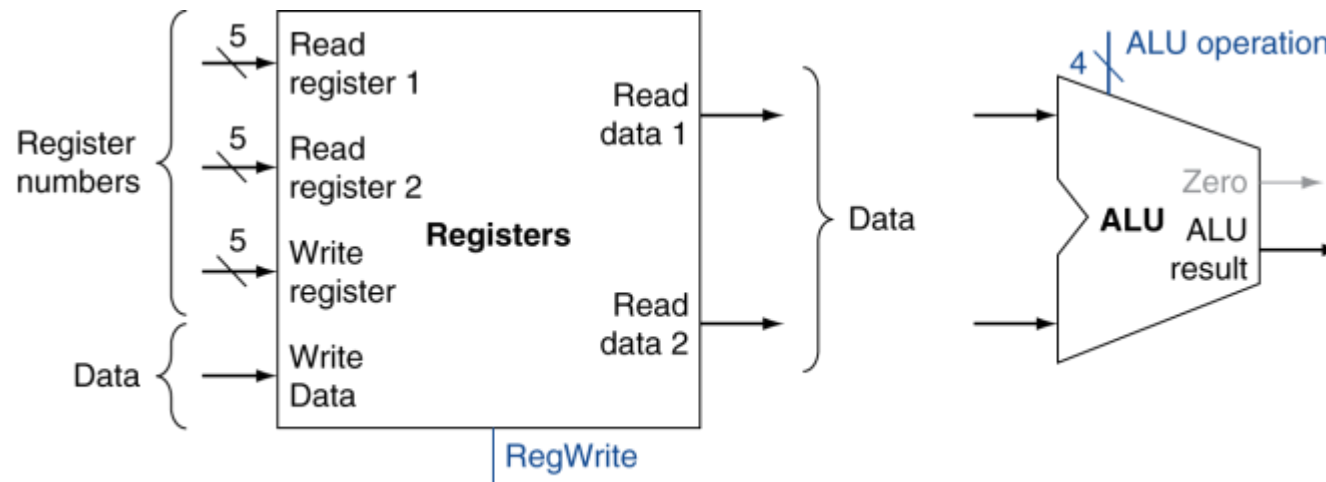


Busca de Instrução



Componentes Básicos: Instruções Aritméticas e Lógicas

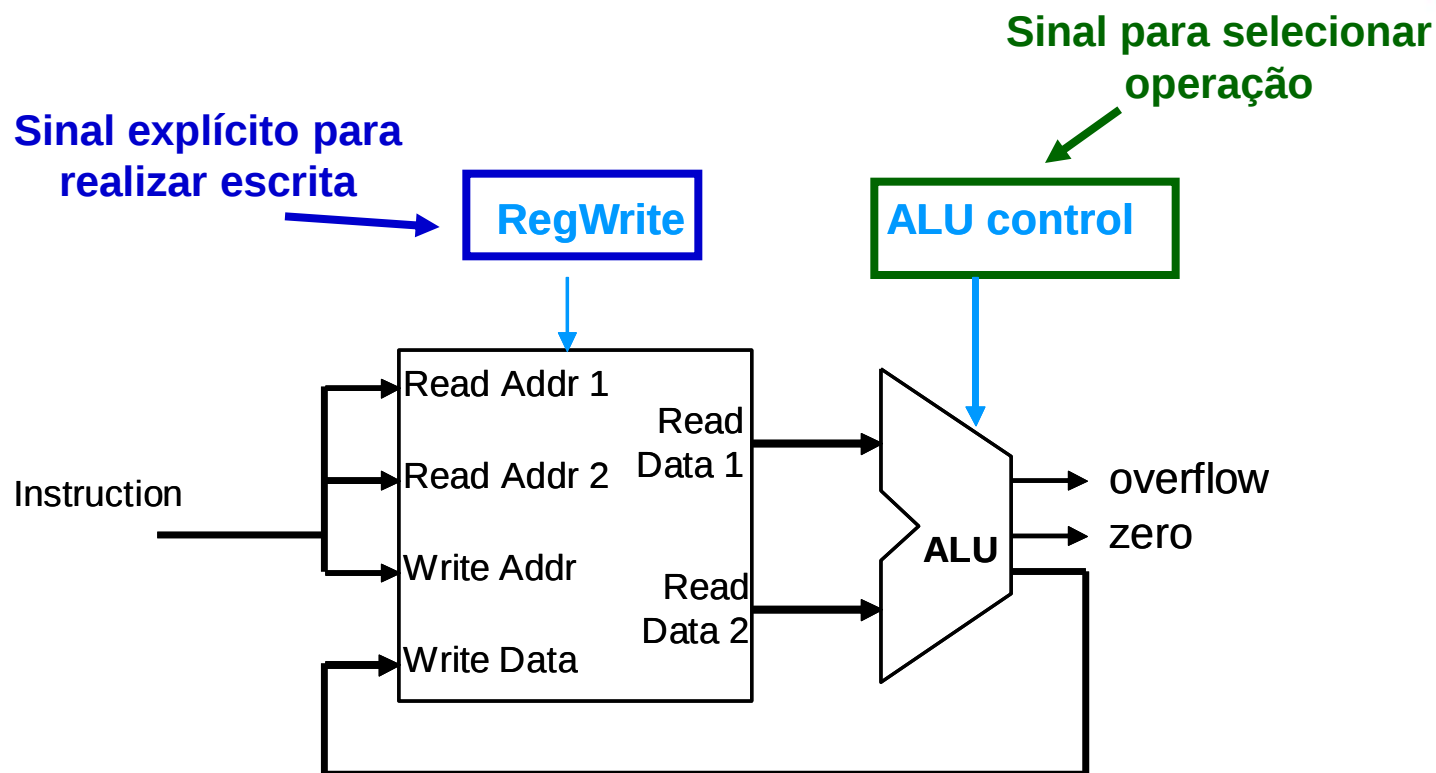
- Banco de registradores
 - Lê dois registradores
 - Armazena resultado em um registrador
- ALU calcula operandos vindos dos registradores



