



Universidade de Brasília

Departamento de Ciência da Computação

Aula 17

Implementação MIPS

MIPS Pipeline – Unidade Operativa e
Controle



Controle do pipeline

- Temos 5 estágios.

O que deve ser controlado em cada estágio?

IF: *Instruction Fetch e PC Increment*

ID: *Instruction Decode / Register Fetch*

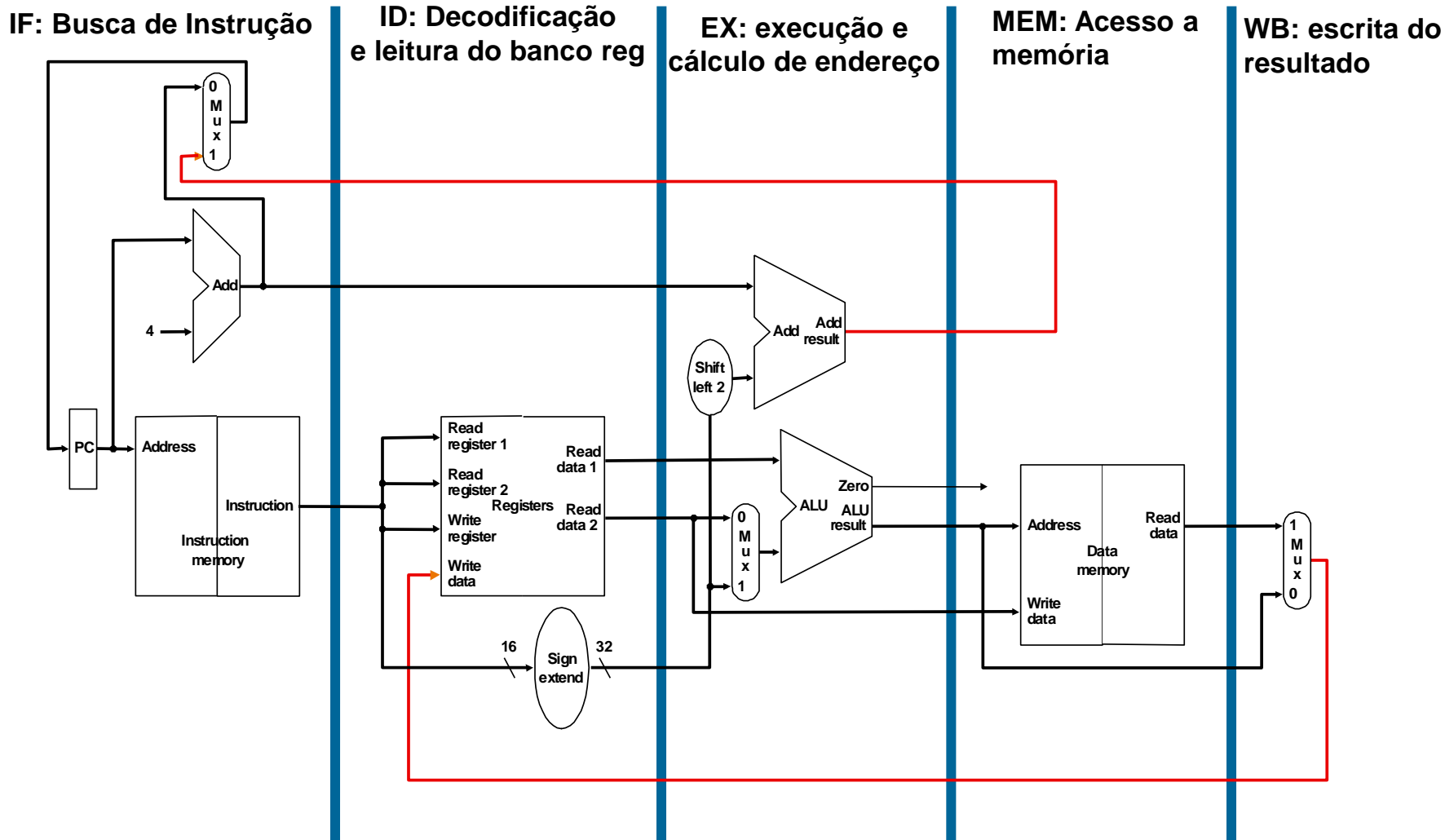
EX: *Execution*

MEM: *Memory Stage*

WB: *Write Back*



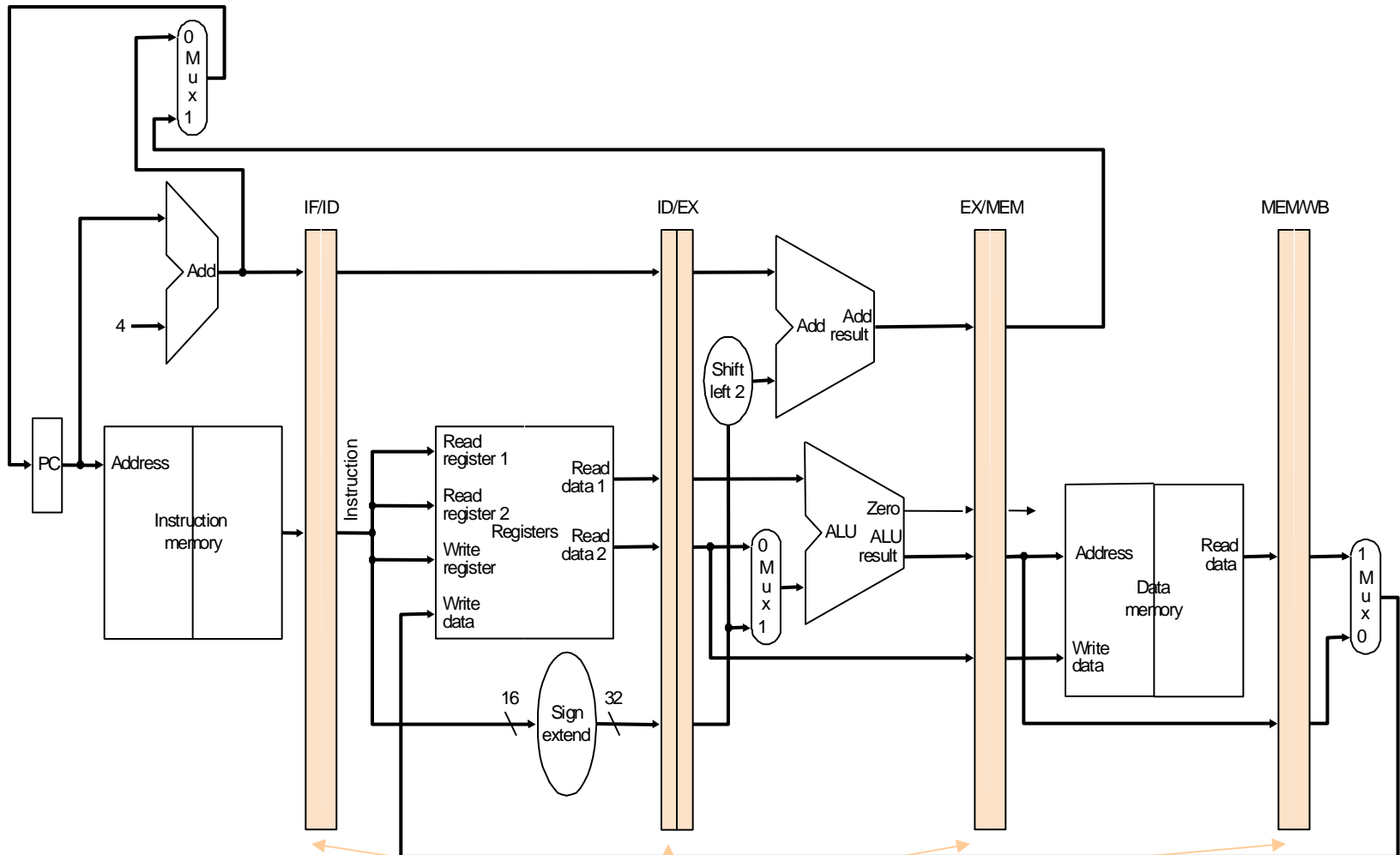
