UNIOESTE – Universidade Estadual do Oeste do Paraná
Centro de Engenharias e Ciências Exatas
Campus de Foz do Iguaçu

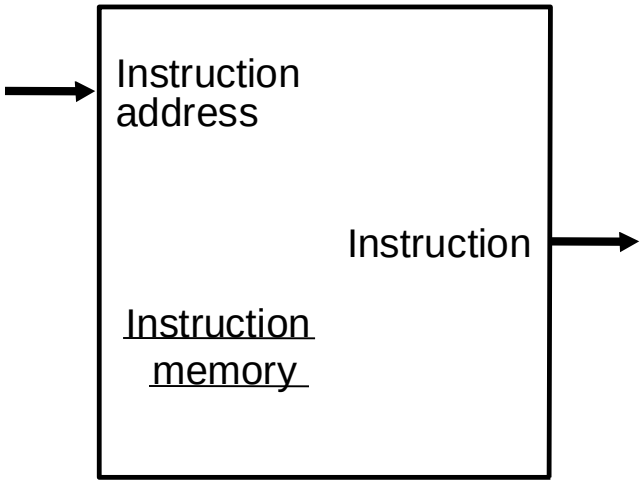
Via de Dados e Unidade de Controle Arquitetura MIPS

Prof.: Fabiana Frata Furlan Peres

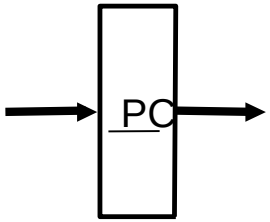
Foz do Iguaçu

Via de Dados

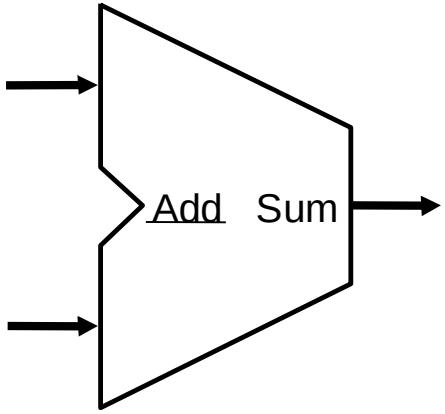
(Elementos essenciais para a execução de qualquer instrução)



a. Instruction memory



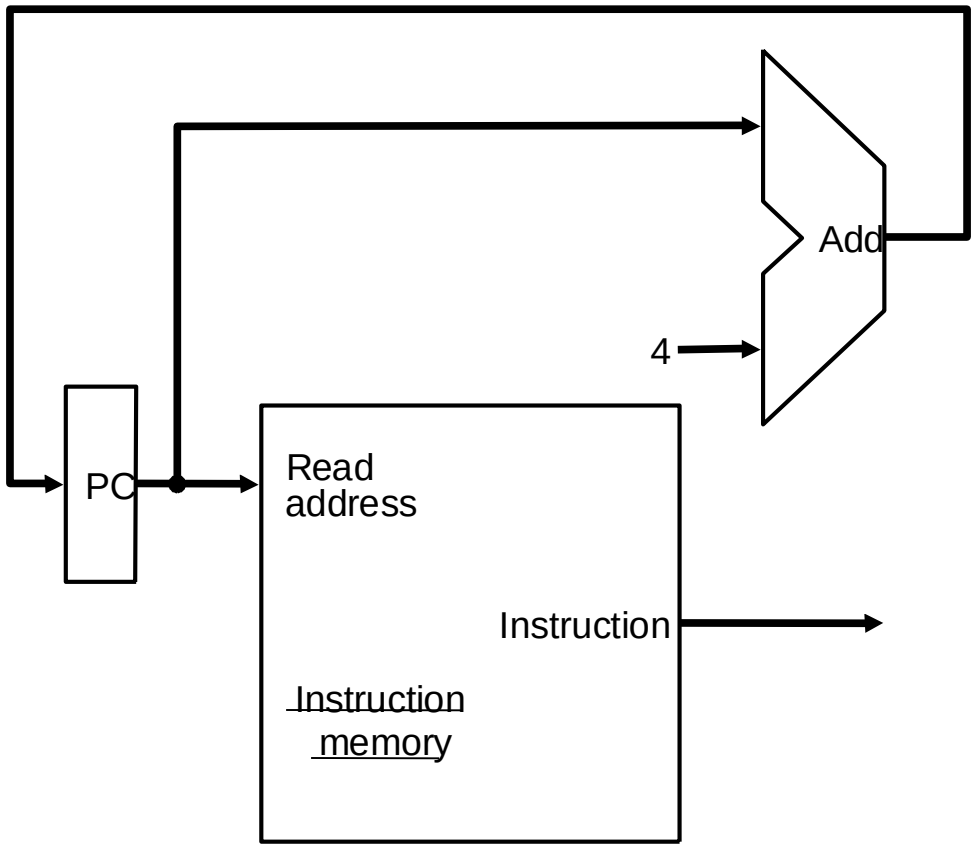
b. Program counter



c. Adder

Via de Dados

(Parte da via de dados usada para buscar instruções e incrementar o contador de instrução (PC))



Via de Dados

(Instruções tipo R)

- **Instruções tipo R:** ***(add, sub, and, or, slt, sll, srl)***

op	*rs*	*rt*	*rd*	*shamt*	*funct*

- **Instância de uma instrução:**

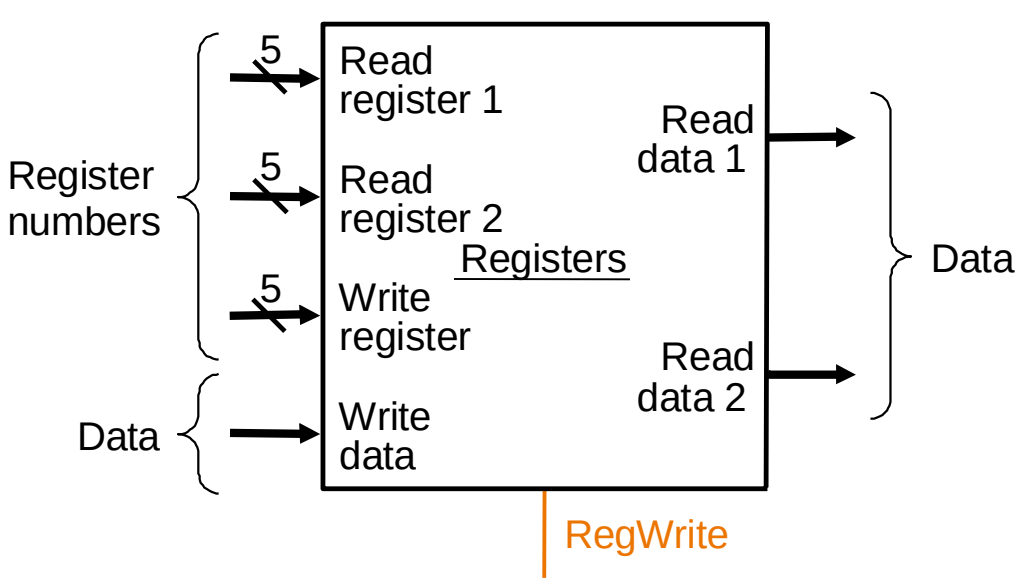
  add rd, rs, rt // rd= rs + rt

- **Requisitos para execução:**

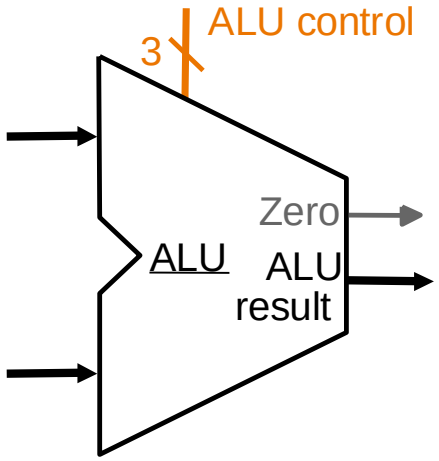
  - Ler o conteúdo de dois registradores (rs e rt);
  - Realizar uma operação, com o conteúdo dos registradores, numa ALU (add, sub, and, ...);
  - Escrever o resultado da operação em um outro registrador (rd).

Via de Dados

(Elementos adicionais para a execução de instruções tipo R)



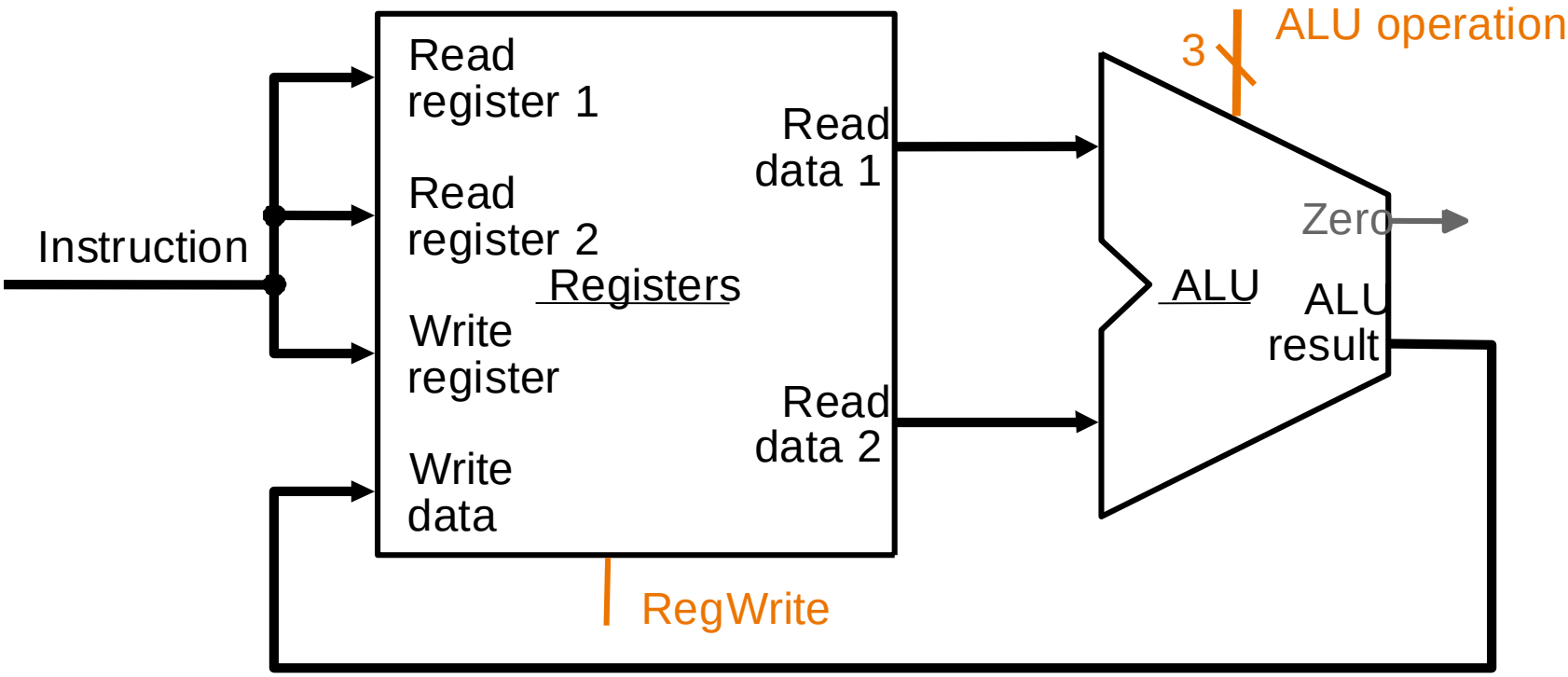
a. Registers



b. ALU

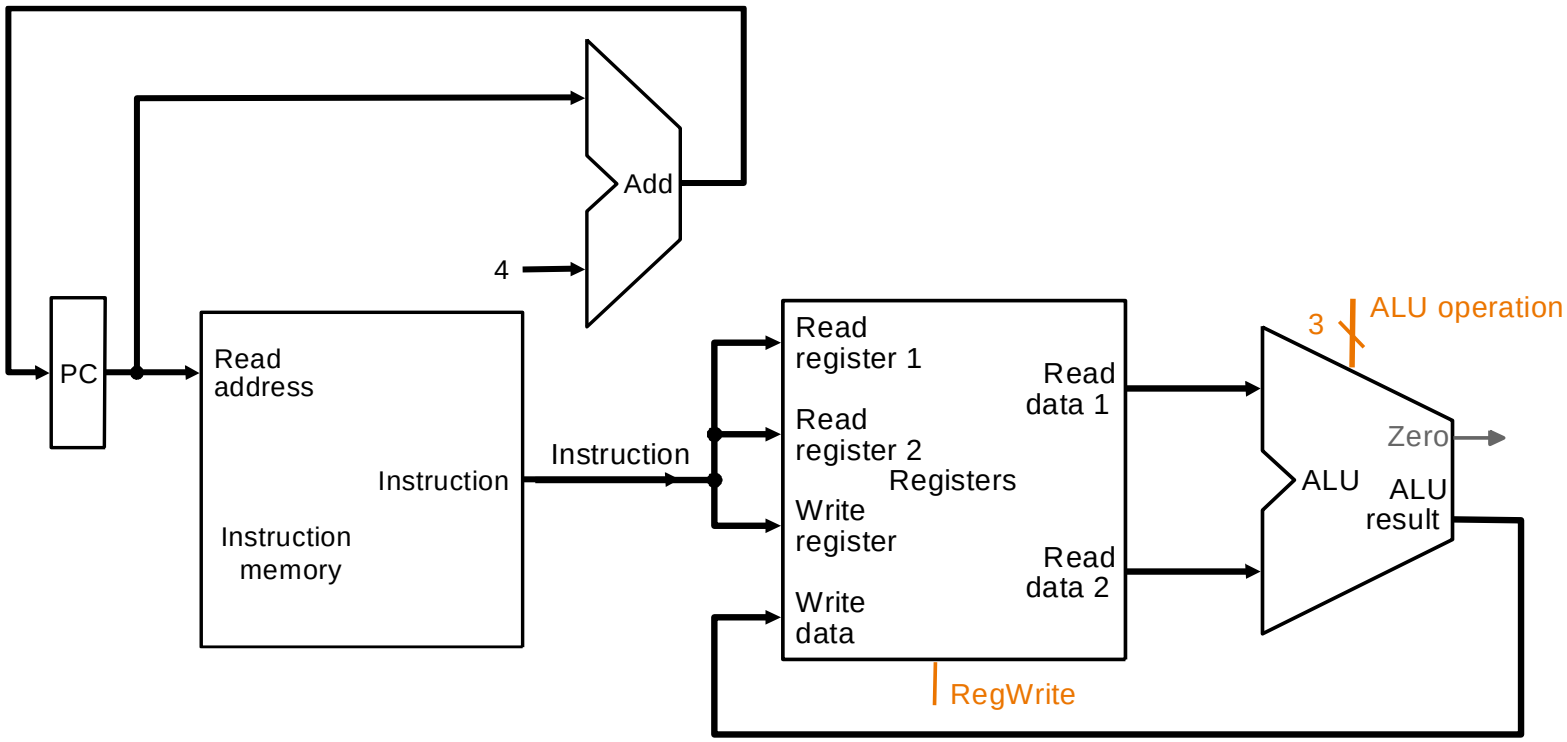
Via de Dados

(Parte da via de dados para a execução)