a. Registers

b. ALU

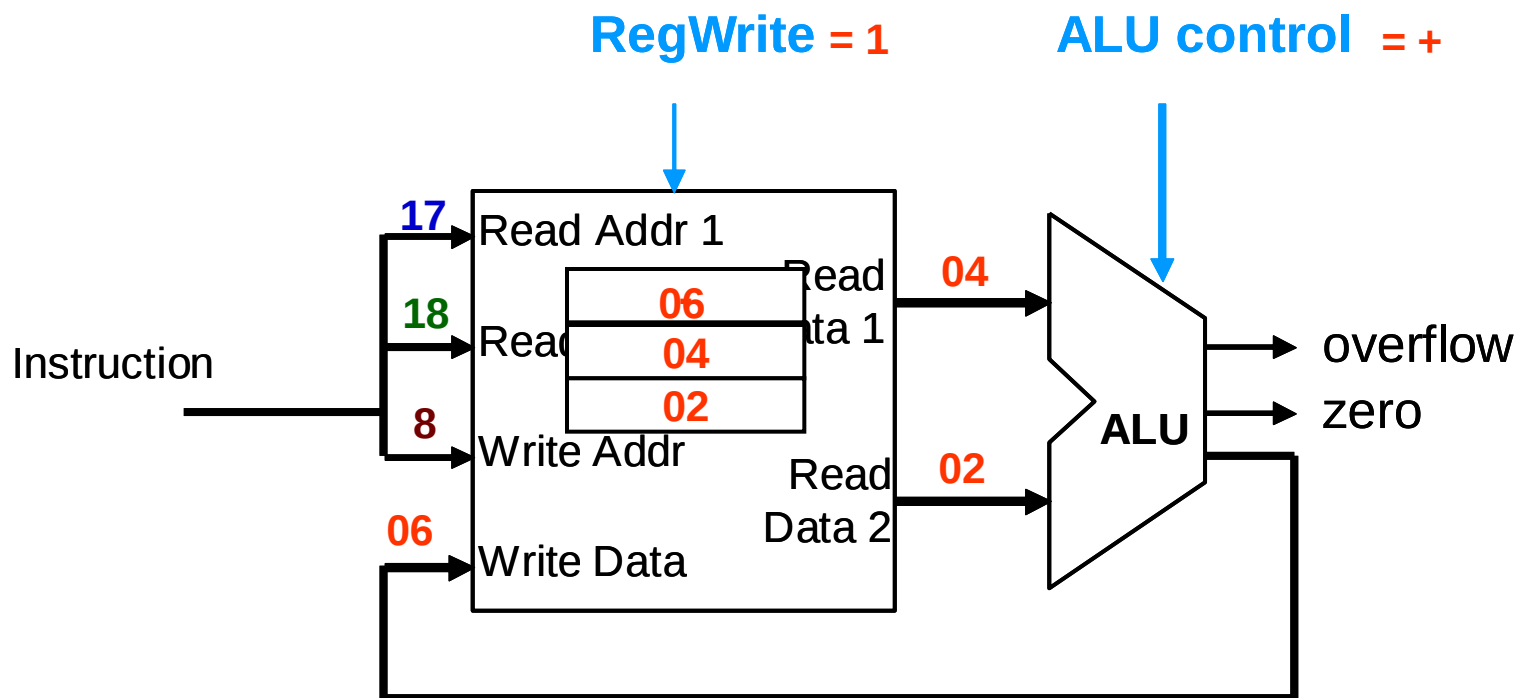
Instruções Aritméticas e Lógicas



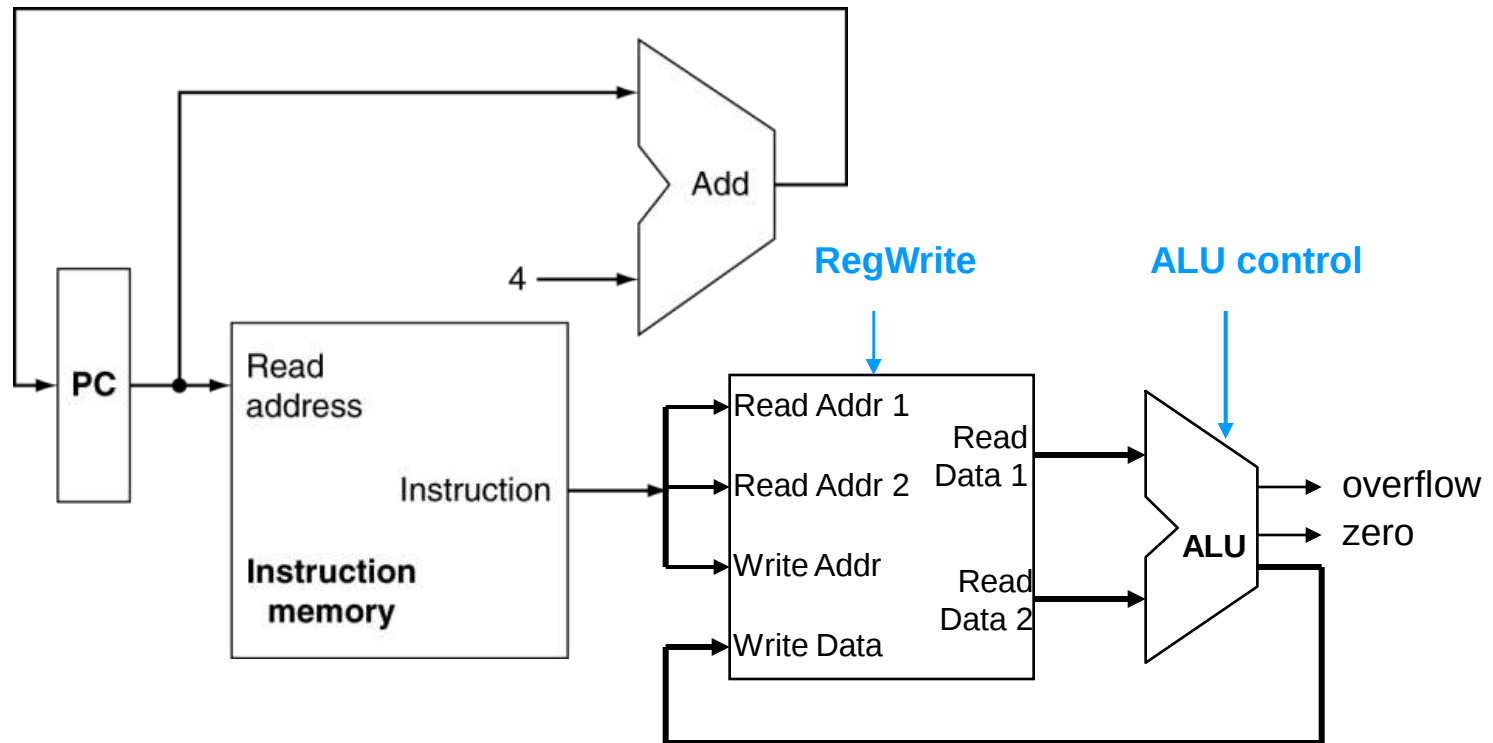
Instruções Aritméticas e Lógicas

`add $t0, $s1, $s2`

op	0
rs	17
rt	18
rd	8
shamt	0
func	32



Busca e Execução de Instruções Aritméticas e Lógicas



Componentes Básicos: Instruções de Armazenamento

- **Memória de dados**

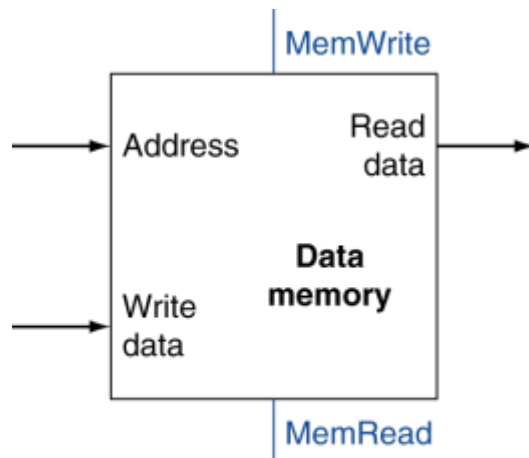
Leitura e escrita

- **Unidade de extensão de sinal**

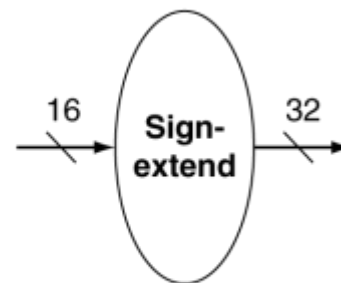
Para transformar o deslocamento de 16 bits em 32 bits