Caminho de dados uniclo



Fluxo de dados → com 2 exceções



Caminho de dados com *pipeline*

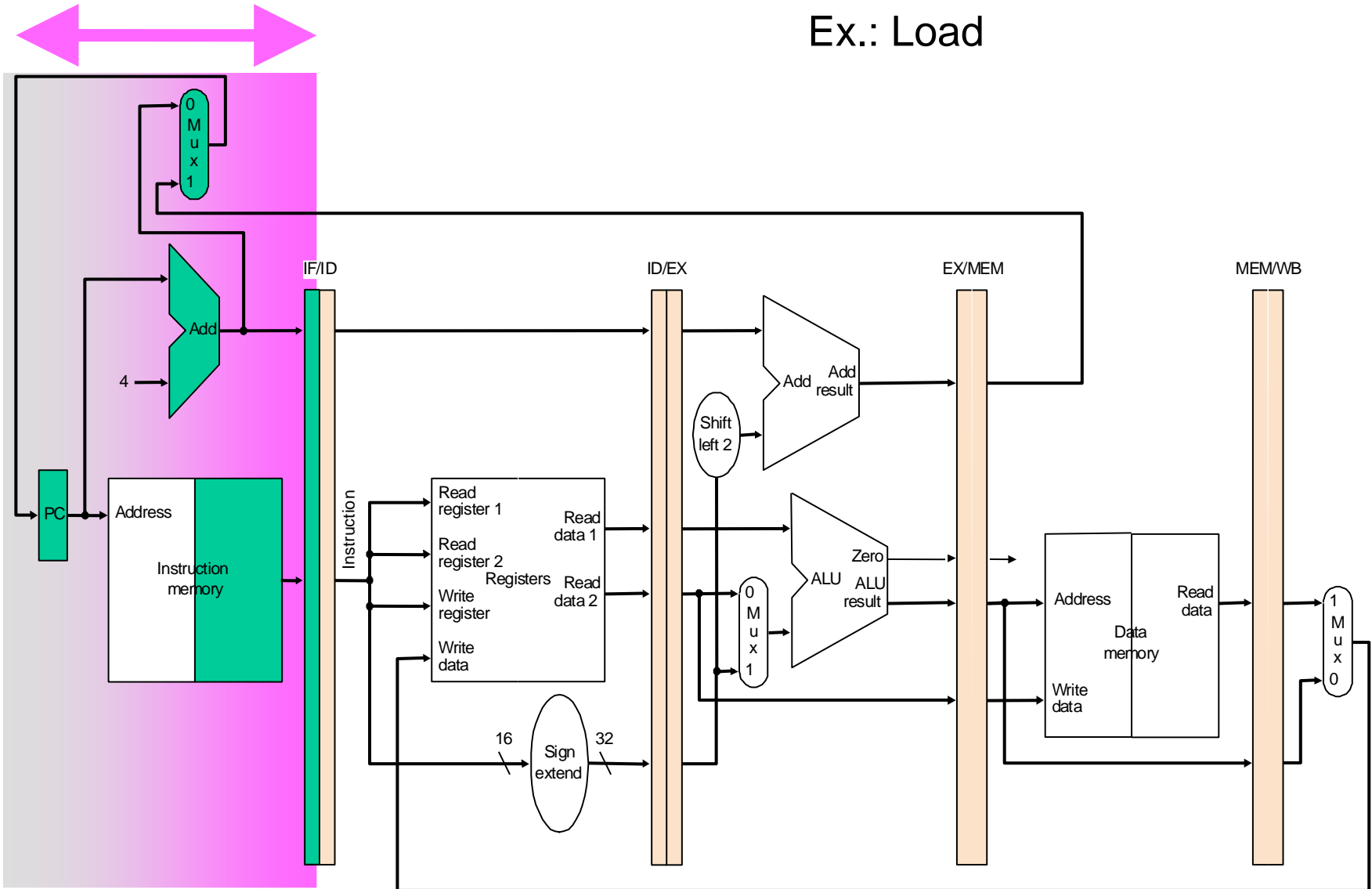


Registradores dos estágios do pipeline
Quais seus tamanhos?