Via de Dados

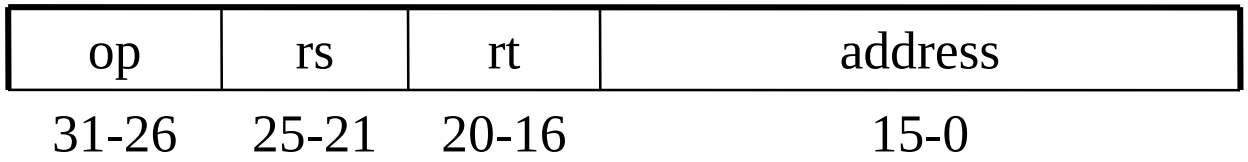
(Via de dados para a execução da R-type)



Via de Dados

(Instruções tipo I - lw e sw)

- **Instrução tipo I:**



- **Instância da instrução lw e sw:**

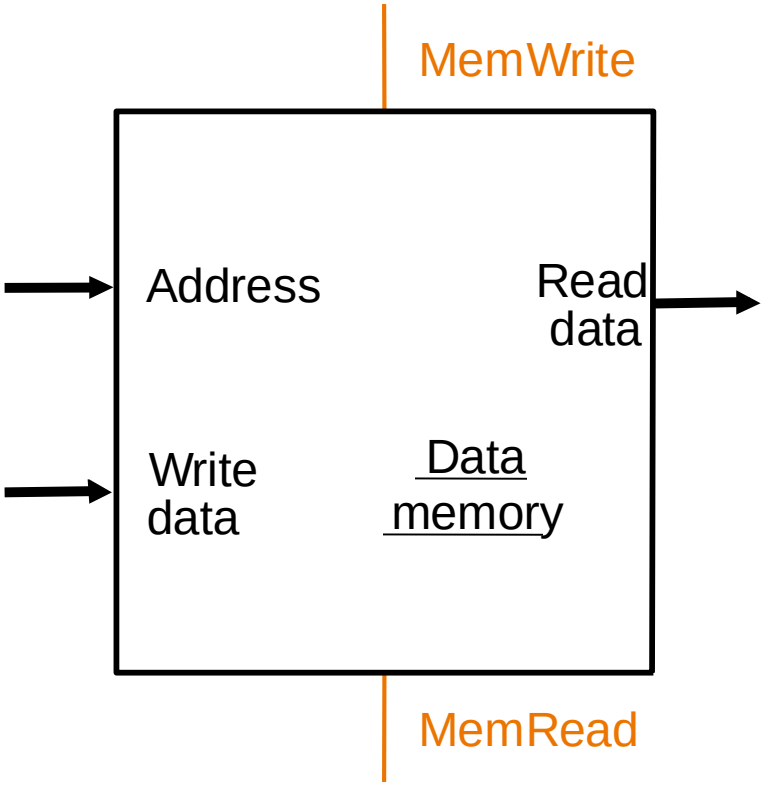
  lw rt, offset_value(rs) // rt= Mem[rs + offset_value]

  sw rt, offset_value(rs) // Mem[rs + offset_value]= rt

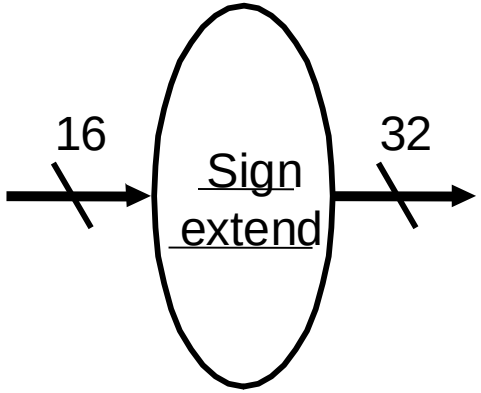
- **Requisitos para execução:**
  - Ler o conteúdo do registrador de base (rs) e o conteúdo do registrador rt se a instrução for sw;
  - Realizar a soma do registrador de base com o offset_value utilizando a ALU para calcular o endereço da memória;
  - Acessar a memória:
    - para ler o valor contido dentro do endereço calculado (se a instrução for a lw);
    - para escrever o valor contido no registrador rt no endereço calculado (se a instrução for sw)
  - Escrever o valor lido da memória dentro de um registrador (rt), se a instrução for lw.

Via de Dados

(Elementos adicionais para a execução da instrução lw e sw)



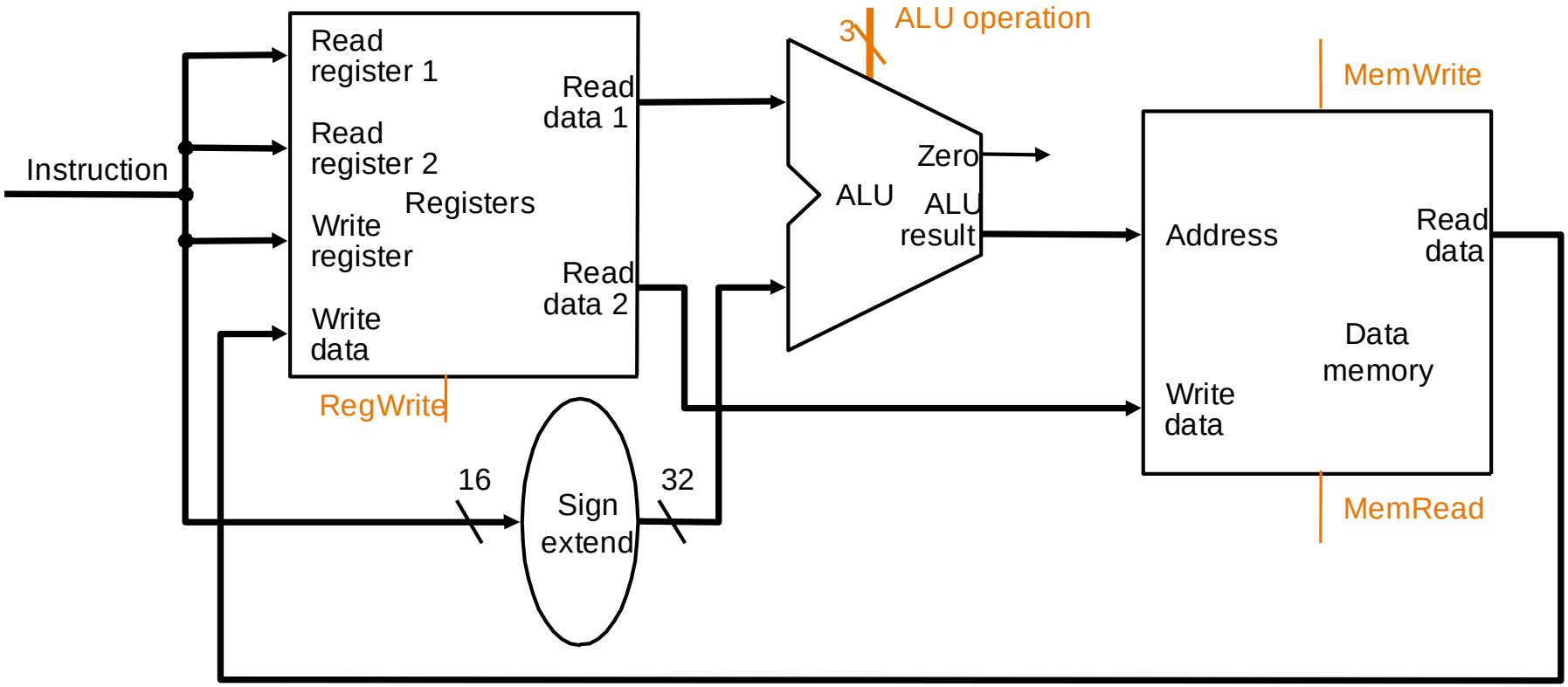
a. Data memory unit



b. Sign-extension unit

Via de Dados

(Parte da via de dados para a execução das instruções lw e sw)



Via de Dados

(Características das instruções beq)

- Utiliza o formato I-type

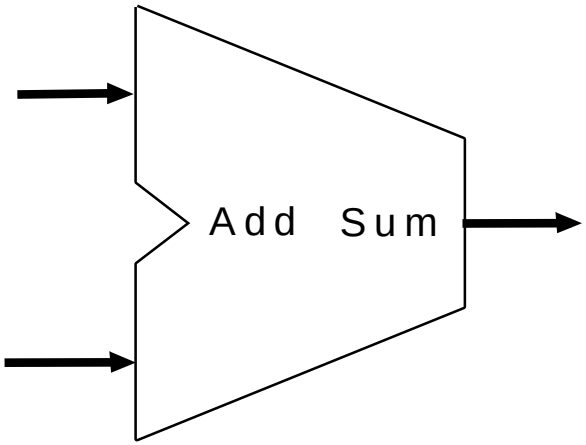
- **Instância da instrução beq:**

  beq rs, rt, offset // if rs = = rt go to PC+4+offset*4

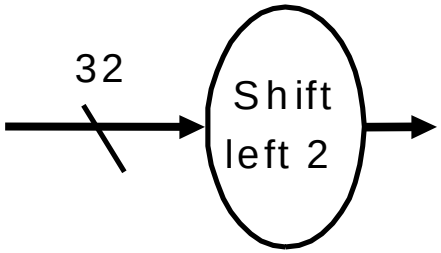
- **Requisitos para execução:**
  - Ler o conteúdo de dois registradores (rs e rt);
  - Realizar a soma do PC +4 com o offset*4 utilizando uma ALU para calcular o endereço da memória;
  - Subtrair o conteúdo dos registradores utilizando a ALU para verificar se são iguais;
  - Se os valores dos registradores são iguais, o novo valor de PC será o endereço calculado (PC=PC+4+offset*4).

Via de Dados

(Elementos adicionais para a execução da instrução beq)



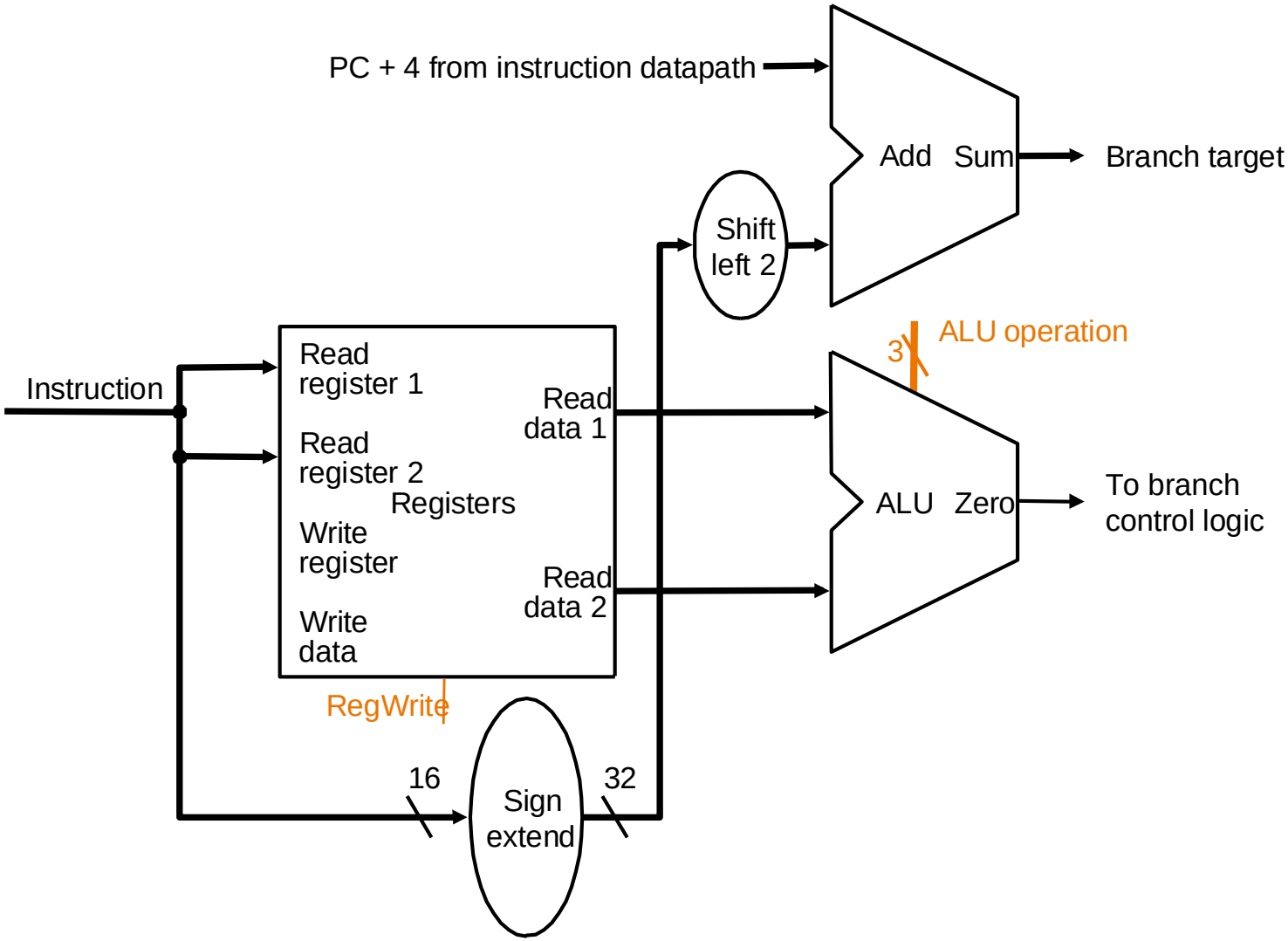
a. Adder



b. shift left 2

Via de Dados

(Via de dados para a execução da instrução beq)