- Registradores para ler e escrever

- ALU para calcular endereço: registrador base + deslocamento

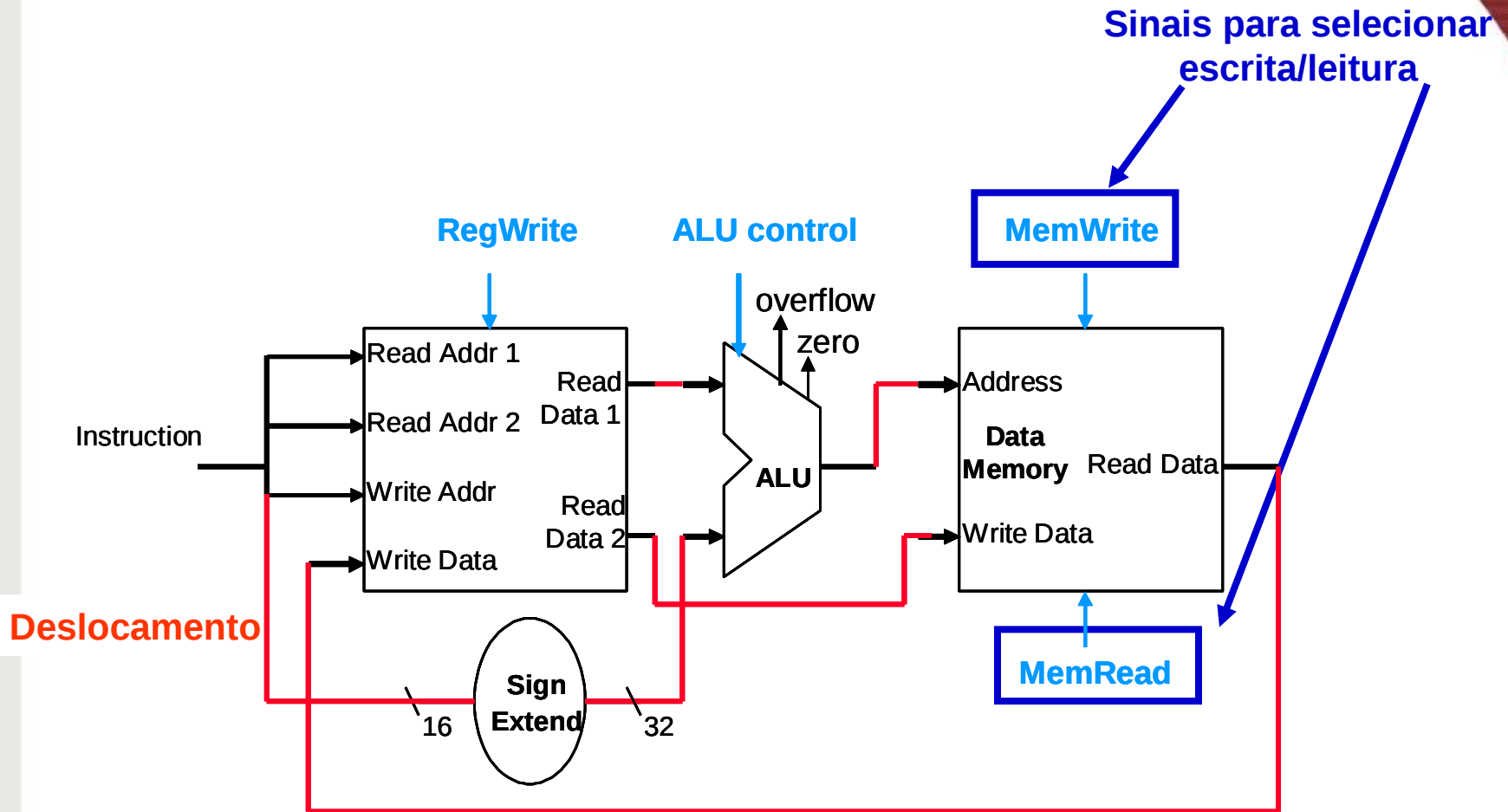


a. Data memory unit

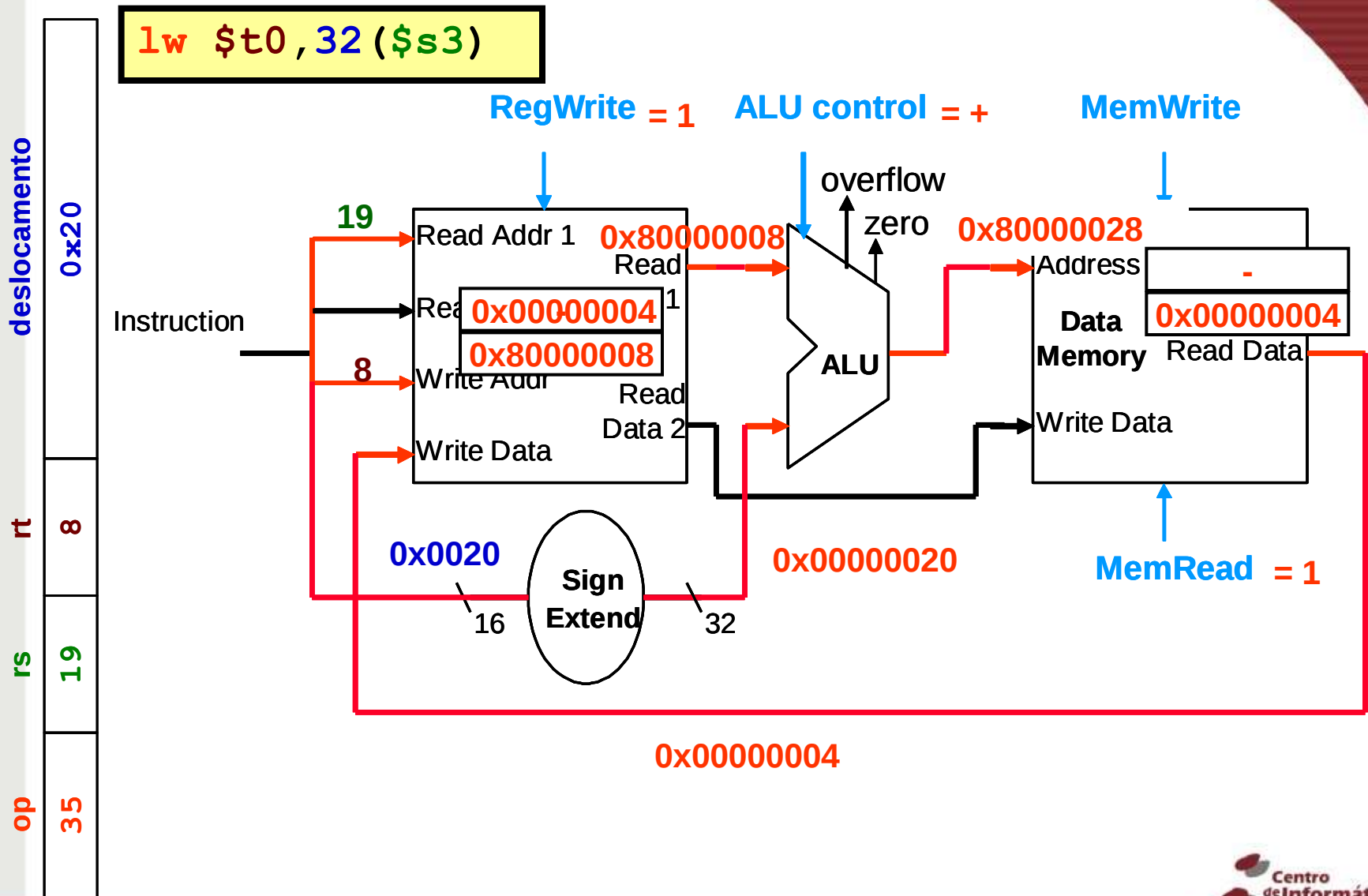


b. Sign extension unit

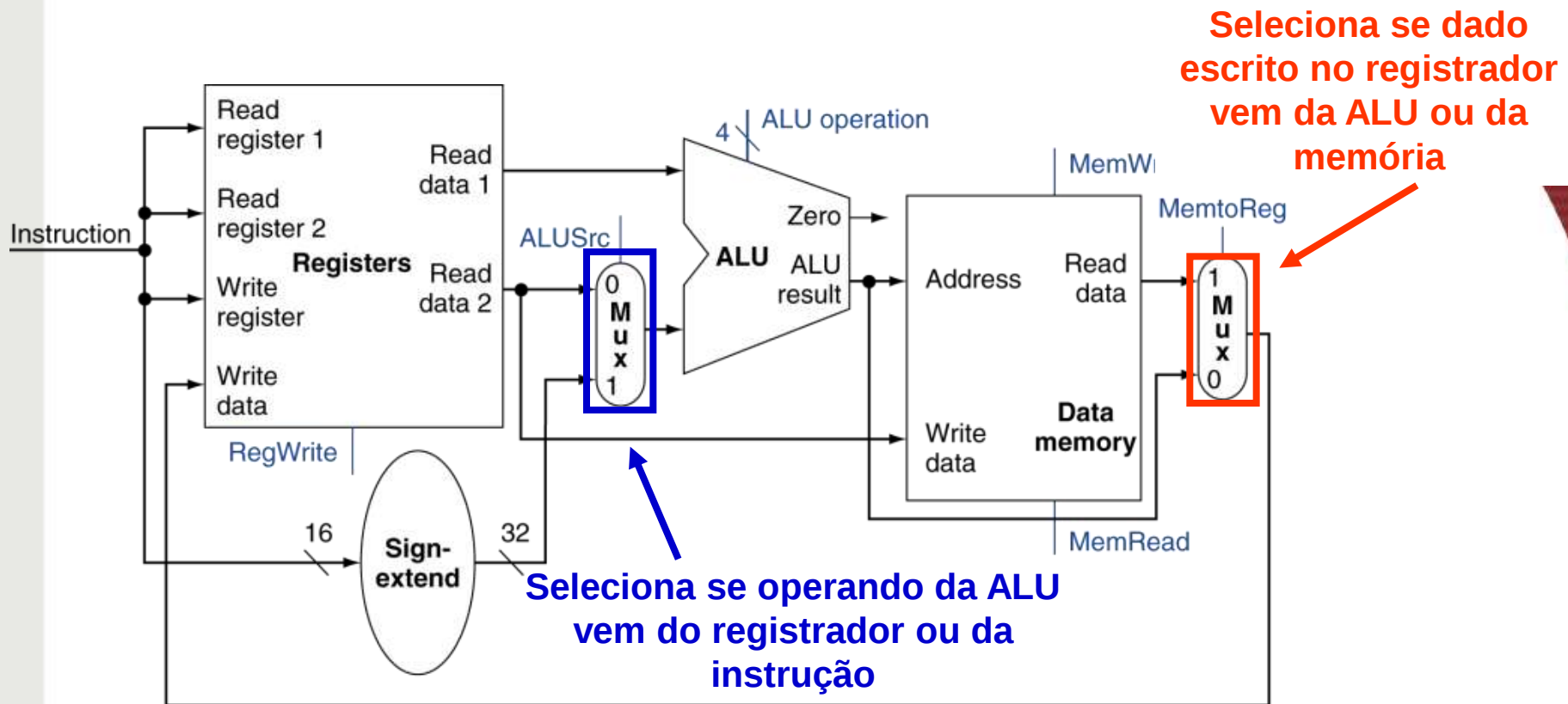
Instruções de Armazenamento