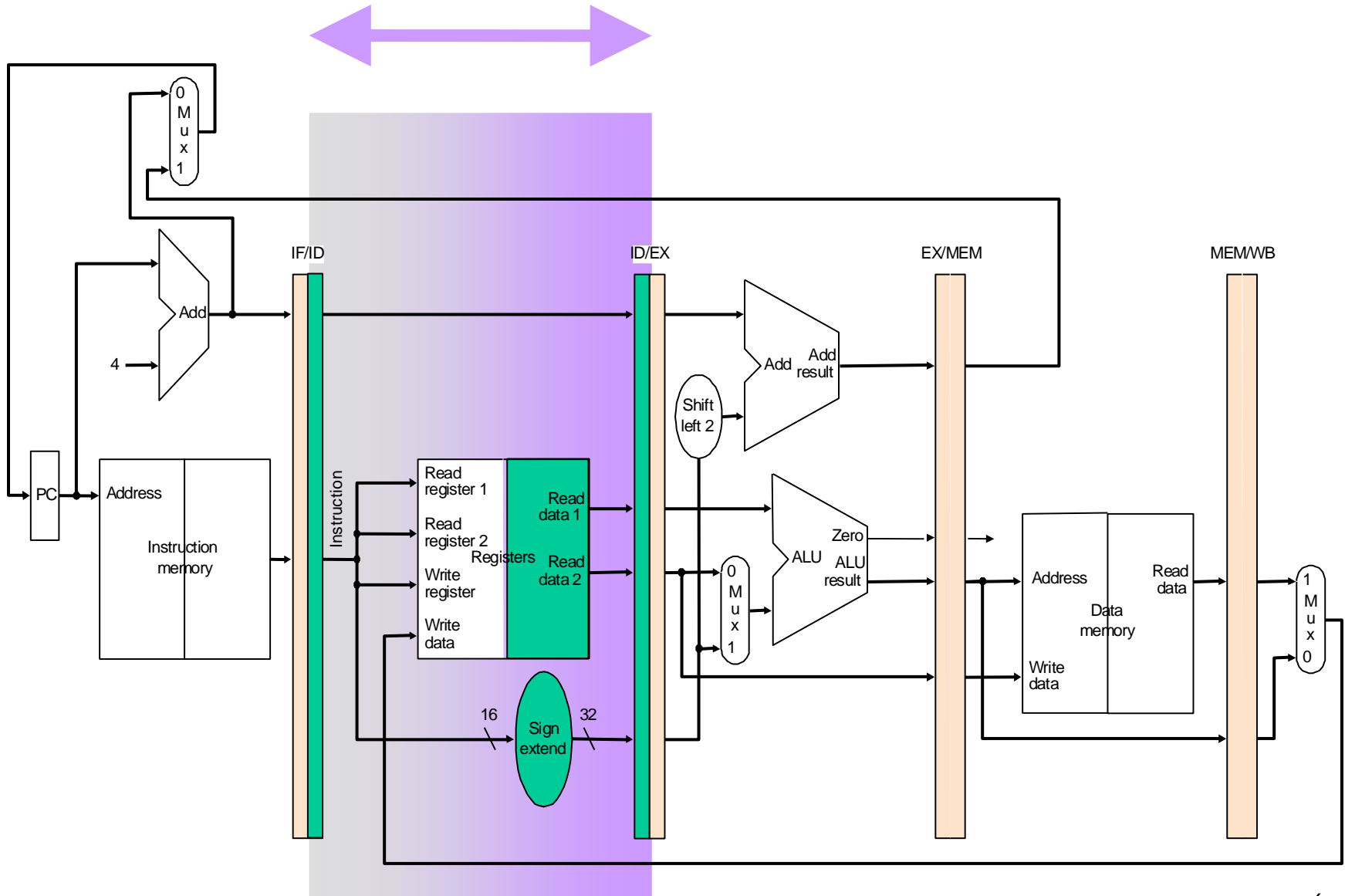
Busca de instrução

Ex.: Load