Via de Dados

(Caracteristicas da via de dados)

•Executa toda instrução em um ciclo de clock;

•Inclui multiplexadores para compartilhar unidades funcionais entre diferentes classes de instruções:

•Detecta as diferenças chaves entre a via de dados para executar cada uma das classes de instruções;

Via de Dados

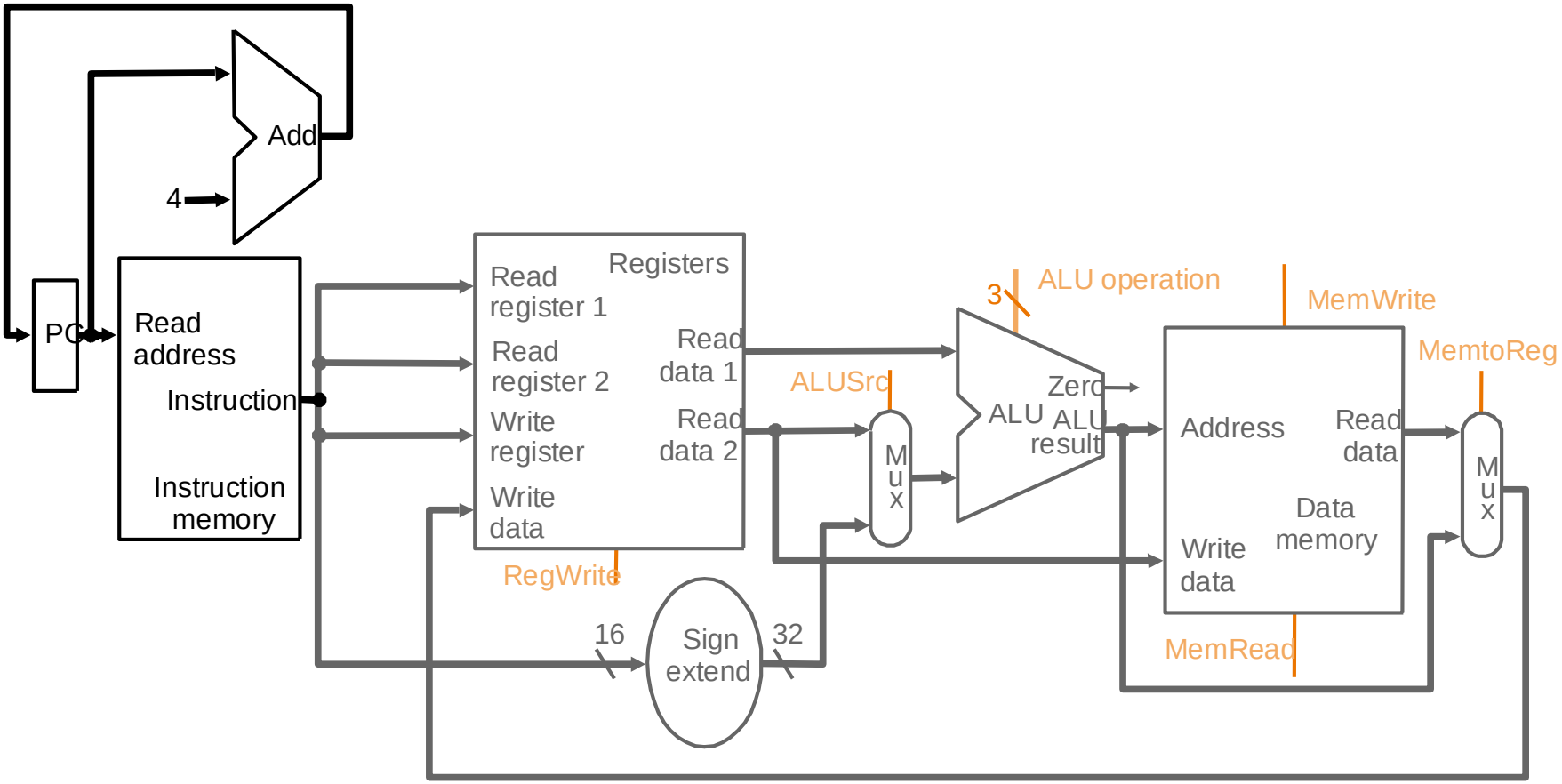
(Diferenças)

•Diferenças entre a via de dados para executar as instruções tipo R e a via de dados para executar as instruções de acesso a memória:

- •A segunda entrada da ALU ou é um registrador (instrução tipo R) ou o offset sing-extend (instrução de acesso a memória);
- •O valor armazenado dentro do registrador destino ou vem da ALU (instrução tipo R) ou da memória (lw);

Via de Dados

(Uma via de dados única e simples)



Via de Dados

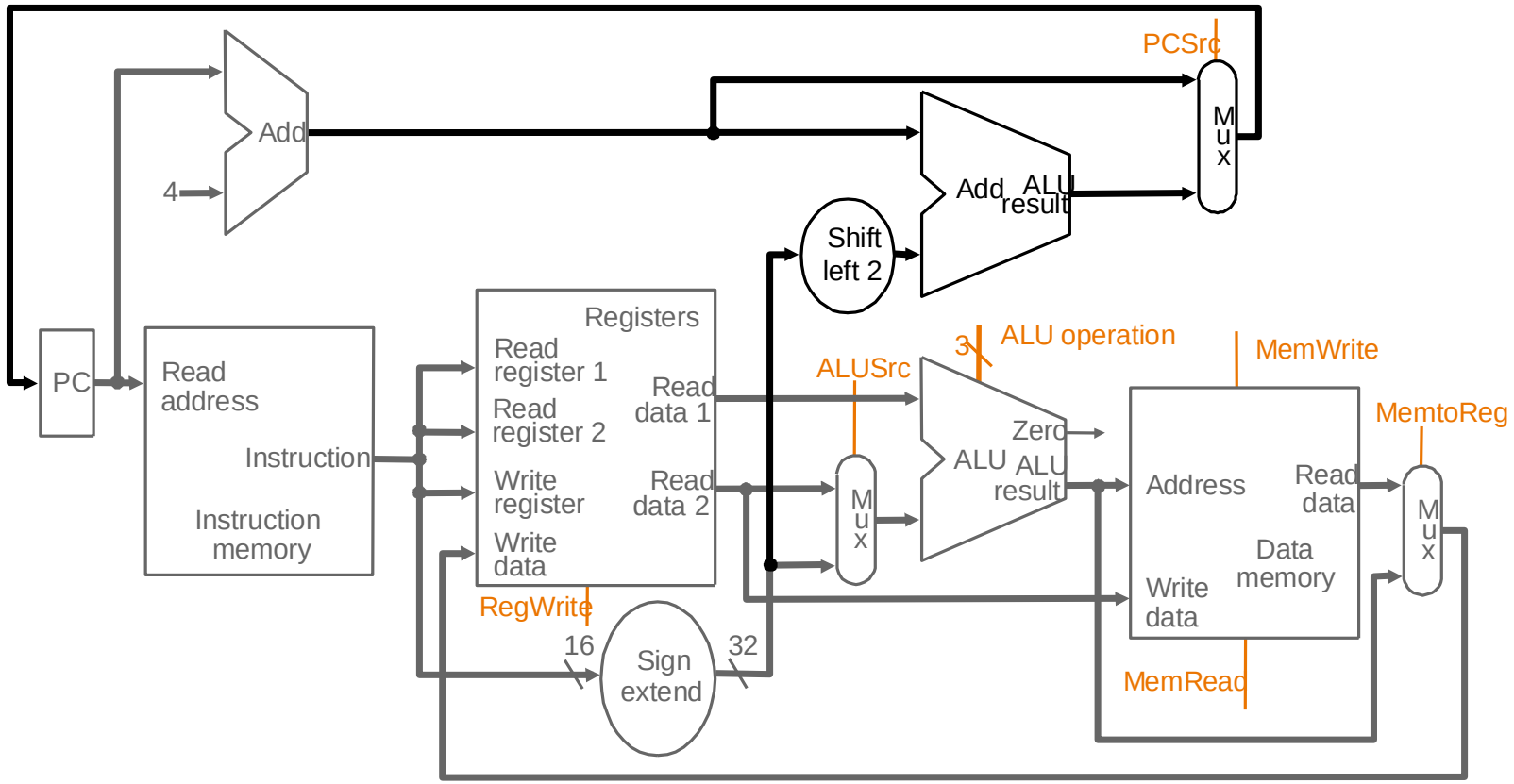
(Diferenças)

•Diferenças chaves entre a via de dados para executar a instrução beq e a via de dados para executar a busca de instruções e incrementar o PC:

- •O novo valor de PC ou é o valor calculado pelo somador de PC +4 ou o valor calculado por outro somador que executa a operação PC+4+ endereço*4;

Via de Dados

(Via de dados única e simples)



Unidade de Controle

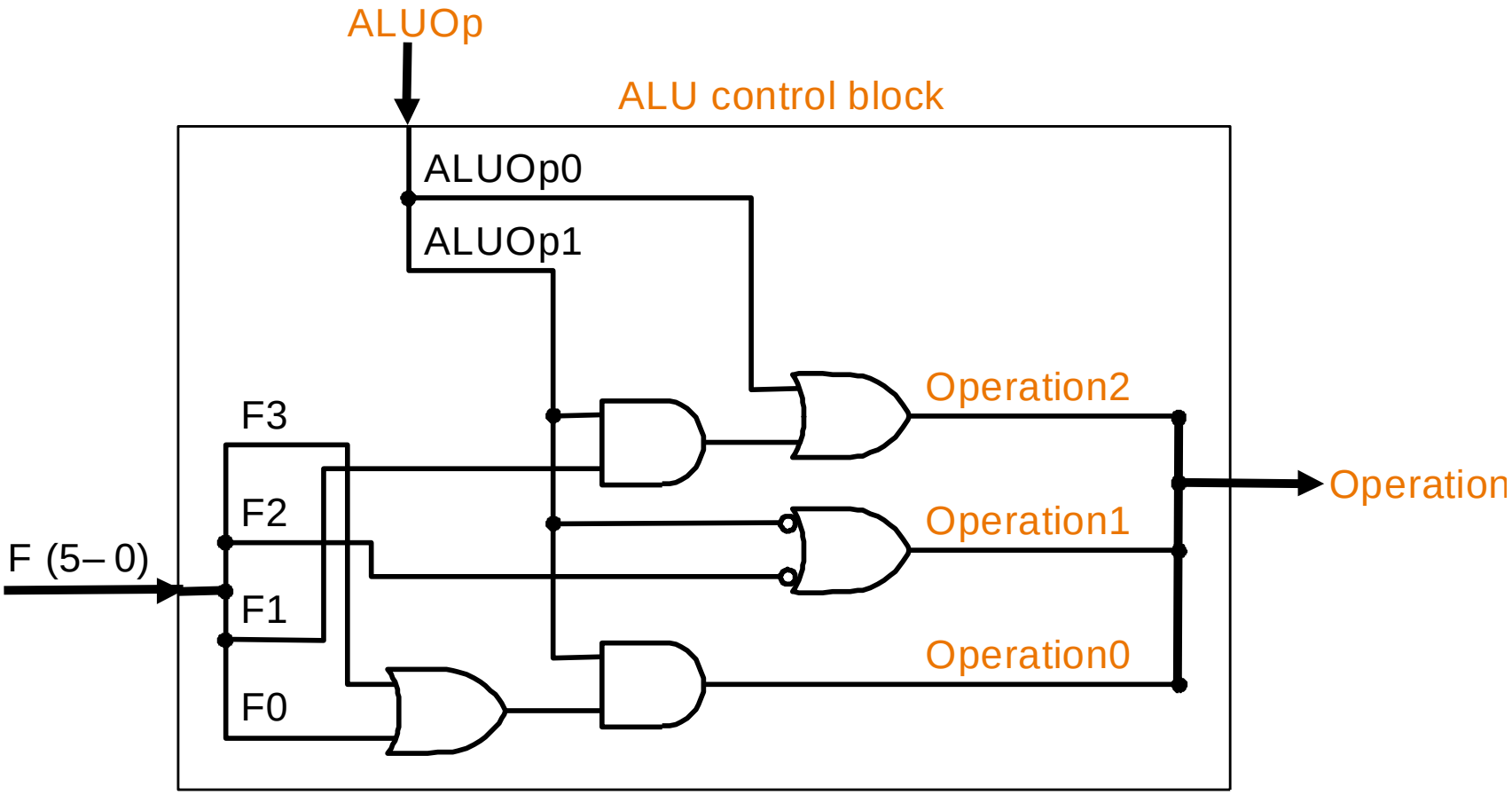
(sinais de controle da ALU)

sinais de controle	operação
000	AND
001	OR
010	add
110	sub
111	set on less than

instruction	ALUOp	função	ação da ALU
lw e sw	00	xxxxxx	add (010)
beq	01	xxxxxx	sub (110)
R-type	10	100000(32)	add (010)
R-type	10	100010(34)	sub (110)
R-type	10	100100(36)	and (000)
R-type	10	100101(37)	or (001)
R-Type	10	101010(42)	set on less than (111)