Instruções de Armazenamento

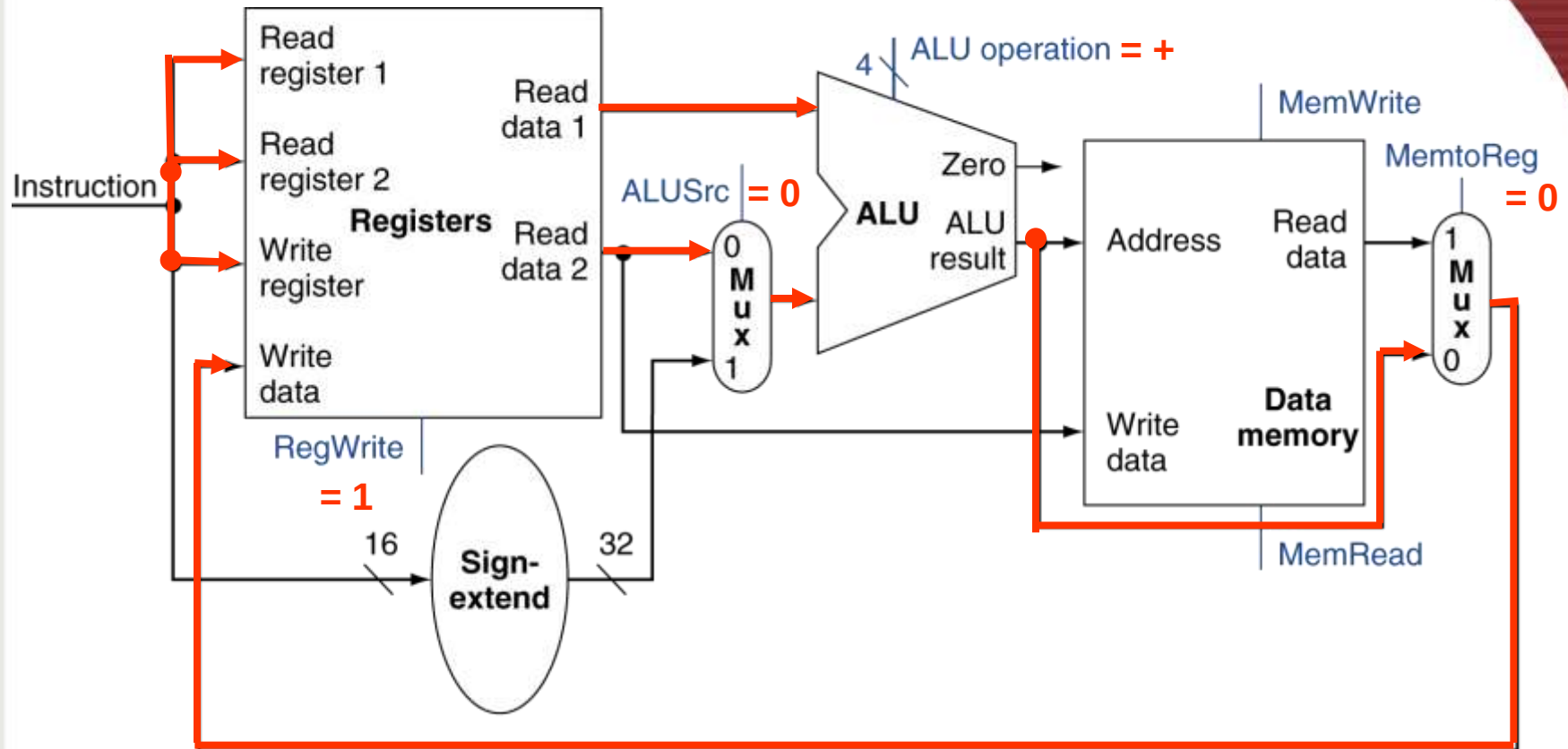


Instruções Aritméticas e de Armazenamento

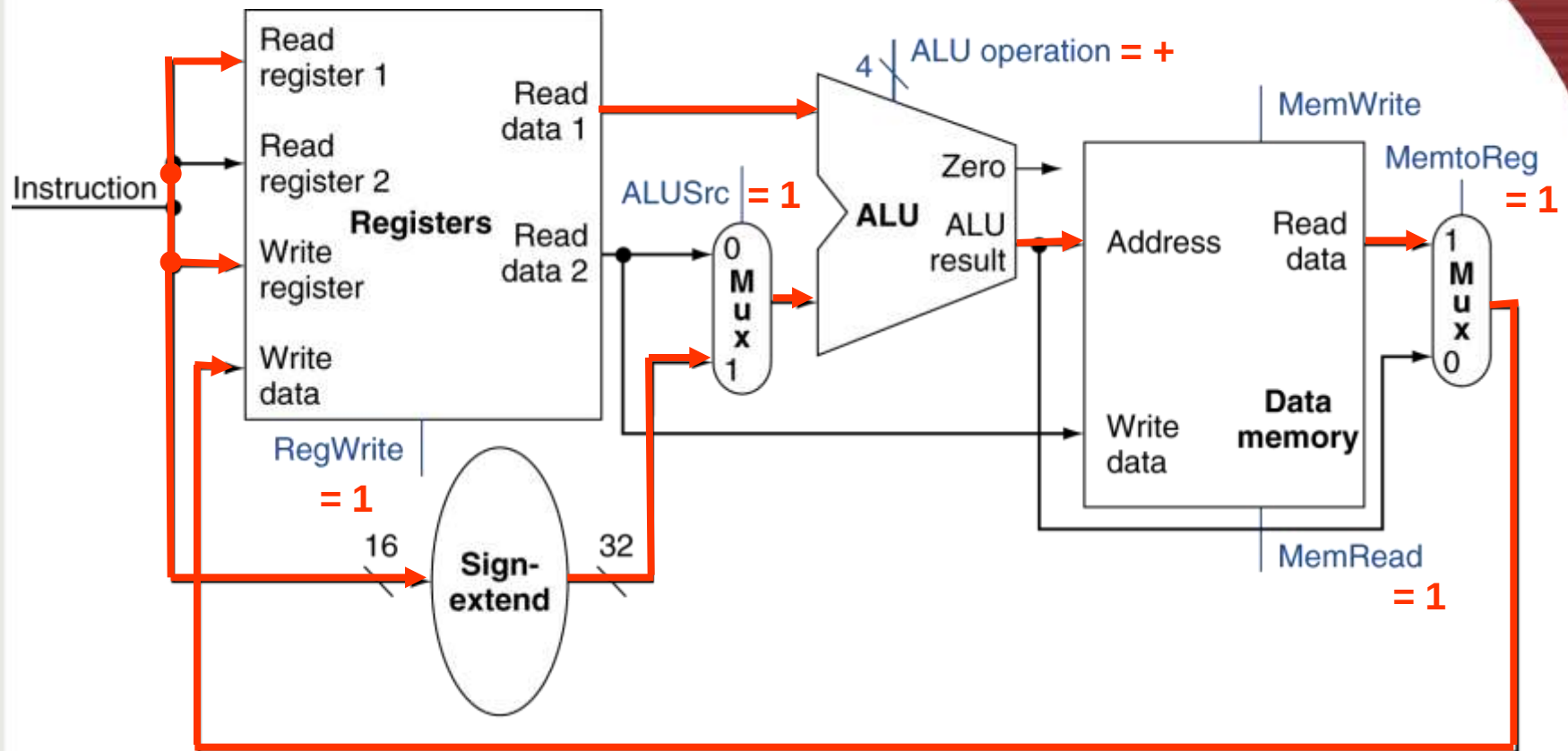


- Dois multiplexadores são necessários para selecionar entrada:
 - Registrador (aritmética) ou Deslocamento (armazenamento)
 - ALU (aritmética) ou Memória (armazenamento)