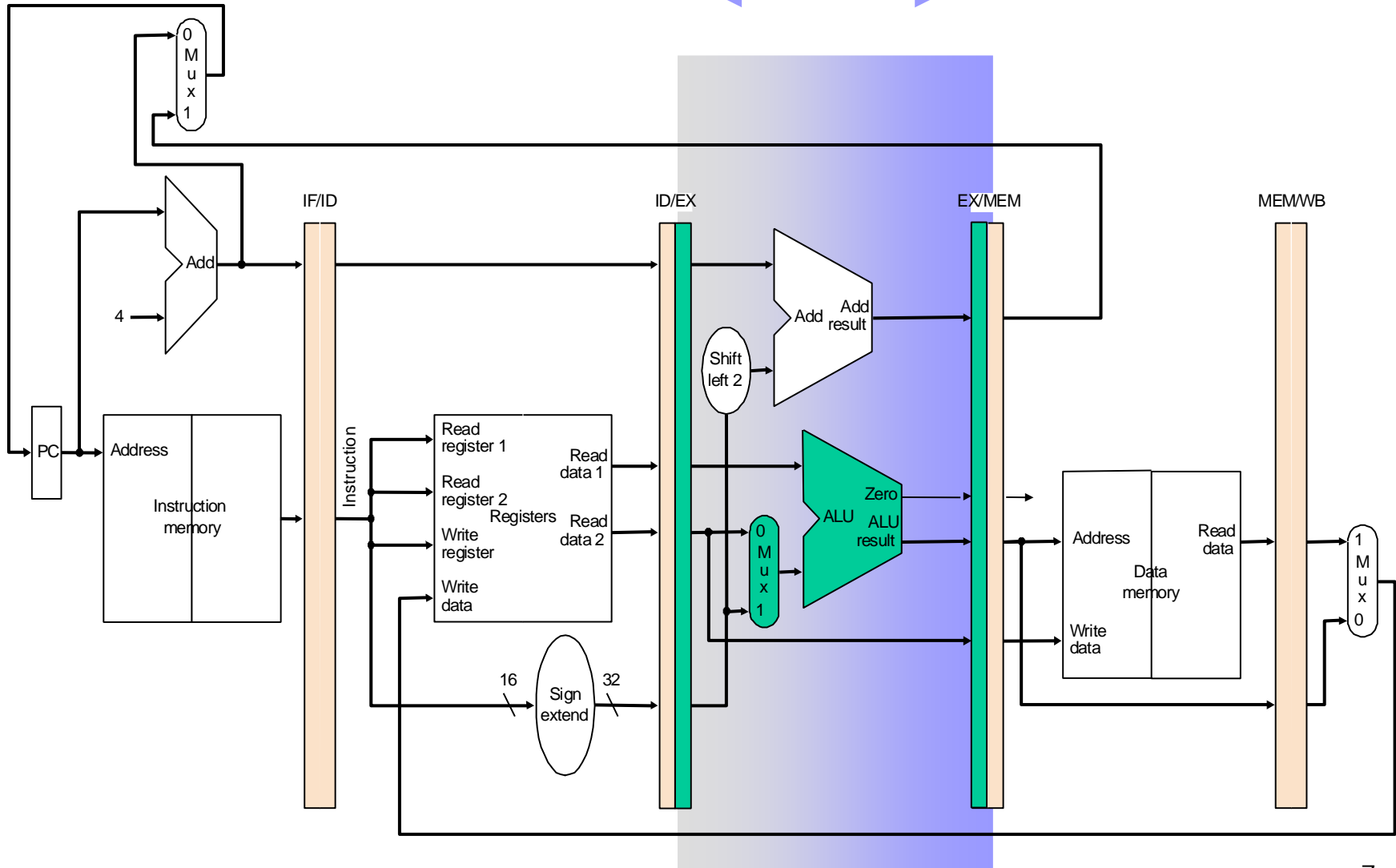


Decodificação/leitura de