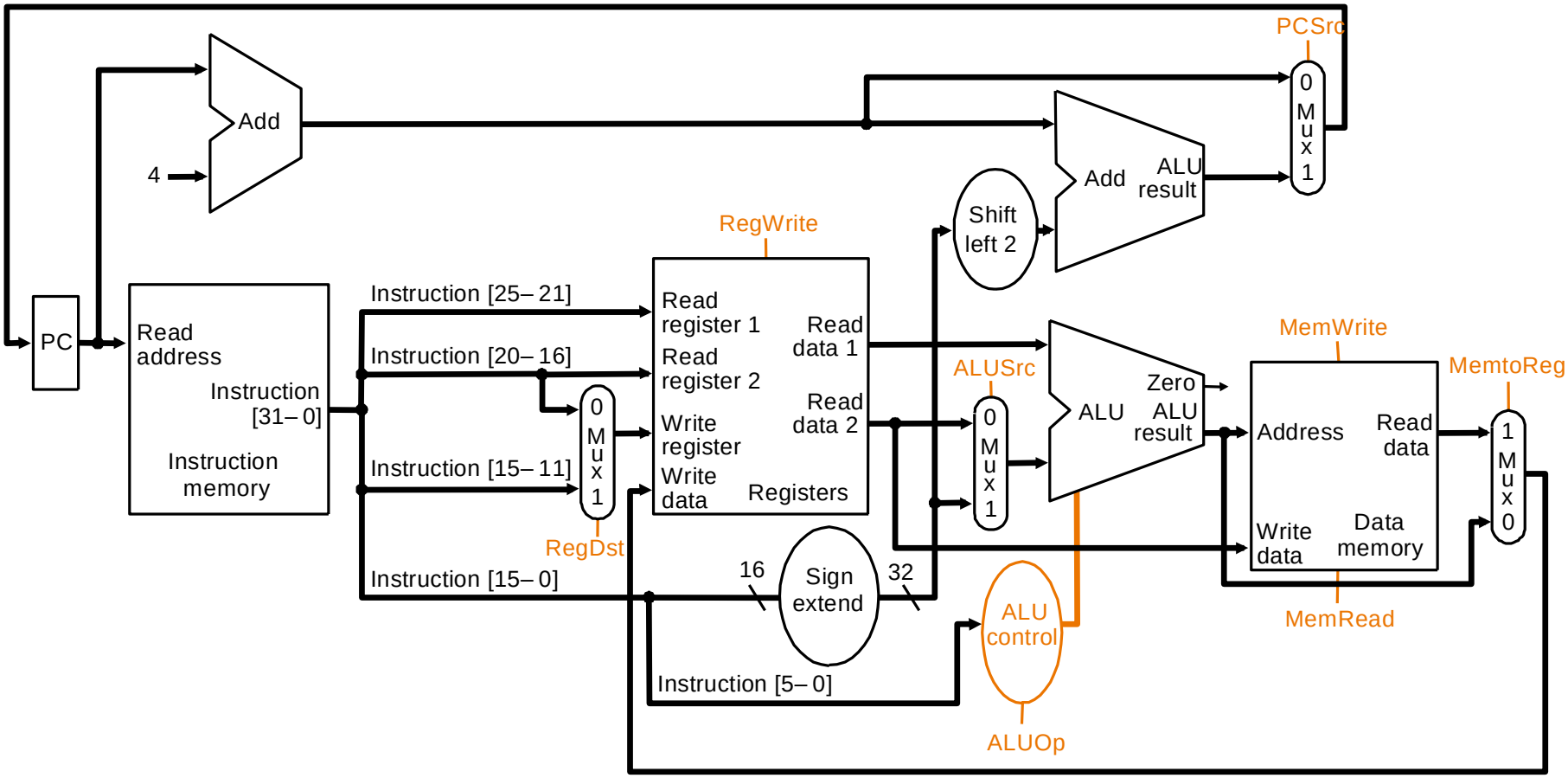
Unidade de Controle

(unidade de controle da ALU)



Unidade de Controle

(via de dados simples com a unidade de controle da ALU e os sinais de controle necessários)



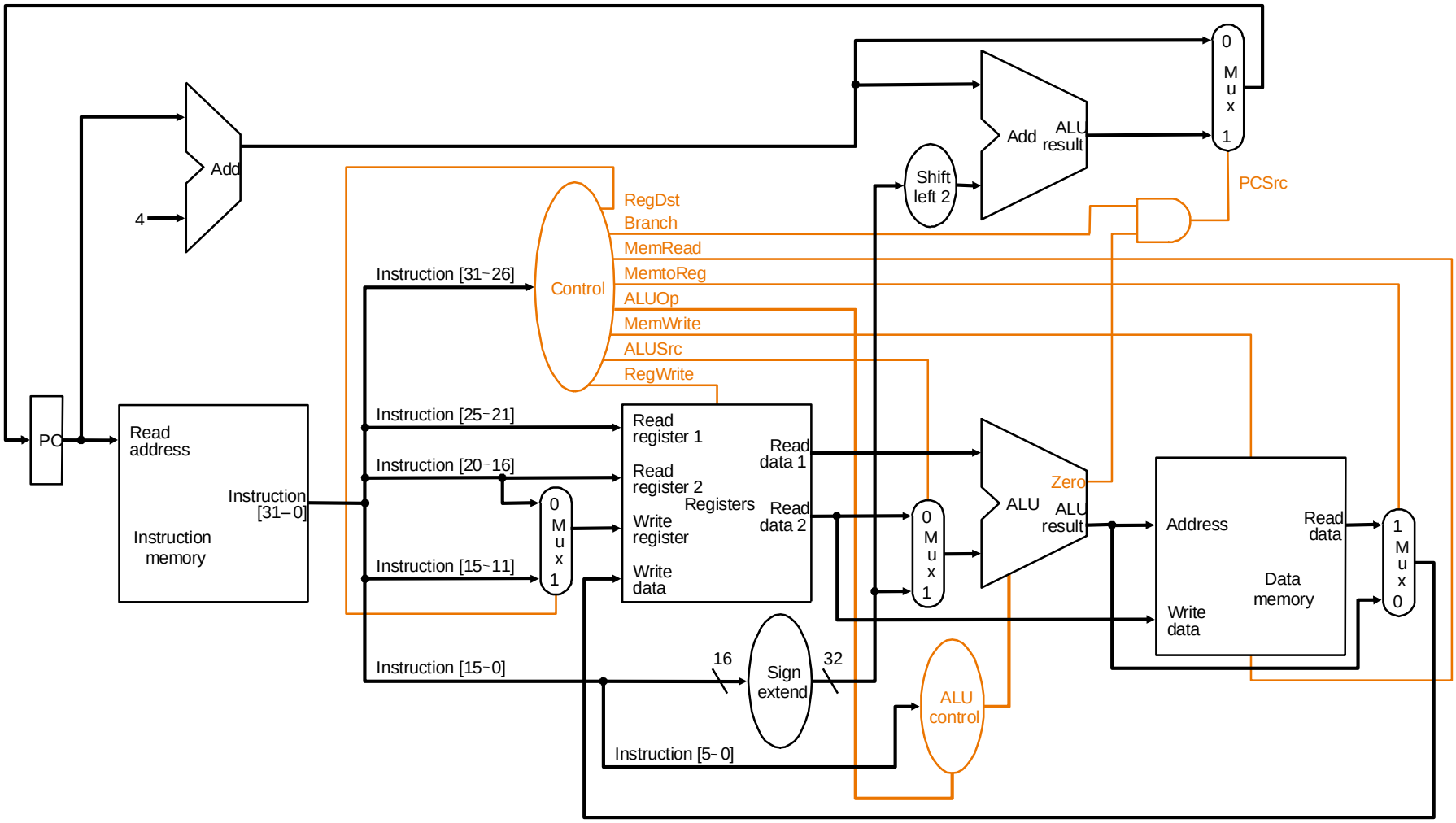
Unidade de Controle

(papel de cada sinal de controle da via de dados)

Nome do sinal	Efeito qdo *deasserted*(0)	Efeito qdo *asserted* (1)
RegDst	O n° do reg. dest. p/ a escrita vem do campo rt (bits 20-16)	O n° do reg. dest. p/ a escrita vem do campo rd (bits 15-11)
RegWrite	None	O dado de entrada do *write data é* escrito no *write register*
ALUSrc	O seg. operando da ALU vem da seg. saída do *register file*	O seg. operando da ALU é o *sign-extended*, 16 bit de mais baixa ordem da instrução
MemRead	None	O conteúdo do endereço da memória é colocado no *read data*
MemWrite	None	O conteúdo do endereço da memória é substituído pelo valor do *write data*
MemtoReg	O valor que será escrito no *write register* vem da ALU	O valor que será escrito no *write register* vem da memória
PCSrc	O PC é substituído pela saída do *adder* que calcula o valor de PC+4	O PC é substituído pela saída do *adder* que calcula o valor do endereço de desvio

Unidade de Controle

(via de dados simples com a unidade de controle principal)



Unidade de Controle

(linhas de controle determinadas pelo *opcode* da instrução)

Instruction	RegDst	ALU Src	Mem-toReg	Reg Write	Mem Read	Mem Write	Branch	ALUOp1	ALUOp0
R_format	1	0	0	1	0	0	0	1	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	X	0	0	0	1	0	1

Unidade de Controle

Execução de Operações na Via de Dados
(passos para a execução de uma soma)

add $t1, $t2, $t3

➔Busca a instrução da memória e incrementa o PC;

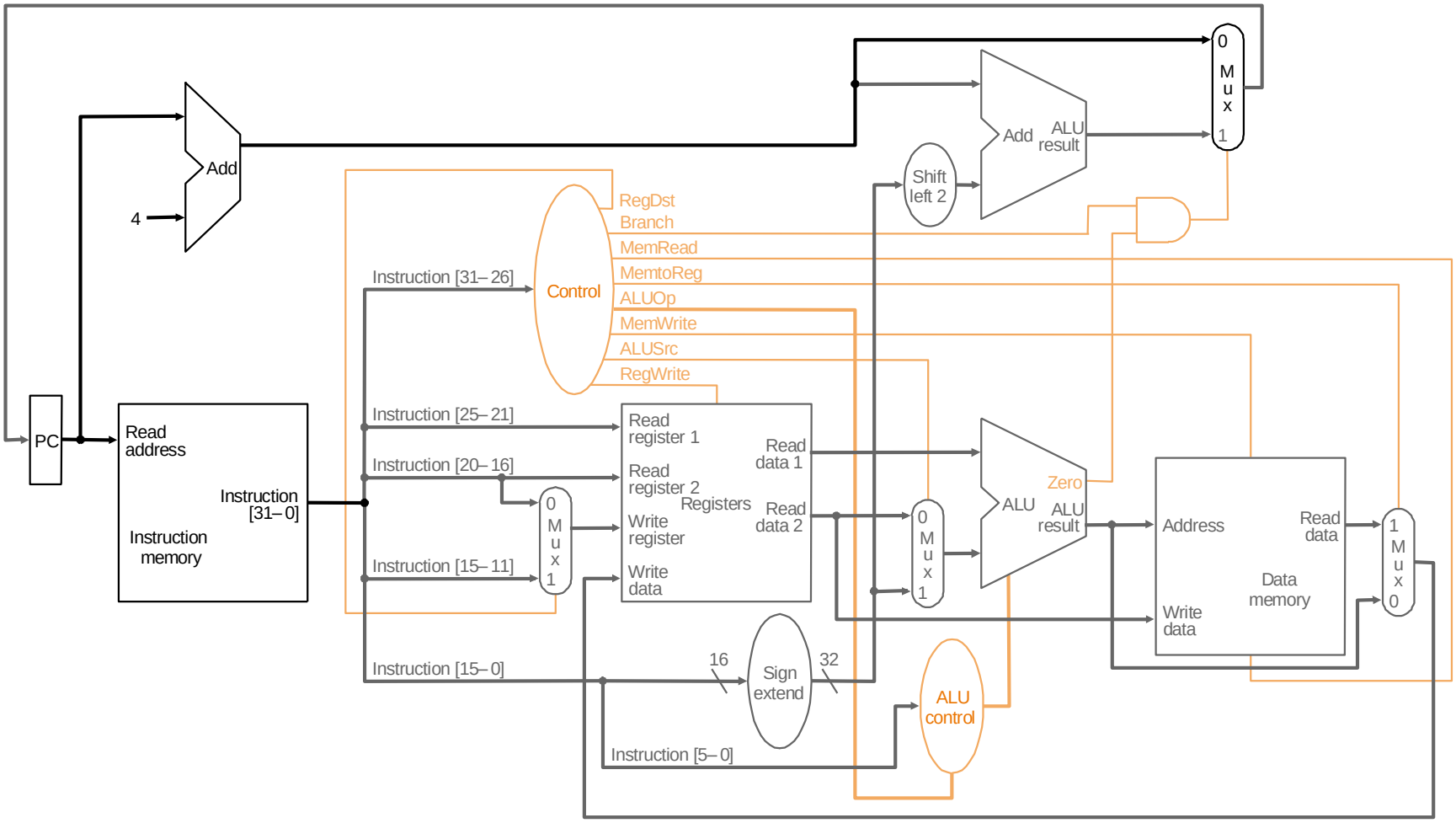
•Dois registradores, $t2, $t3, são lidos do *register file* e a unidade de controle principal calcula os valores dos sinais de controles;

•A ALU, utiliza o valor do campo *funct* para selecionar sua função, faz uma operação com os dados lidos do r*egister file;*

•O resultado da ALU é escrito no *register file*, usando os bits 15 –11 para selecionar o registrador destino.