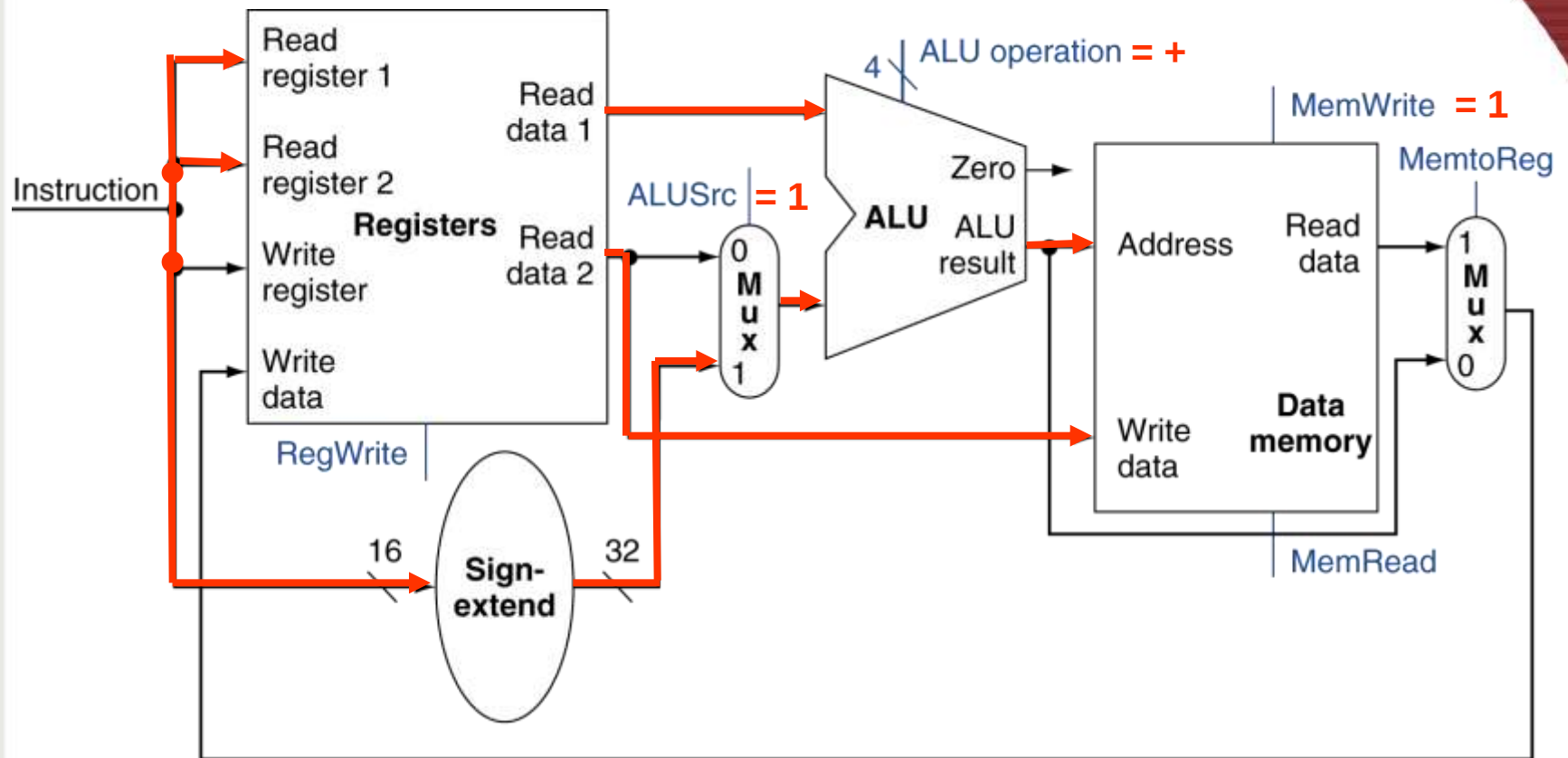
Executando ADD



Executando LW



Executando SW

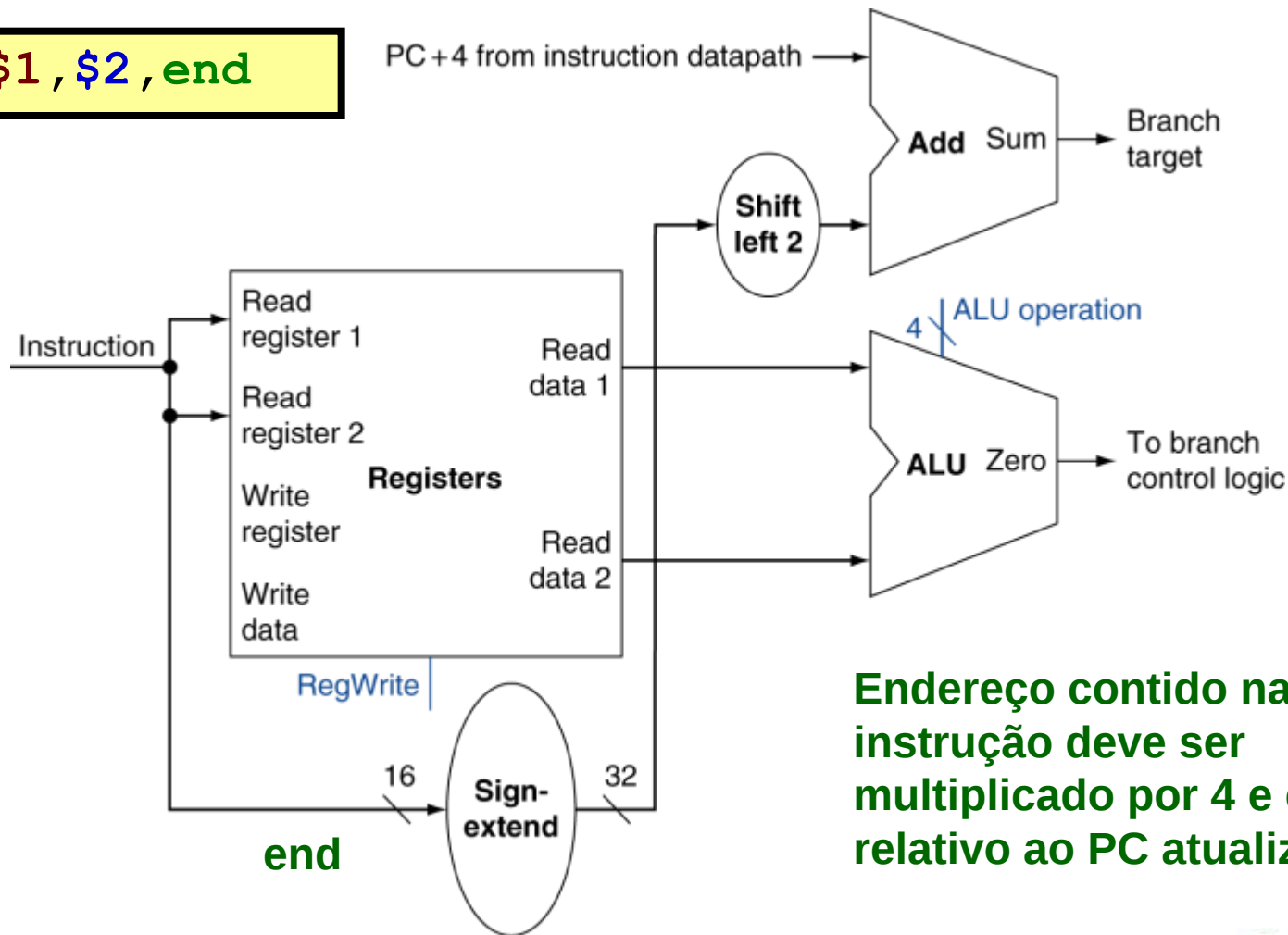


Componentes Básicos: Instruções de Branch

- Registradores para ler e escrever
- ALU para comparar operandos
 - Subtrai operandos e checa a saída 0
- Unidade de extensão de sinal
 - Para transformar o endereço contido na instrução (relativo) de 16 bits em 32 bits
- **Unidade para deslocar de 2 bits** para esquerda o endereço relativo
 - Multiplicar por 4
- Somador **para adicionar o endereço relativo x 4 ao endereço do PC**

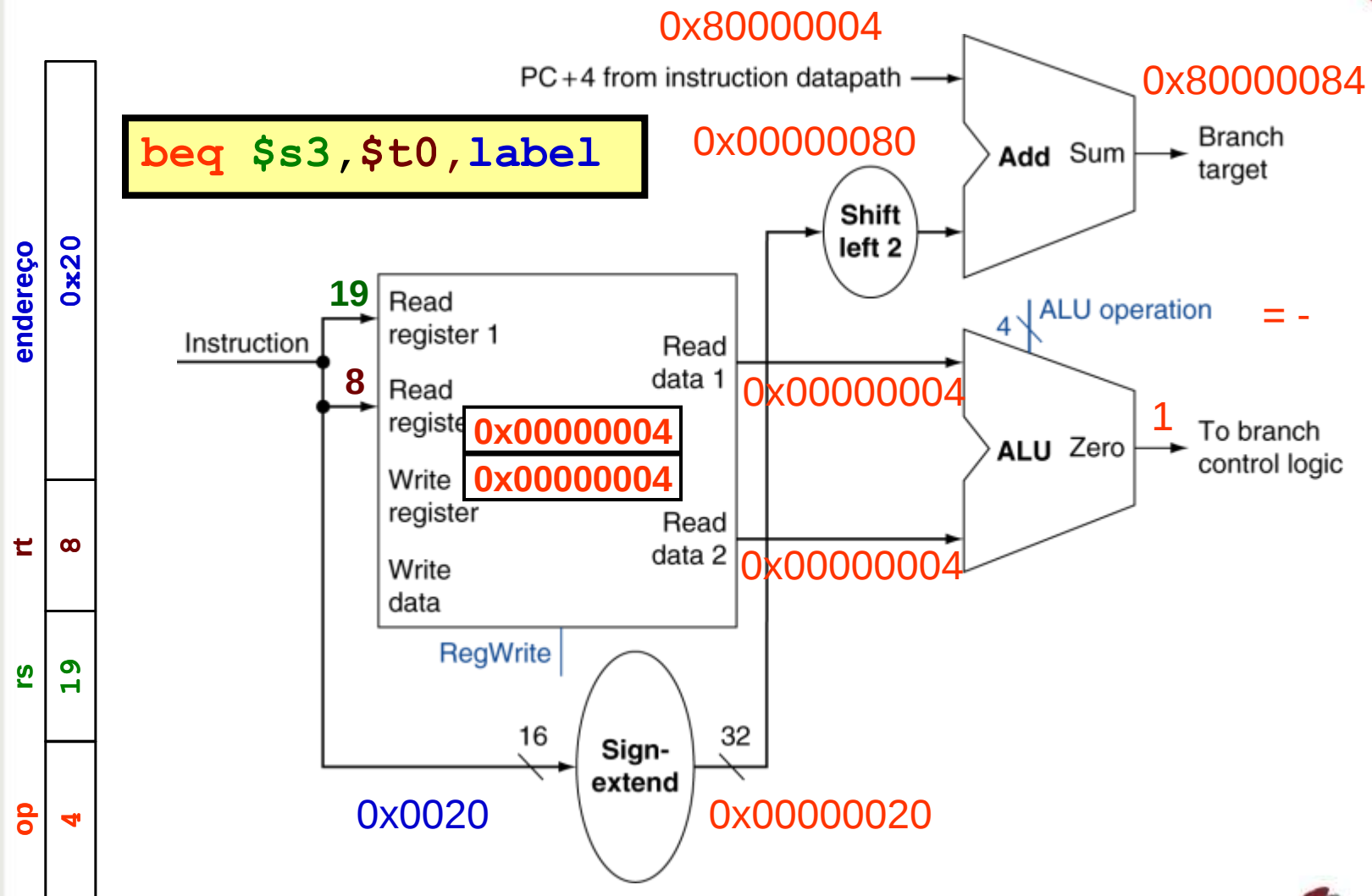
Instrução de Branch

beq \$1, \$2, end

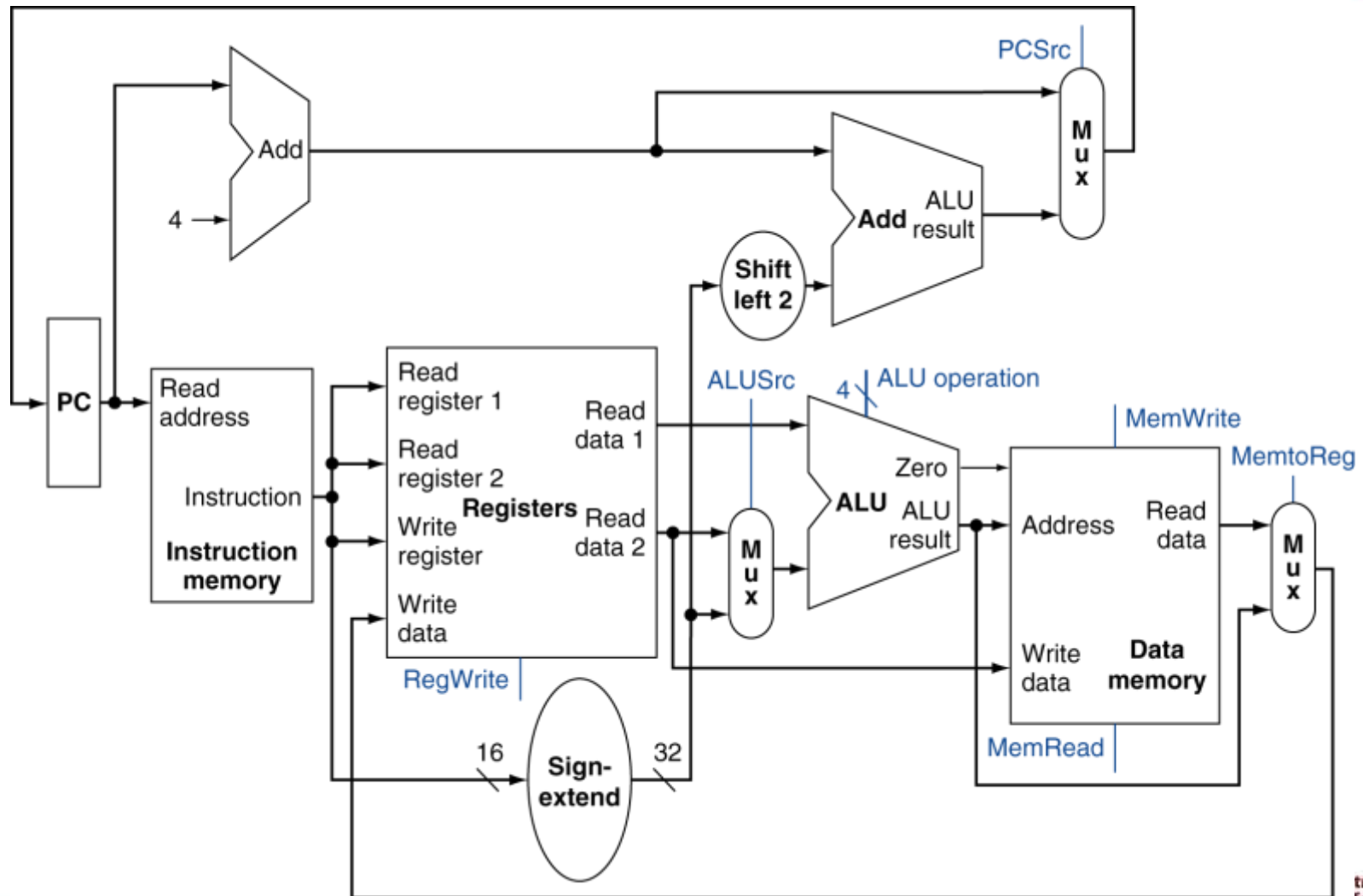


Endereço contido na instrução deve ser multiplicado por 4 e é relativo ao PC atualizado