registros



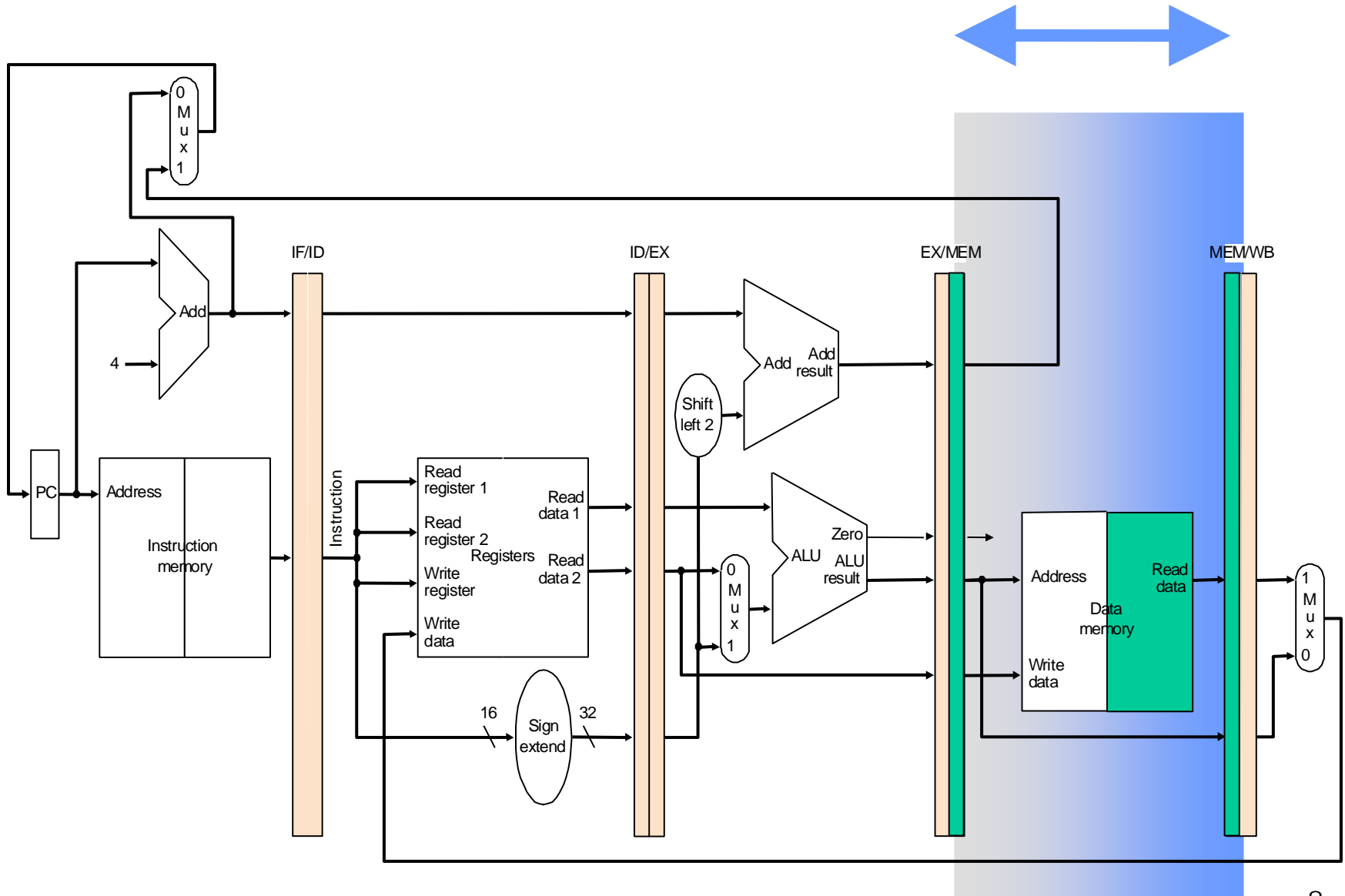


Execução/cálculo