Unidade de Controle

Execução de Operações na Via de Dados
(execução de uma soma, 1° passo)



Unidade de Controle

Execução de Operações na Via de Dados
(passos para a execução de uma soma)

•Busca a instrução da memória e incrementa o PC;

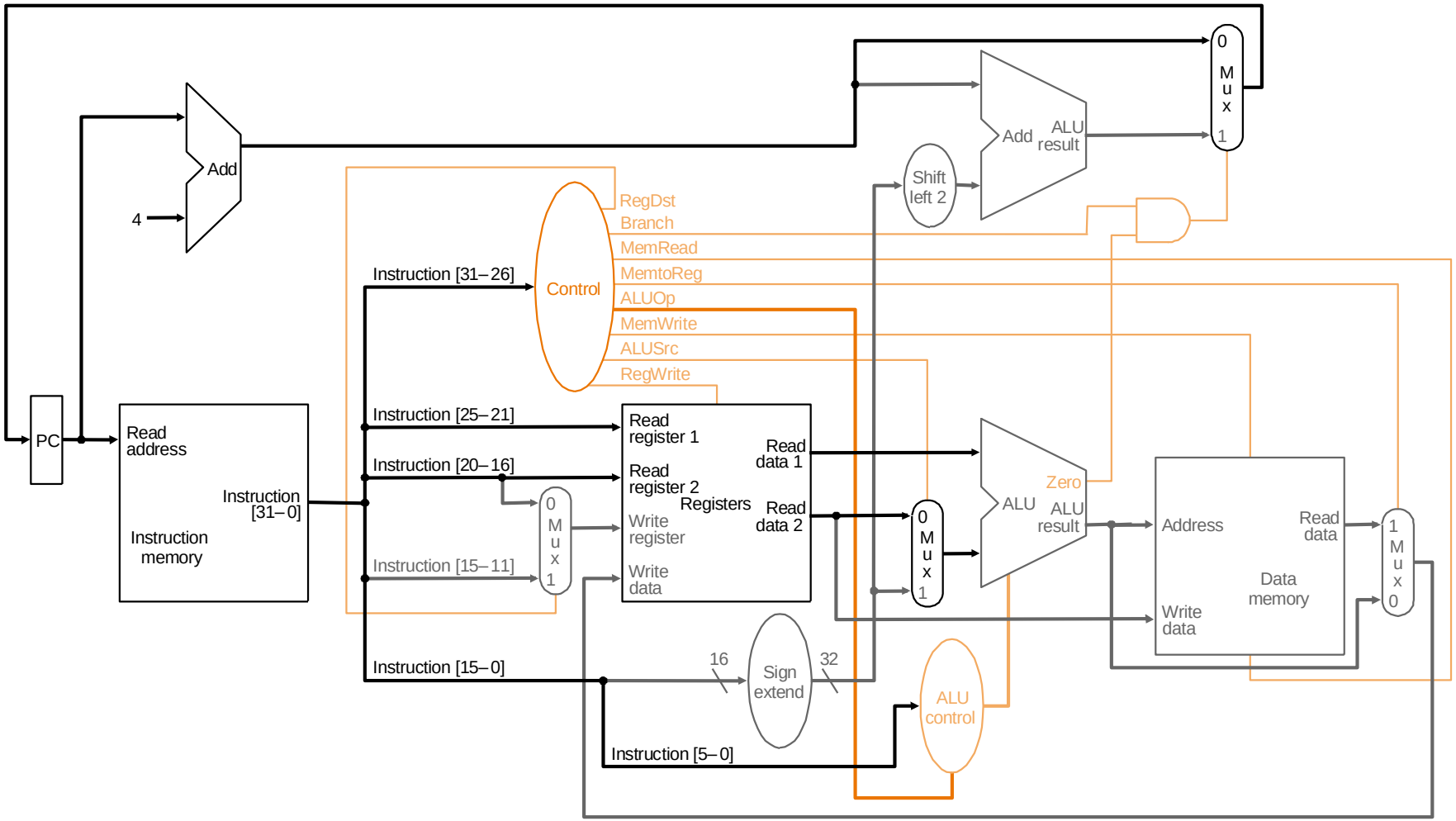
➔Dois registradores, $t2, $t3, são lidos do *register file* e a unidade de controle principal calcula os valores dos sinais de controles;

•A ALU, utilizando o valor do campo *funct* para selecionar sua função, faz uma operação com os dados lidos do r*egister file;*

•O resultado da ALU é escrito no *register file*, usando os bits 15 –11 para selecionar o registrador destino.

Unidade de Controle

Execução de Operações na Via de Dados (execução de uma soma, 2° passo)



Unidade de Controle

Execução de Operações na Via de Dados
(passos para a execução de uma soma)

•Busca a instrução da memória e incrementa o PC;

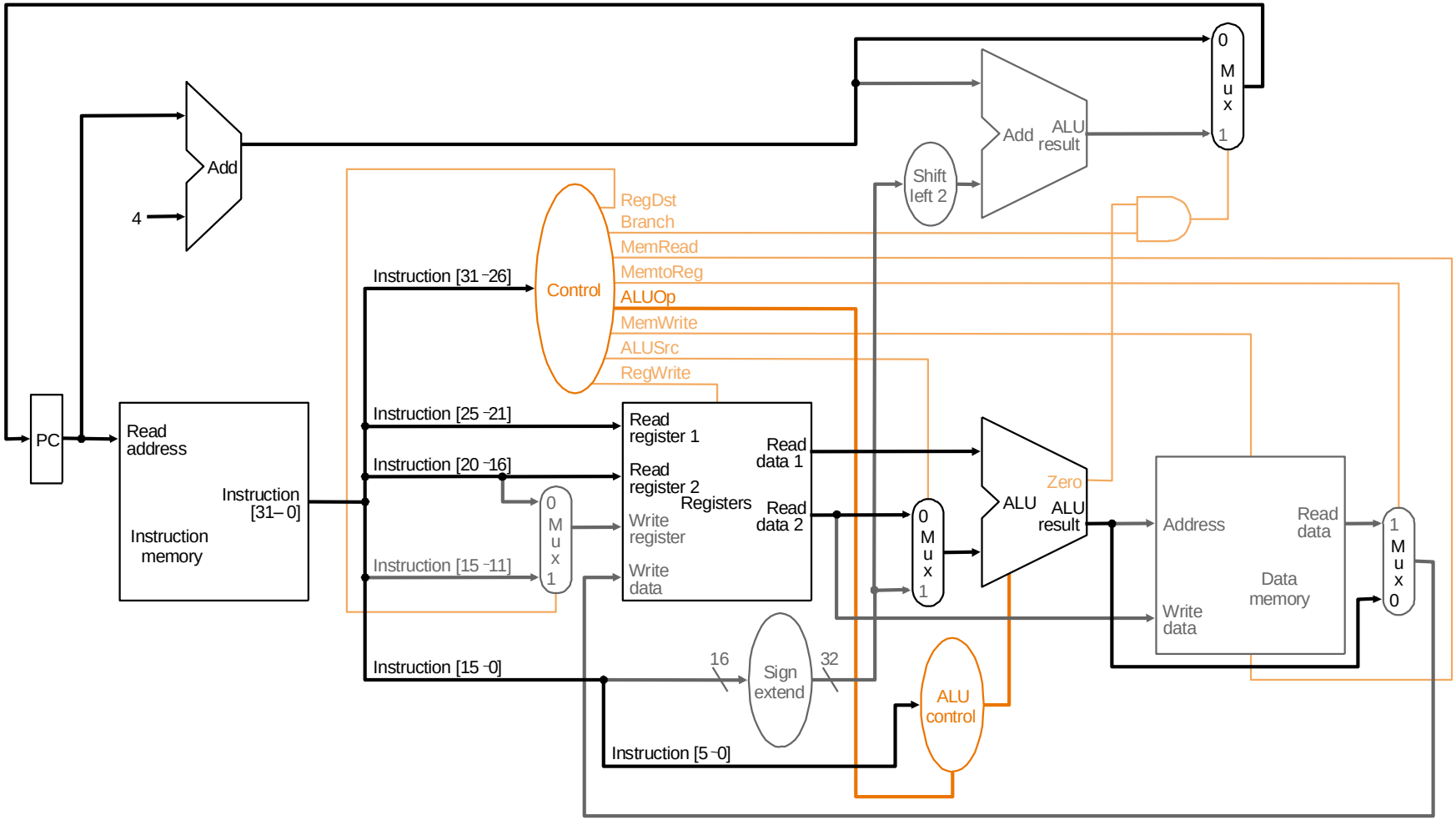
•Dois registradores, $t2, $t3, são lidos do *register file* e a unidade de controle principal calcula os valores dos sinais de controles;

➔•A ALU, utiliza o valor do campo *funct* para selecionar sua função, faz uma operação com os dados lidos do r*egister file;*

•O resultado da ALU é escrito no *register file*, usando os bits 15 –11 para selecionar o registrador destino.

Unidade de Controle

Execução de Operações na Via de Dados (execução de uma soma, 3° passo)



Unidade de Controle

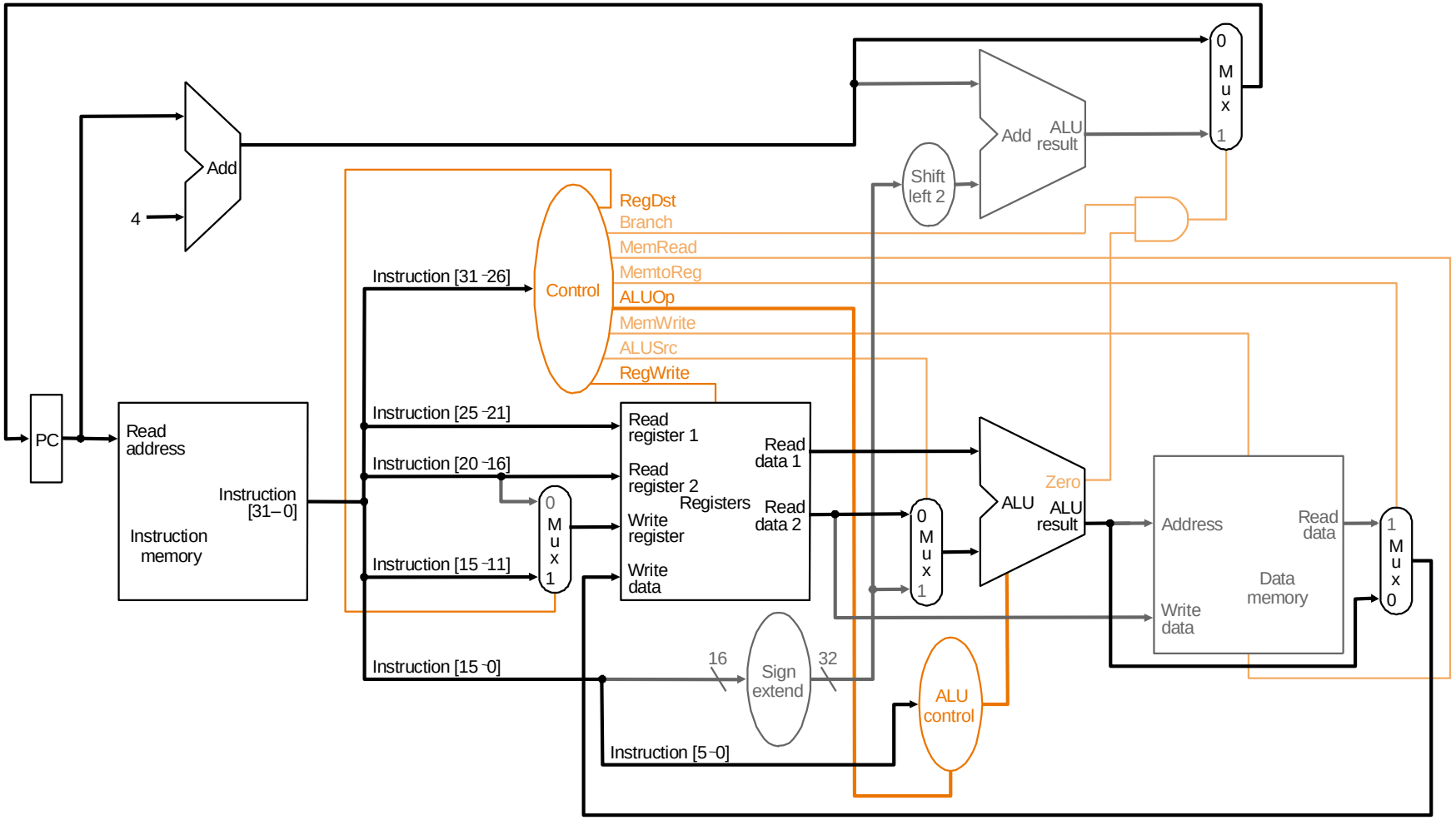
Execução de Operações na Via de Dados
(passos para a execução de uma soma)

- Busca a instrução da memória e incrementa o PC;
- Dois registradores, $t2, $t3, são lidos do *register file* e a unidade de controle principal calcula os valores dos sinais de controles;
- A ALU, utiliza o valor do campo *funct* para selecionar sua função, faz uma operação com os dados lidos do r*egister file;*

➔ O resultado da ALU é escrito no *register file*, usando os bits 15 –11 para selecionar o registrador destino.