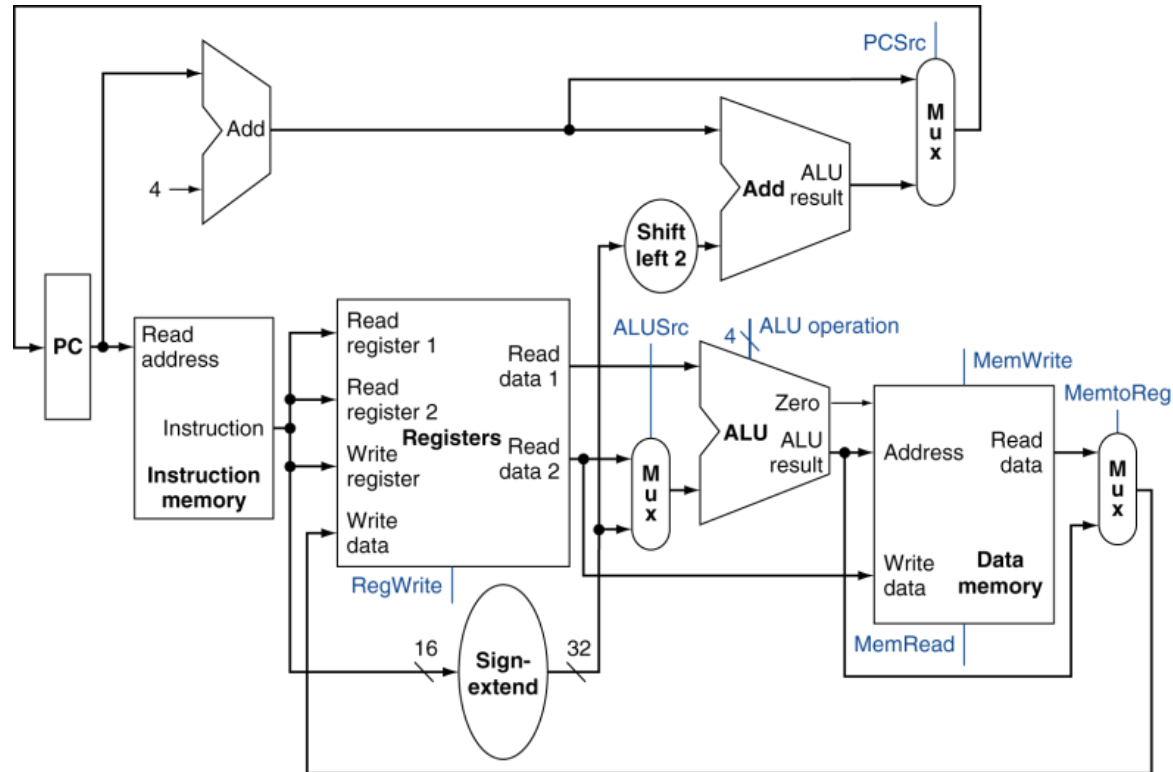
Executando BEQ



Unidade de Processamento Completa



O Que É Que Falta?



- Unidade que decodifique as instruções

Escolha operação da ALU, controle escrita em registrador, controle leitura/escrita na memória, etc

Controle da ALU

■ ALU usada para:

Load/Store: Função = soma

Branch: Função = subtração

Aritmética/Lógica: depende do campo funct

ALU control	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set-on-less-than
1100	NOR

Controle da ALU

- Podemos derivar do opcode um sinal de 2 bits chamado ALUOp

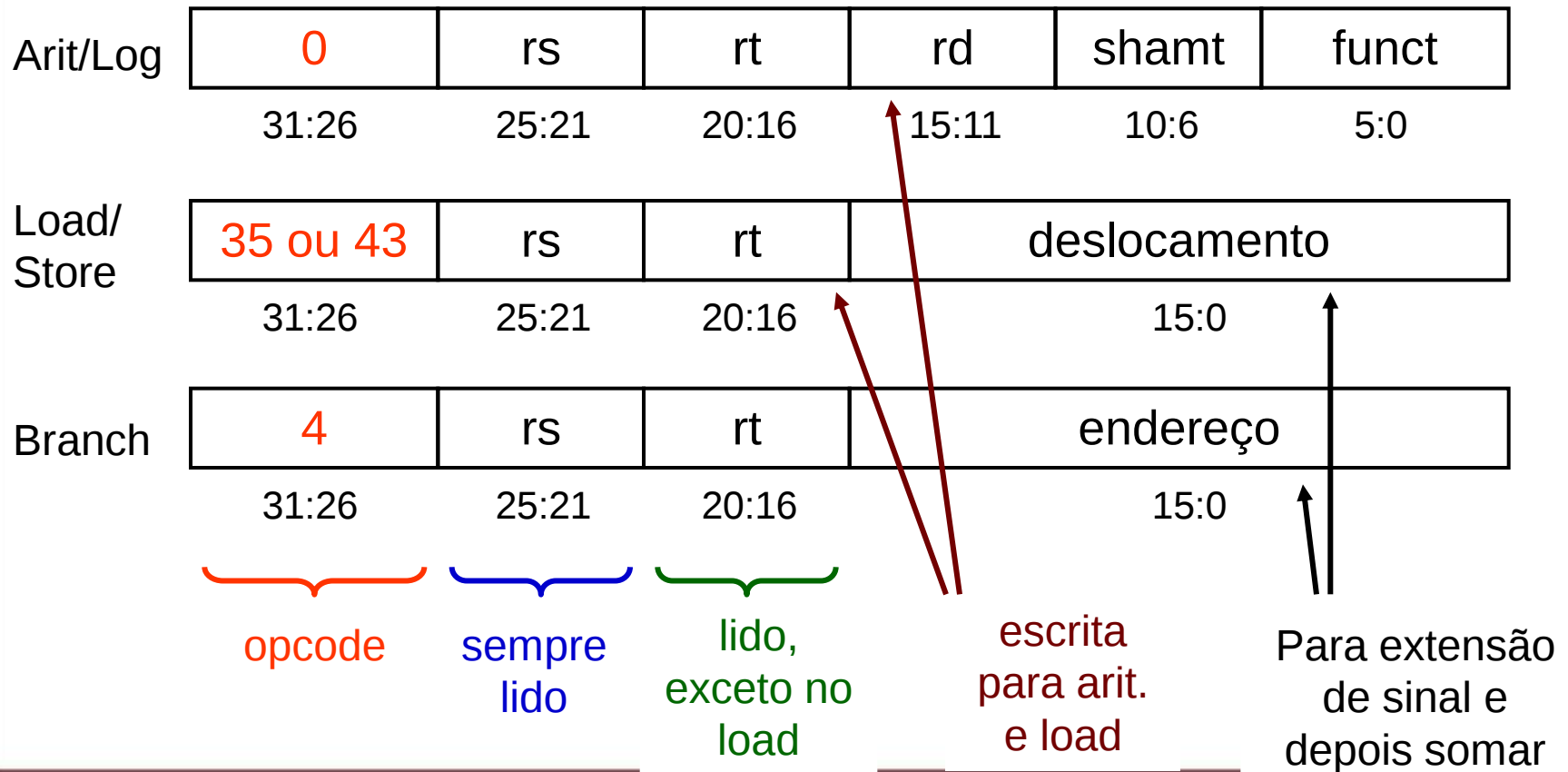
Lógica combinacional deriva o controle da ALU

opcode	ALUOp	Operation	funct	ALU function	ALU control
lw	00	load word	XXXXXX	add	0010
sw	00	store word	XXXXXX	add	0010
Beq	01	branch equal	XXXXXX	subtract	0110
Aritmética/ lógica	10	add	100000	add	0010
		subtract	100010	Subtract	0110
		AND	100100	AND	0000
		OR	100101	OR	0001
		set-on-less-than	101010	set-on-less-than	0111