de endereço



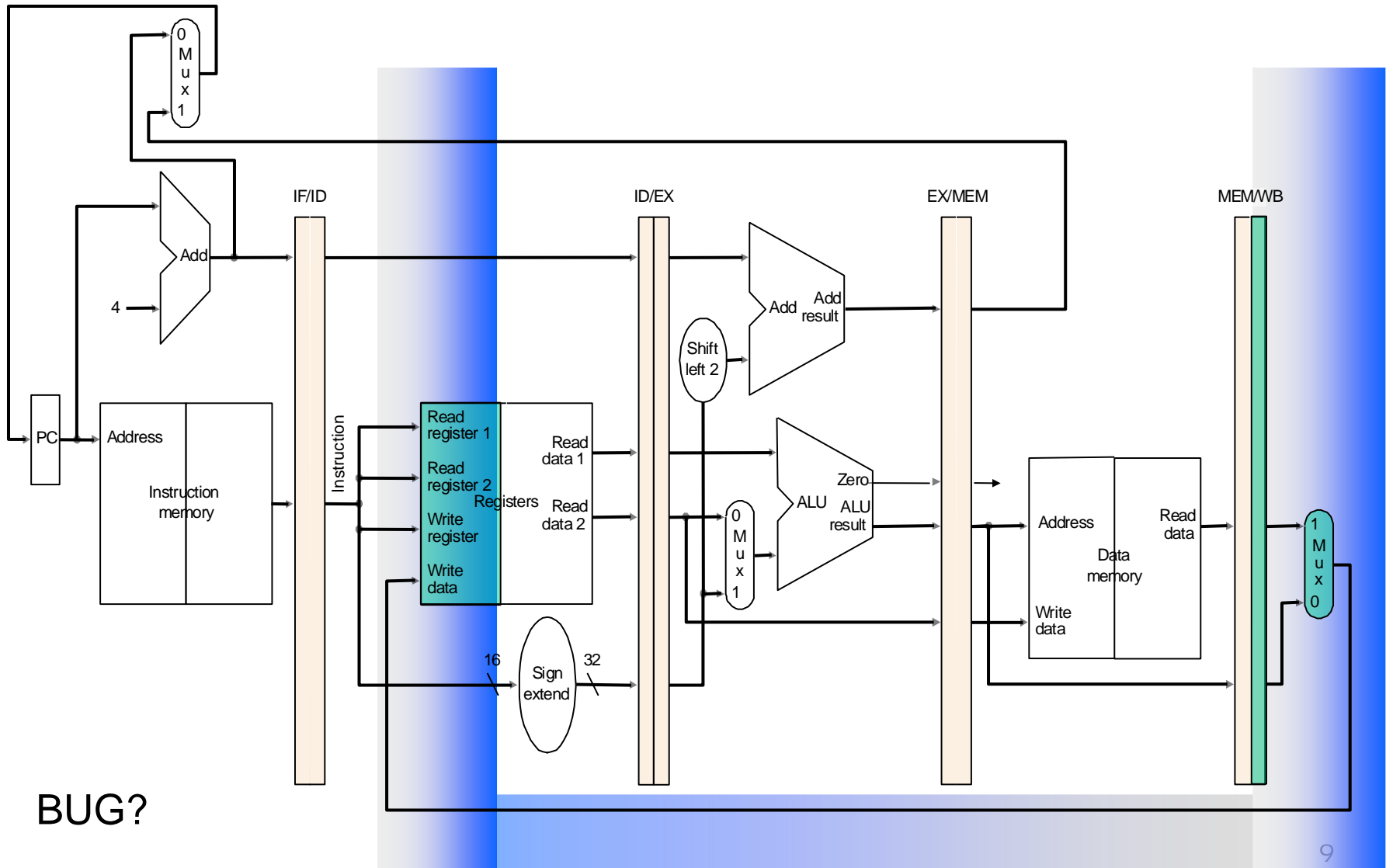


Acesso à memória