Unidade de Controle

Execução de Operações na Via de Dados (execução de uma soma, 4° passo)



Unidade de Controle

Execução de Operações na Via de Dados
(passos para a execução de um lw)

lw $t1, offset($t2)

➔Busca a instrução da memória e incrementa o PC;

•Um registrador ($t2) é lido do *register file* e a unidade de controle principal calcula os valores dos sinais de controles;

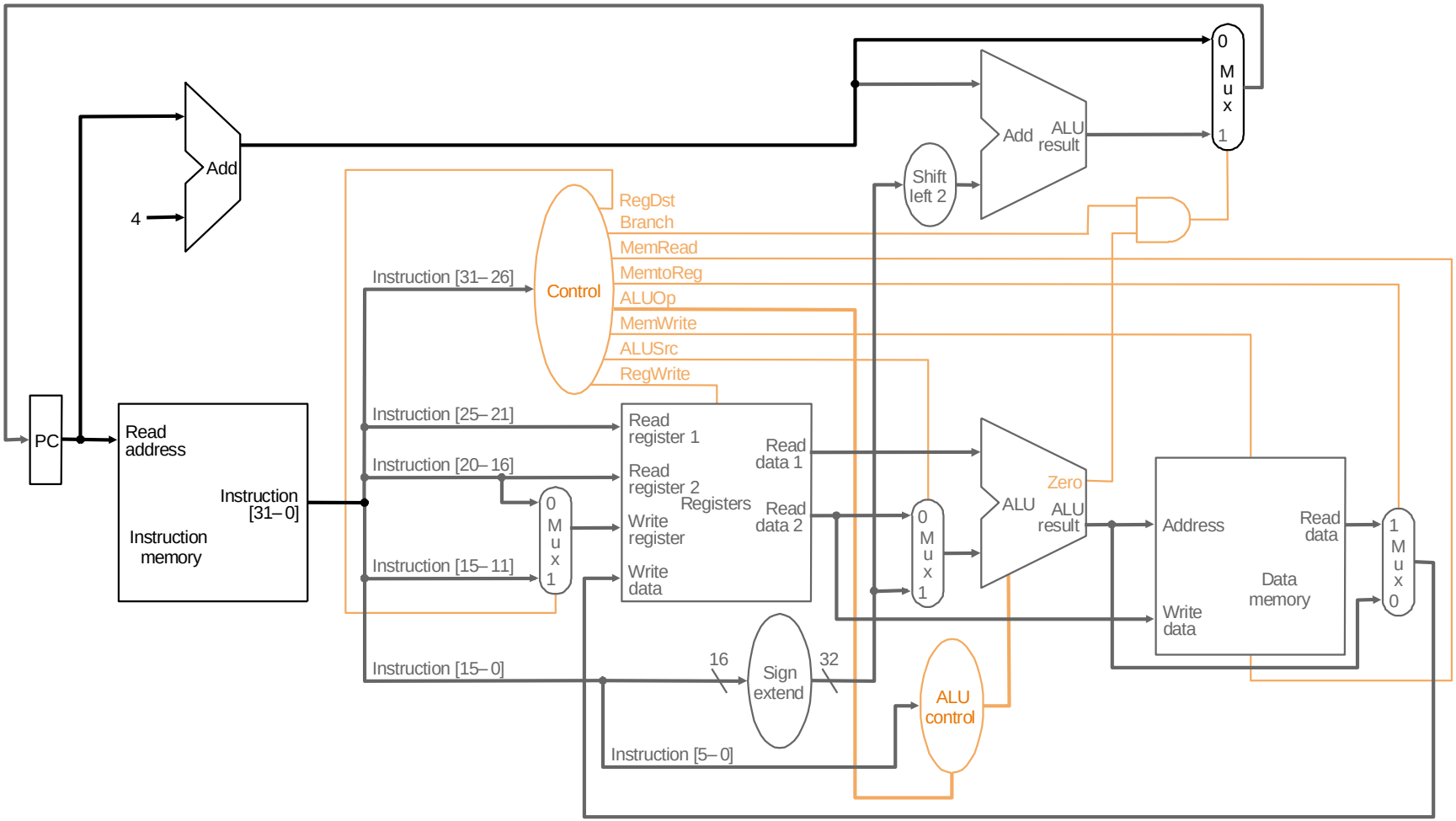
•A ALU, realiza a soma do valor lido do *register file* com os 16 bits do offset*;*

•A resultado da soma realizada pela ALU é usada como o endereço da memória de dados;

•O valor lido da memória é escrito dentro de *register file;* o registrador destino é dado pelos bits 20-16 da instrução ($t1).

Unidade de Controle

Execução de Operações na Via de Dados (execução de um lw, 1° passo)



Unidade de Controle

Execução de Operações na Via de Dados
(passos para a execução de um lw)

•Busca a instrução da memória e incrementa o PC;

➔Um registrador ($t2) é lido do *register file* e a unidade de controle principal calcula os valores das linhas de controles;

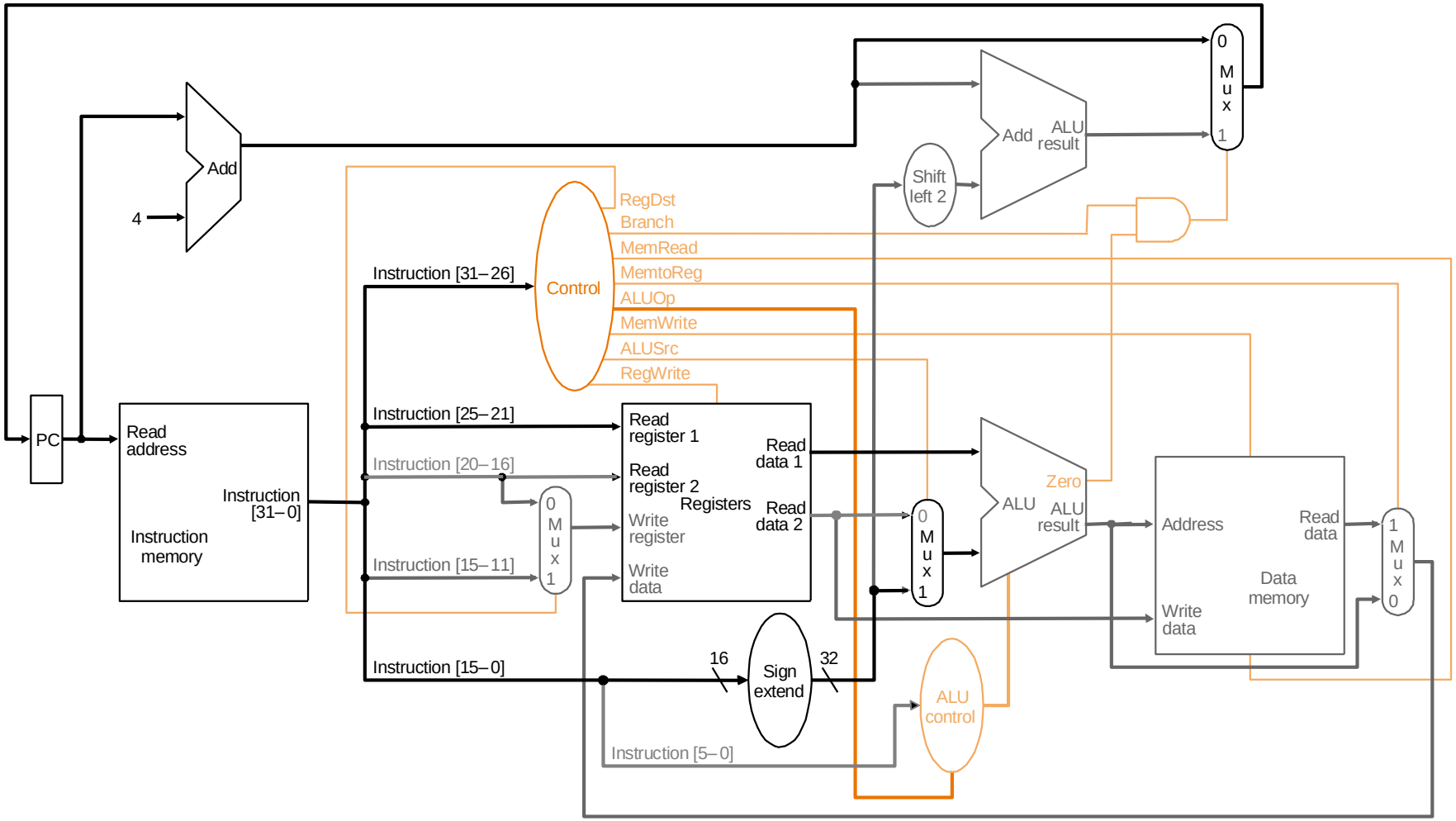
•A ALU, realiza a soma do valor lido do *register file* com os 16 bits do offset*;*

•A resultado da soma realizada pela ALU é usada como o endereço da memória de dados;

•O valor lido da memória é escrito dentro de *register file;* o registrador destino é dado pelos bits 20-16 da instrução ($t1).

Unidade de Controle

Execução de Operações na Via de Dados (execução de um lw, 2° passo)



Unidade de Controle

Execução de Operações na Via de Dados
(passos para a execução de um lw)

•Busca a instrução da memória e incrementa o PC;

•Um registrador ($t2) é lido do *register file* e a unidade de controle principal calcula os valores das linhas de controles;

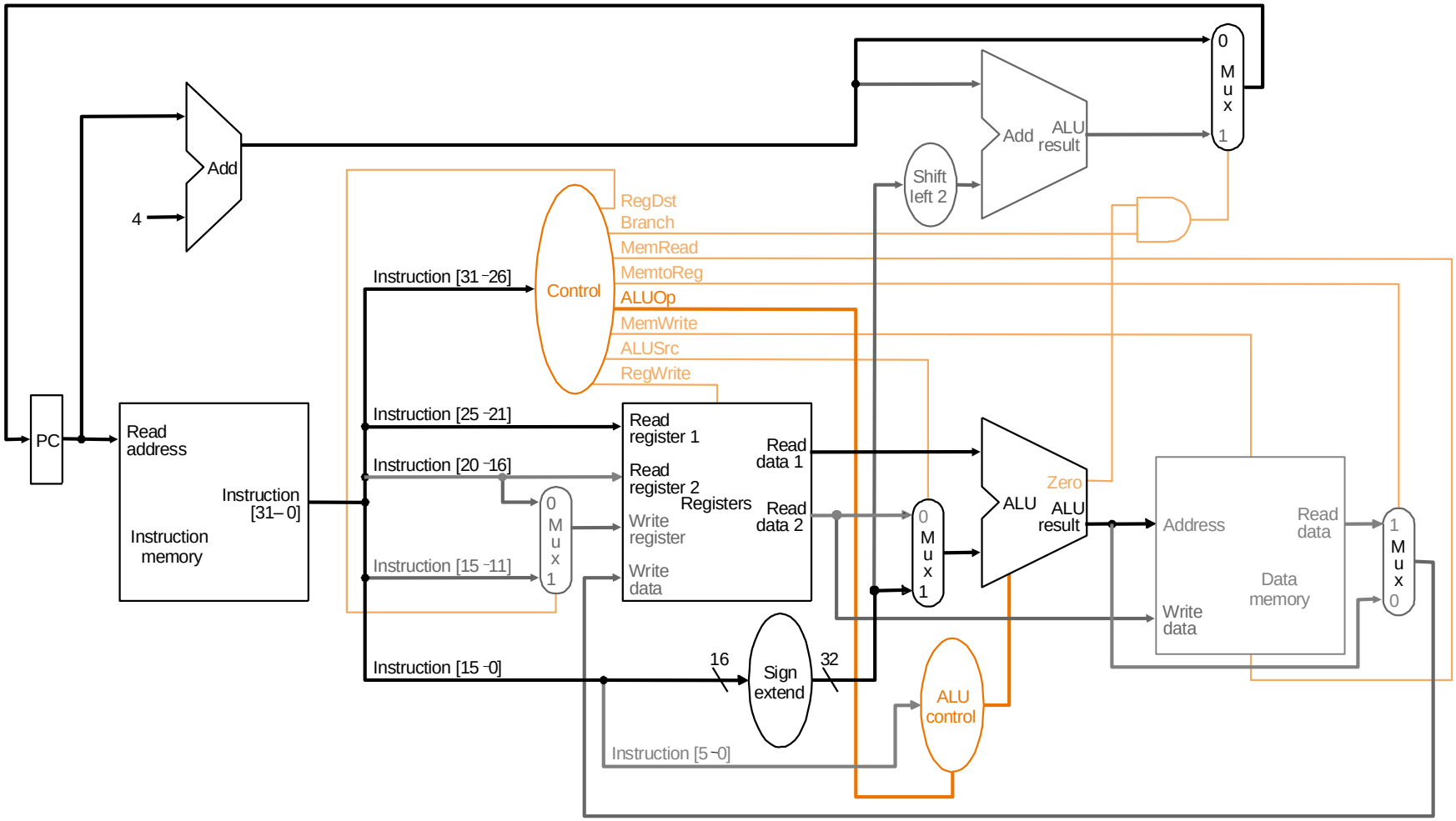
➔A ALU, realiza a soma do valor lido do *register file* com os 16 bits do offset*;*

•A resultado da soma realizada pela ALU é usada como o endereço da memória de dados;

•O valor lido da memória é escrito dentro de *register file;* o registrador destino é dado pelos bits 20-16 da instrução ($t1).