Unidade de Controle Principal

■ Sinais de controle são derivados da instrução

Decodificação da instrução

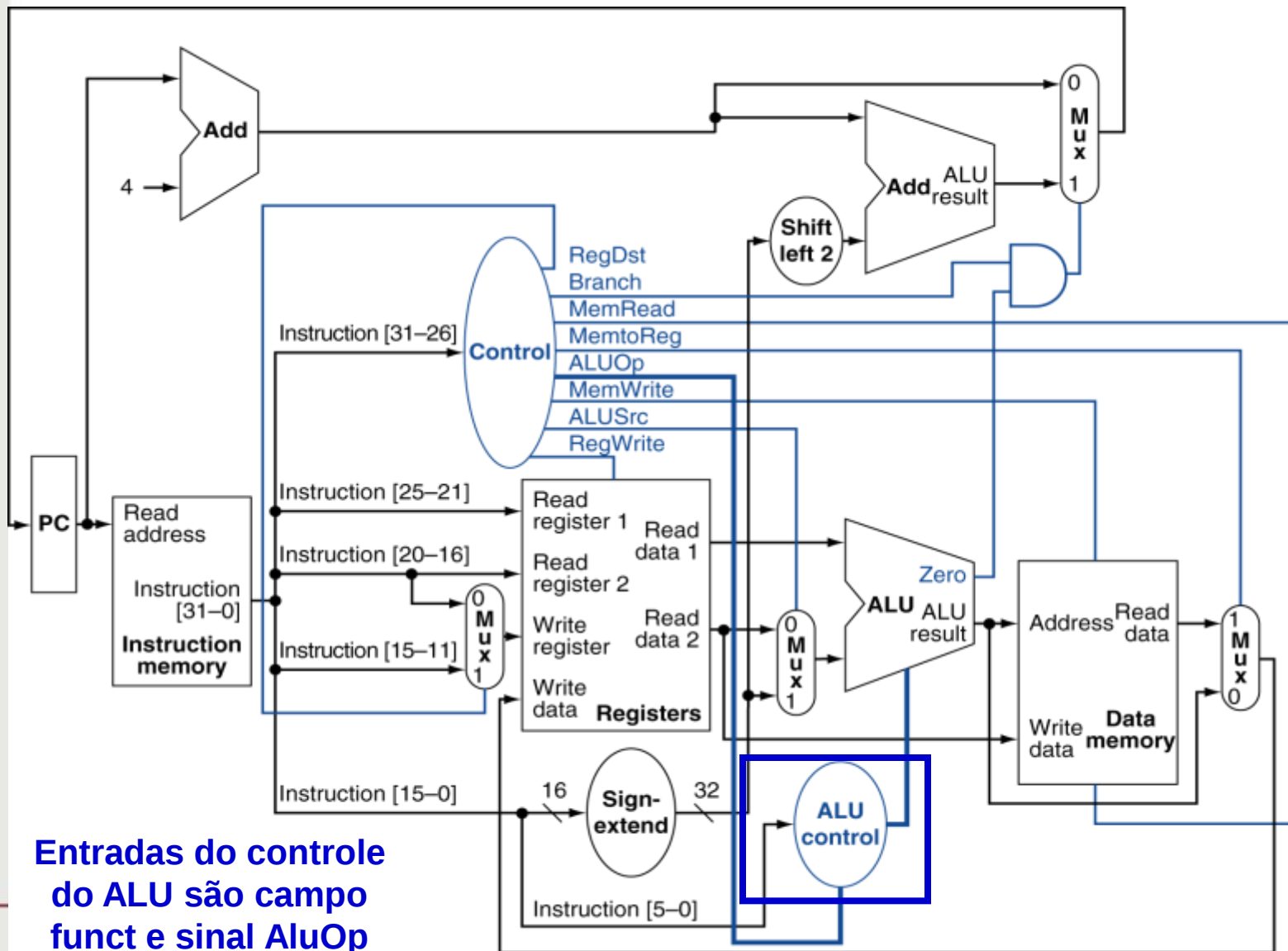


Sinais da Unidade de Controle

- Unidade de controle tem como entrada a instrução e como saída 8 sinais + ALUOp

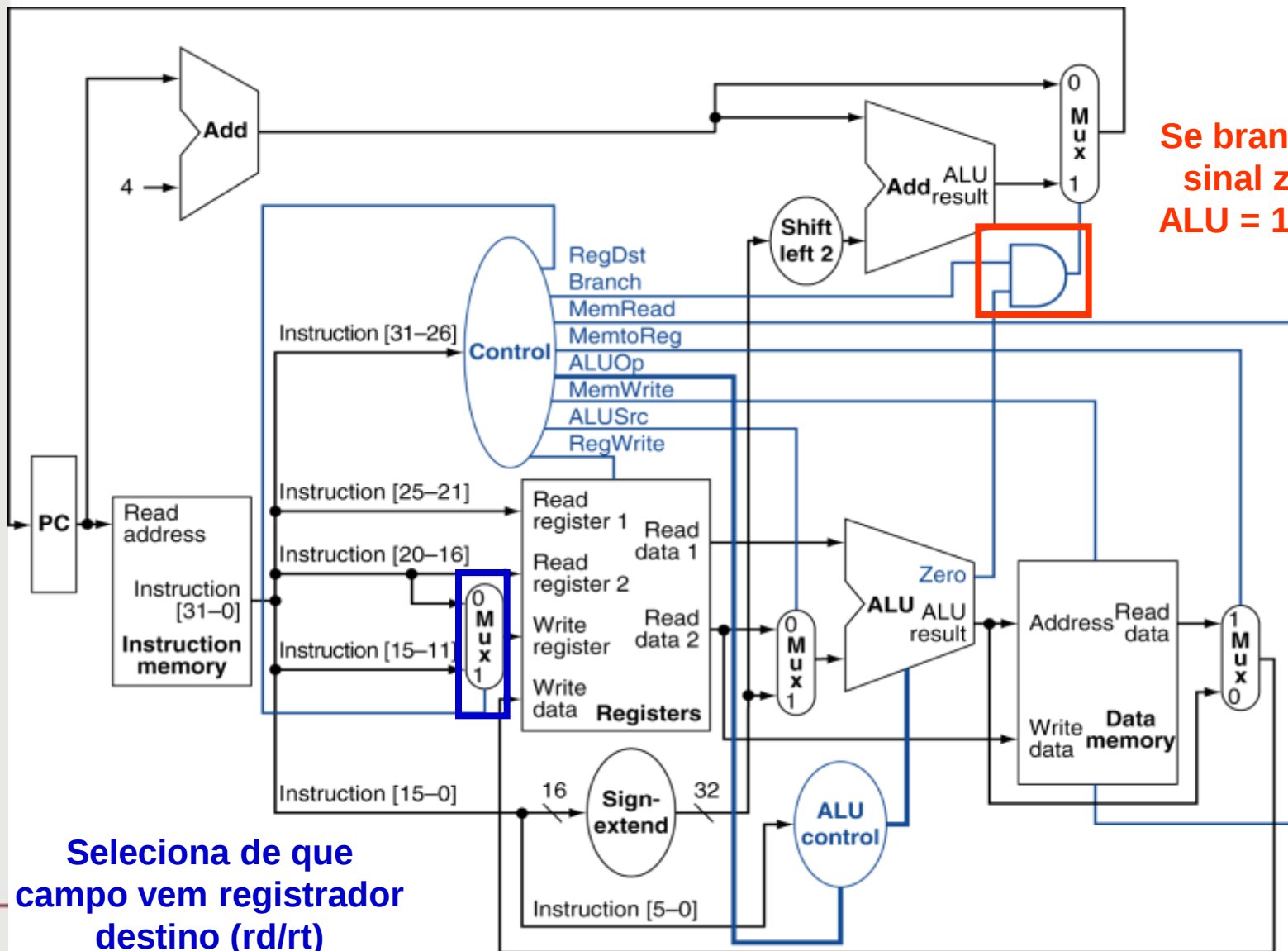
Nome	Efeito quando 0	Efeito quando 1
RegDst	Número do registrador destino para escrita vem de rt	Número do registrador destino para escrita vem de rd
RegWrite	Nada	Registrador da entrada “Write register” recebe valor da entrada “Write data”
ALUSrc	2º operando da ALU vem da 2a saída do banco de registradores	2º operando da ALU vem do valor estendido contido na instrução
Branch	PC recebe valor de PC + 4 que vem do somador	PC recebe valor do endereço do branch se operandos =
MemRead	Nada	Dado da memória relativo ao endereço especificado é colocado na saída
MemWrite	Nada	Dado da memória relativo ao endereço especificado é substituído pelo que tem no “Write data”
MemTo Reg	Valor escrito na entrada “Write data” vem da ALU	Valor escrito na entrada “Write data” vem da memória