Unidade de Controle

Execução de Operações na Via de Dados (execução de um lw, 3° passo)



Unidade de Controle

Execução de Operações na Via de Dados
(passos para a execução de um lw)

•Busca a instrução da memória e incrementa o PC;

•Um registrador ($t2) é lido do *register file* e a unidade de controle principal calcula os valores das linhas de controles;

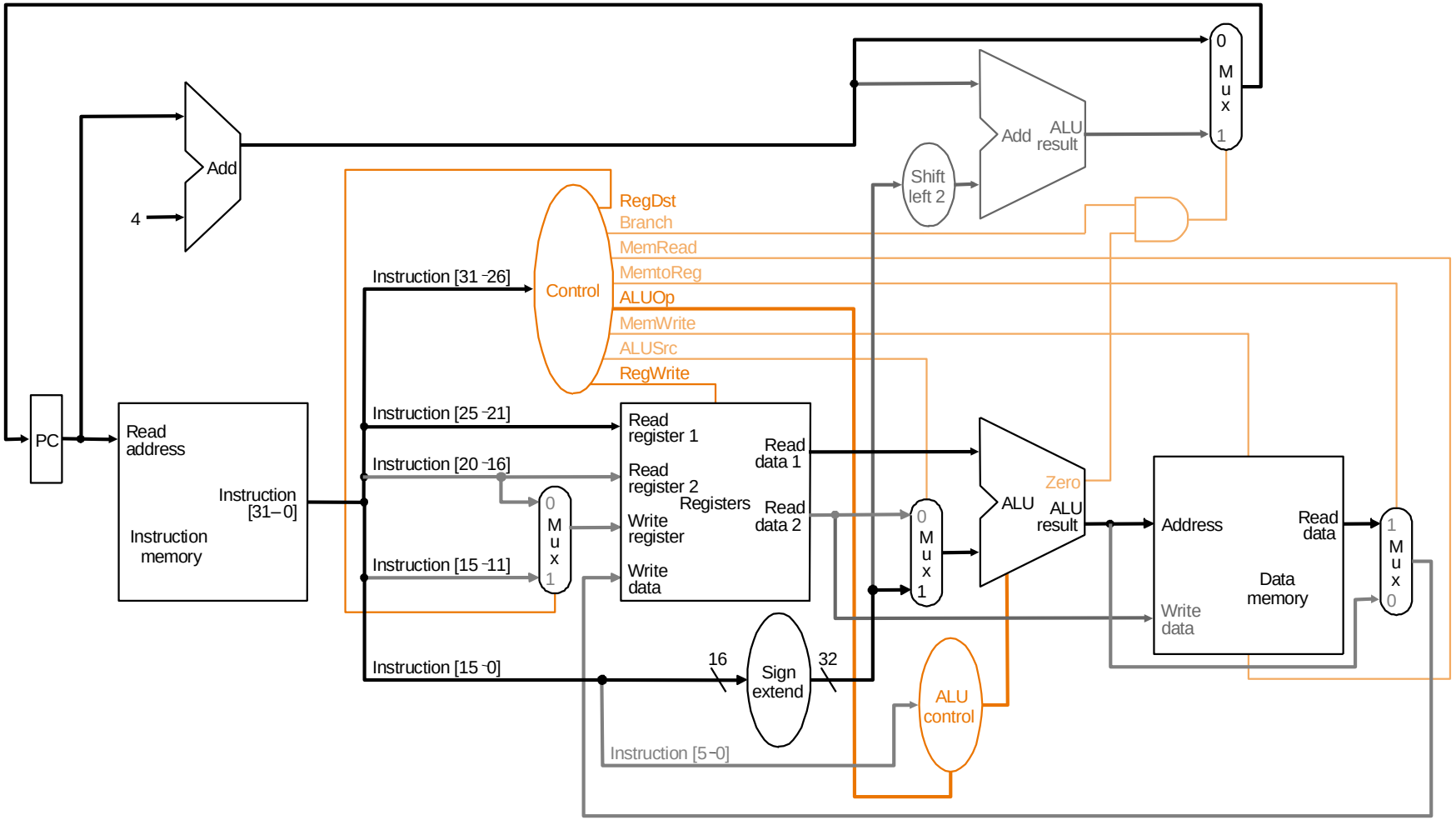
•A ALU, realiza a soma do valor lido do *register file* com os 16 bits do offset*;*

➔•A resultado da soma realizada pela ALU é usada como o endereço da memória de dados;

•O valor lido da memória é escrito dentro de *register file;* o registrador destino é dado pelos bits 20-16 da instrução ($t1).

Unidade de Controle

Execução de Operações na Via de Dados (execução de um lw, 4° passo)



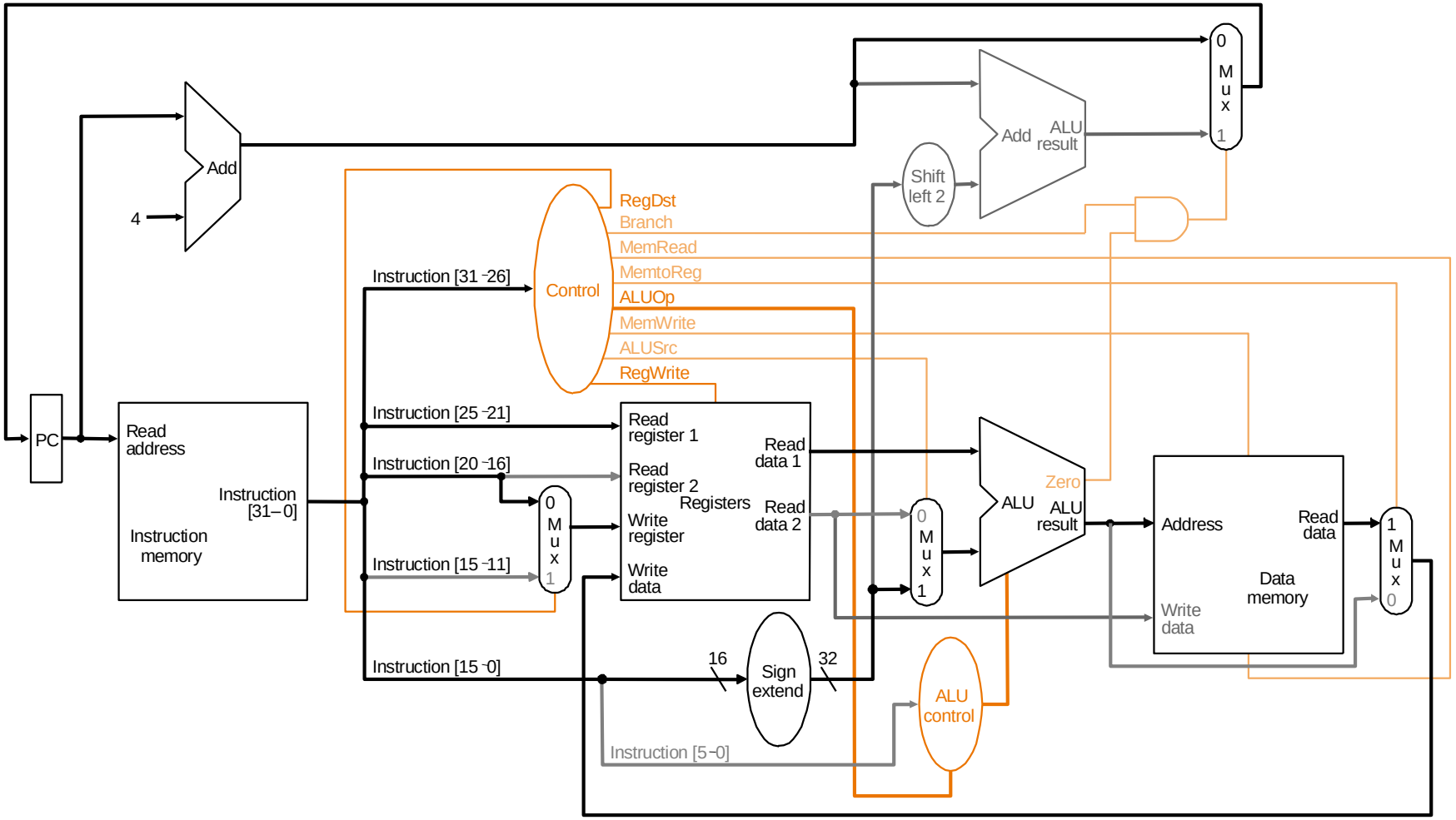
Unidade de Controle

Execução de Operações na Via de Dados
(passos para a execução de um lw)

- Busca a instrução da memória e incrementa o PC;
- Um registrador ($t2) é lido do *register file* e a unidade de controle principal calcula os valores das linhas de controles;
- A ALU, realiza a soma do valor lido do *register file* com os 16 bits do offset*;*
- A resultado da soma realizada pela ALU é usada como o endereço da memória de dados;
- ➔O valor lido da memória é escrito dentro de *register file;* o registrador destino é dado pelos bits 20-16 da instrução ($t1).

Unidade de Controle

Execução de Operações na Via de Dados (execução de um lw, 5° passo)



Unidade de Controle

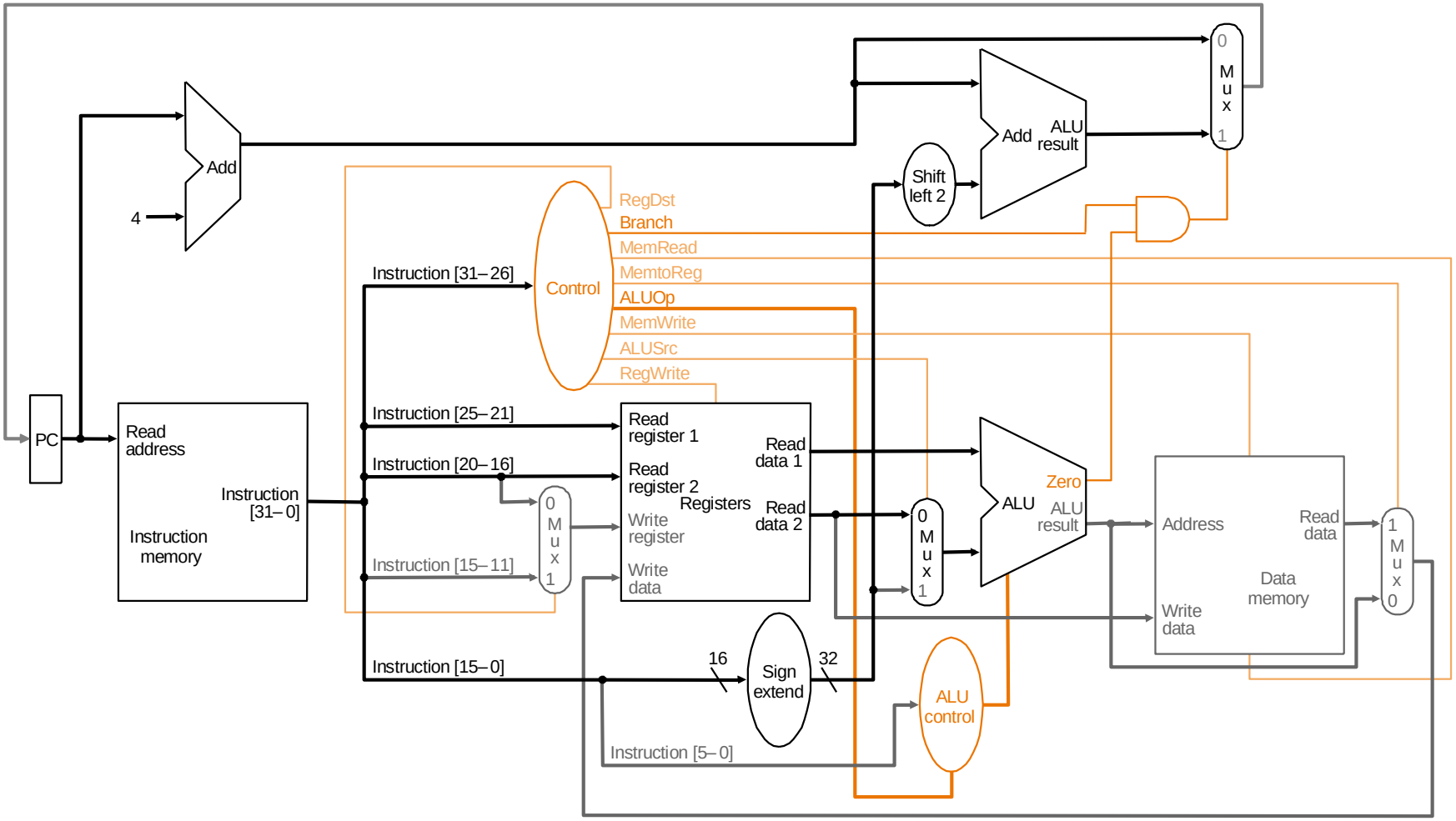
Execução de Operações na Via de Dados
(passos para a execução de um beq)

beq $t1, $t2, offset

- Busca a instrução da memória e incrementa o PC;
- Dois registradores, $t1, $t2, são lidos do *register file* e a unidade de controle principal calcula os valores das linhas de controles;
- ➔ A ALU faz uma operação de subtração com os dados lidos do r*egister file e* o valor de PC+4 é somado aos 16 bits de mais baixa ordem com sinal extendido e deslocamento de dois bits para a esquerda (o resultado é o end. de desvio)*;*
- O Zero do resultado da ALU é usado para decidir qual será o novo valor de PC (PC+ 4 ou endereço de desvio).

Unidade de Controle

Execução de Operações na Via de Dados (execução de um beq, 3° passo)



Unidade de Controle

Execução de Operações na Via de Dados
(passos para a execução de um beq)

•Busca a instrução da memória e incrementa o PC;

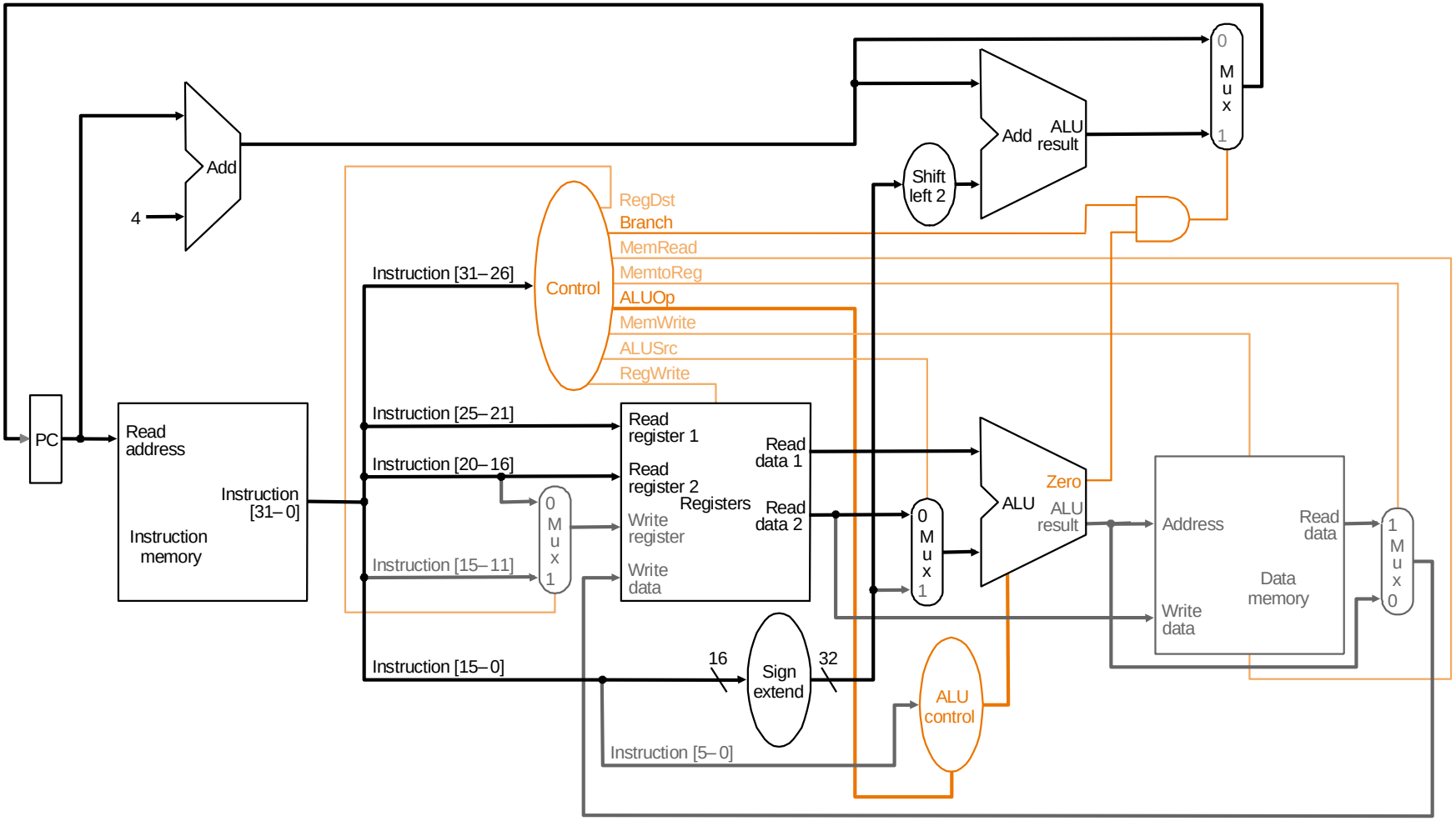
•Dois registradores, $t1, $t2, são lidos do *register file* e a unidade de controle principal calcula os valores das linhas de controles;

•A ALU faz uma operação de subtração com os dados lidos do r*egister file e* o valor de PC+4 é somado aos 16 bits de mais baixa ordem com sinal extendido e deslocamento de dois bits para a esquerda (o resultado é o end. de desvio)*;*

➔O Zero do resultado da ALU é usado para decidir qual será o novo valor de PC (PC+ 4 ou endereço de desvio).

Unidade de Controle

Execução de Operações na Via de Dados (execução de um beq, 4° passo)



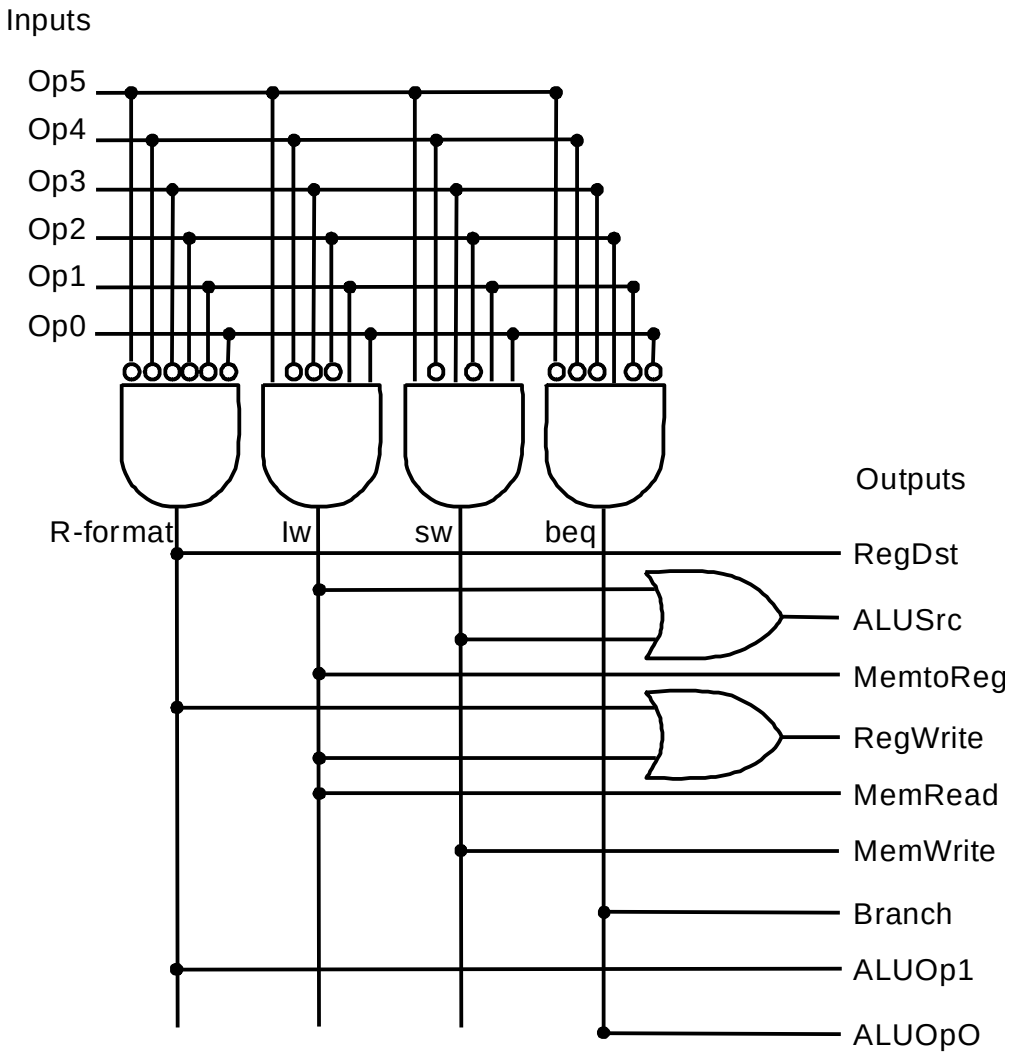
Unidade de Controle

(função da unidade de controle)

Input or Output	Signal name	R-format	Lw	Sw	Beq
Inputs	Op5	0	1	1	0
	Op4	0	0	0	0
	Op3	0	0	1	0
	Op2	0	0	0	1
	Op1	0	1	1	0
	Op0	0	1	1	0
Output	RegDst	1	0	X	X
	ALUSrc	0	1	1	0
	MemtoReg	0	1	X	X
	RegWrite	1	1	0	0
	MemRead	0	1	0	0
	MemWrite	0	0	1	0
	Branch	0	0	0	1
	ALUOp1	1	0	0	0
	ALUOp0	0	0	0	1

Unidade de Controle

(implementação da via de dados)

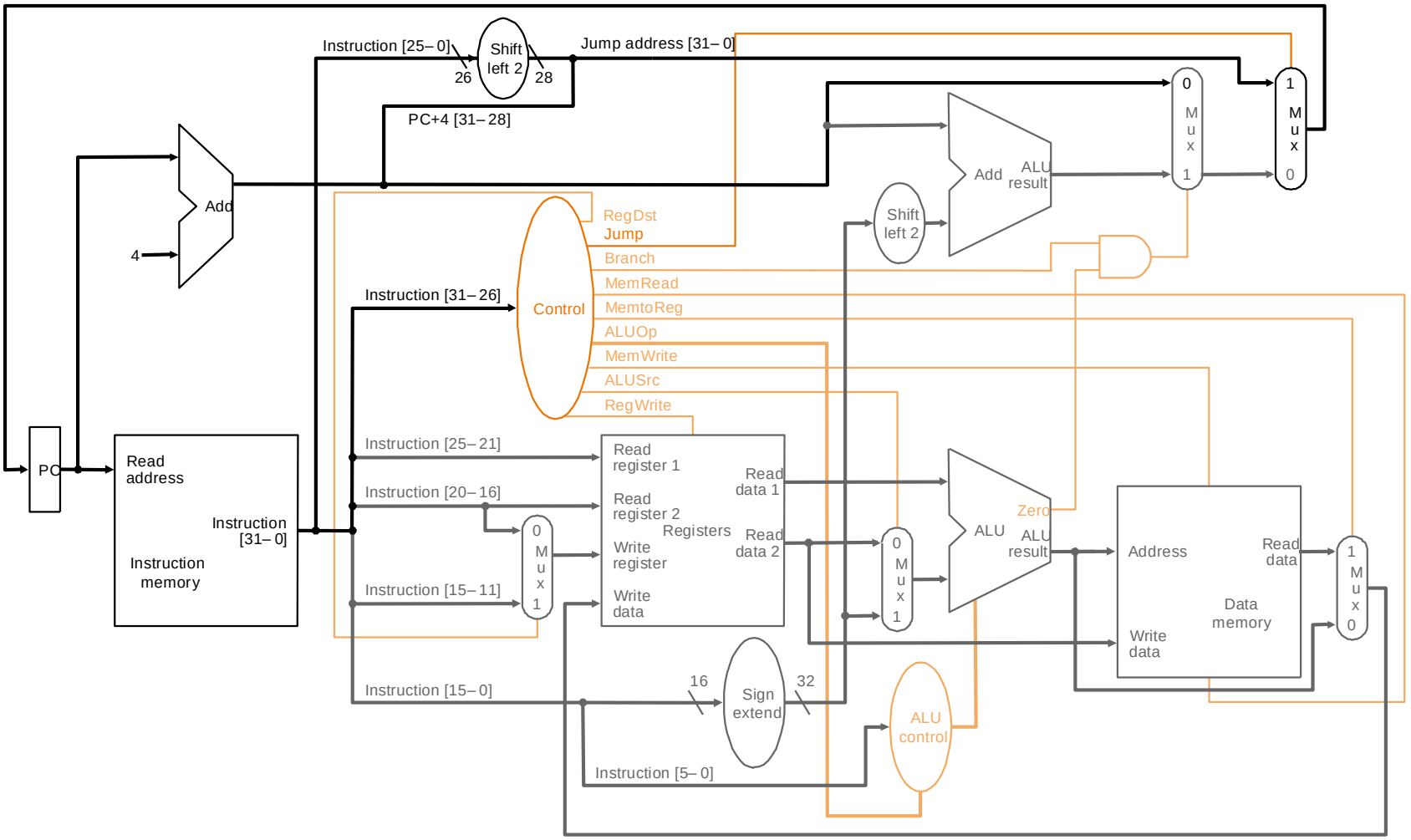


Unidade de Controle

(via de dados para execução da instrução jump)

J-type

2	Adress
31-26	25-0



Porque a implementação em um único ciclo de clock é inviável?

Lista de Exercícios