Unidade de Processamento Com Controle

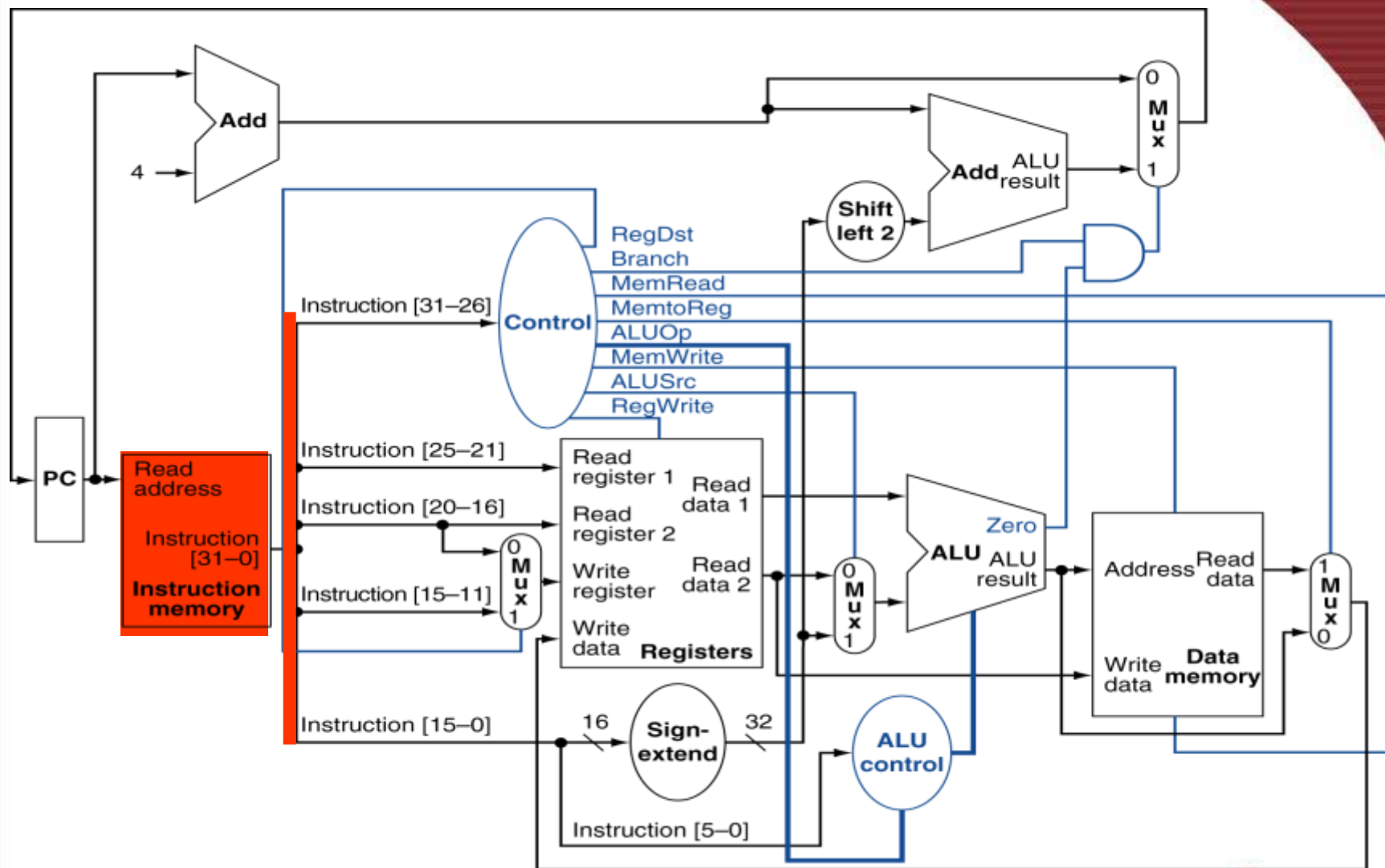


Entradas do controle
do ALU são campo
funct e sinal AluOp

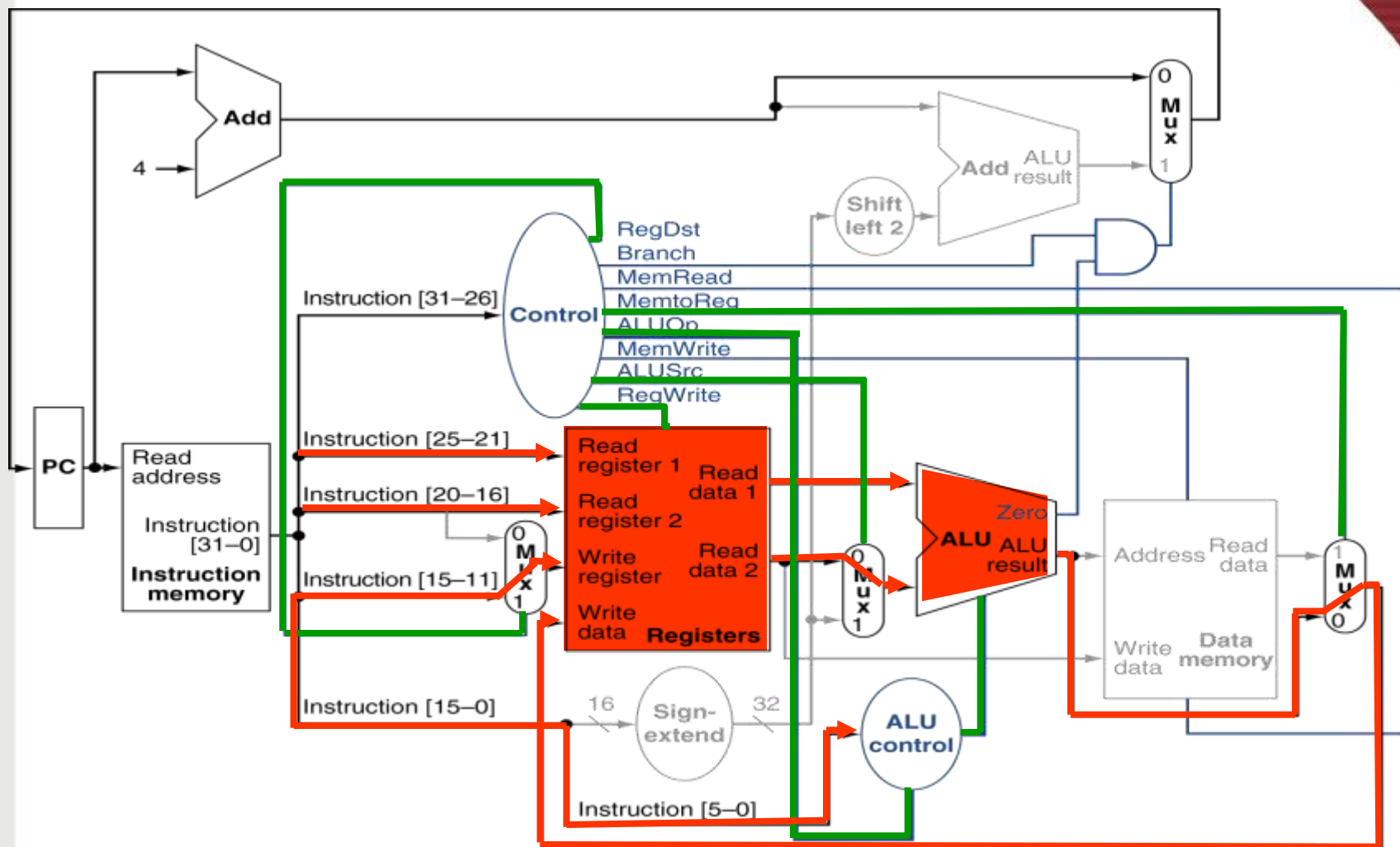
Unidade de Processamento Com Controle



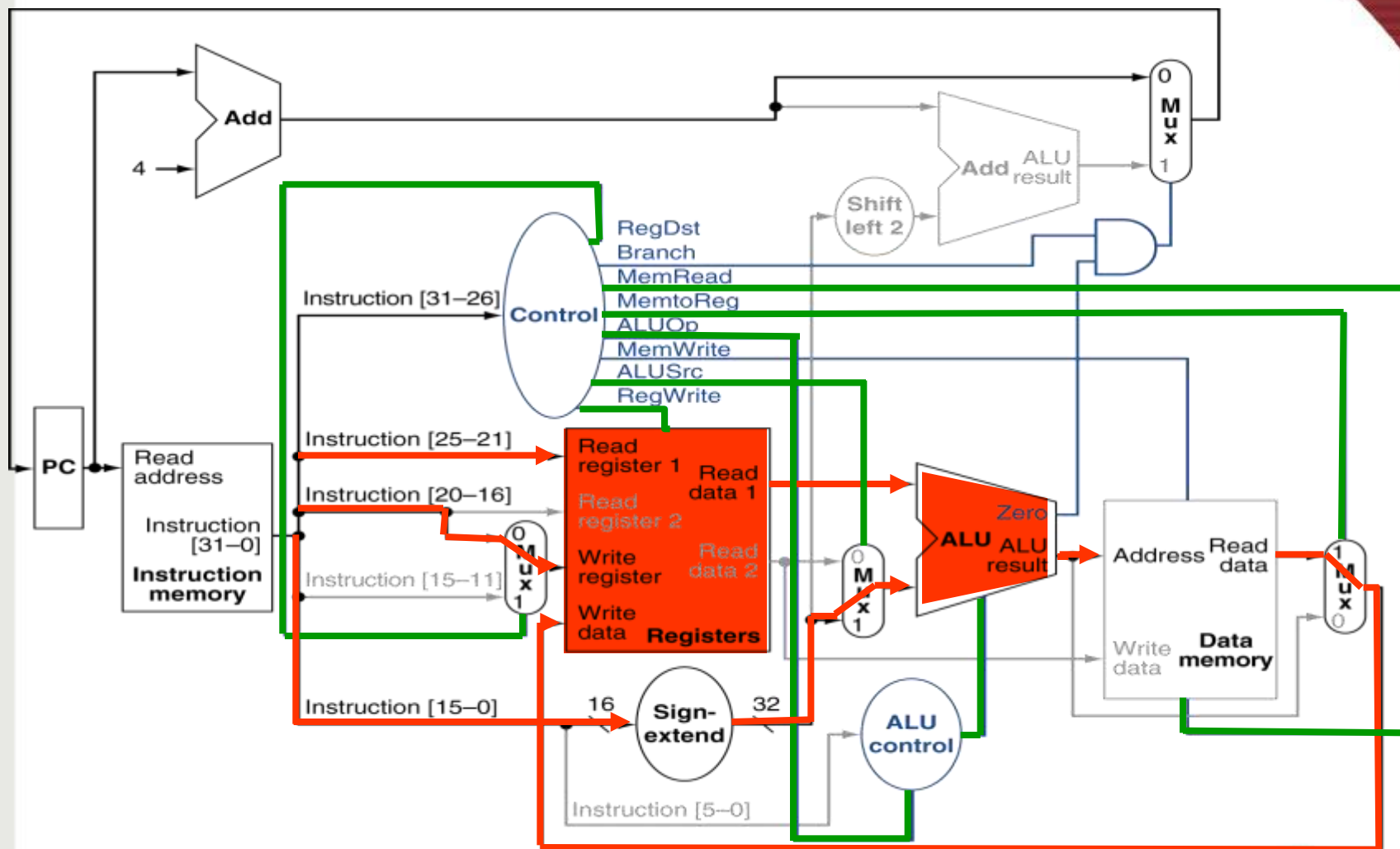
Busca de Instrução



Instruções Aritméticas/Lógicas



Instrução LW



Instrução BEQ