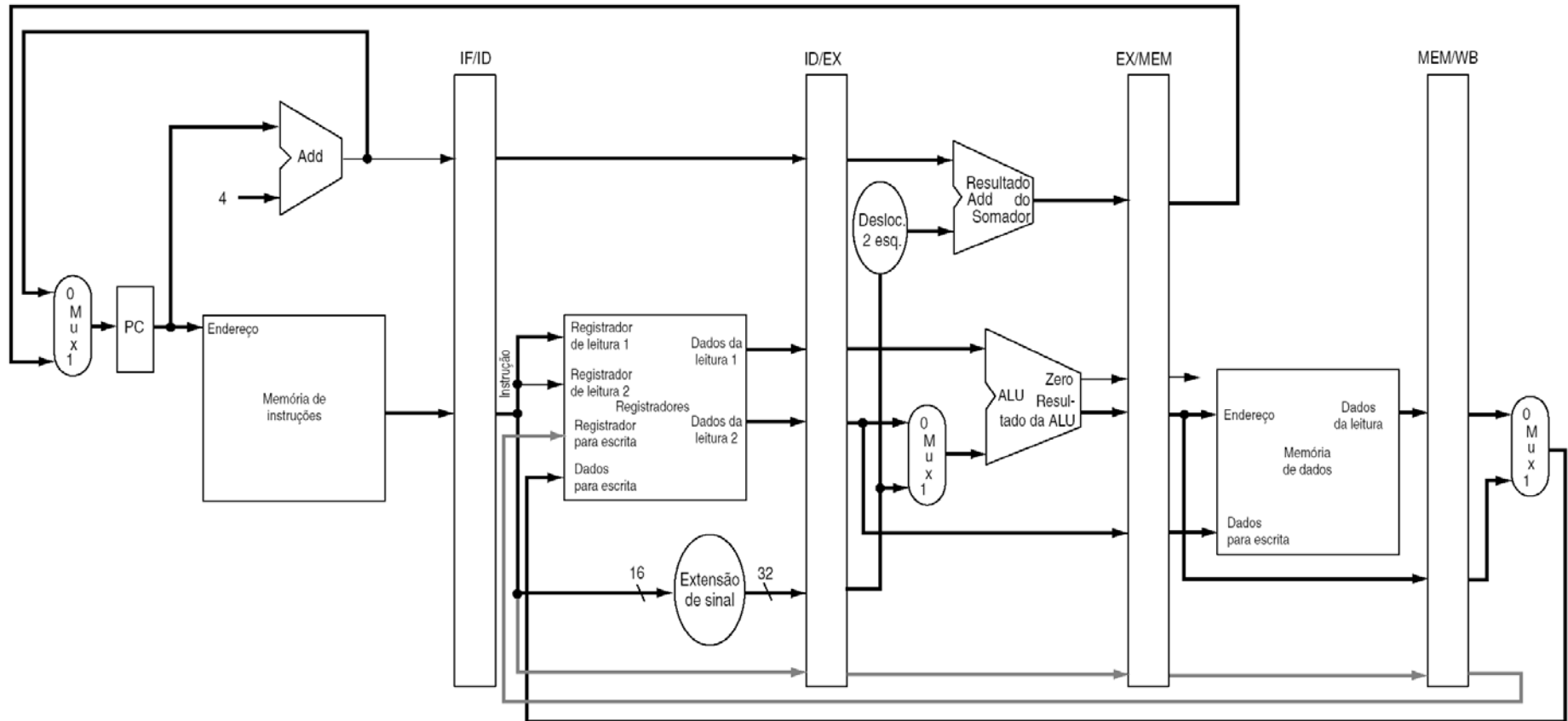


Escrita do Resultado



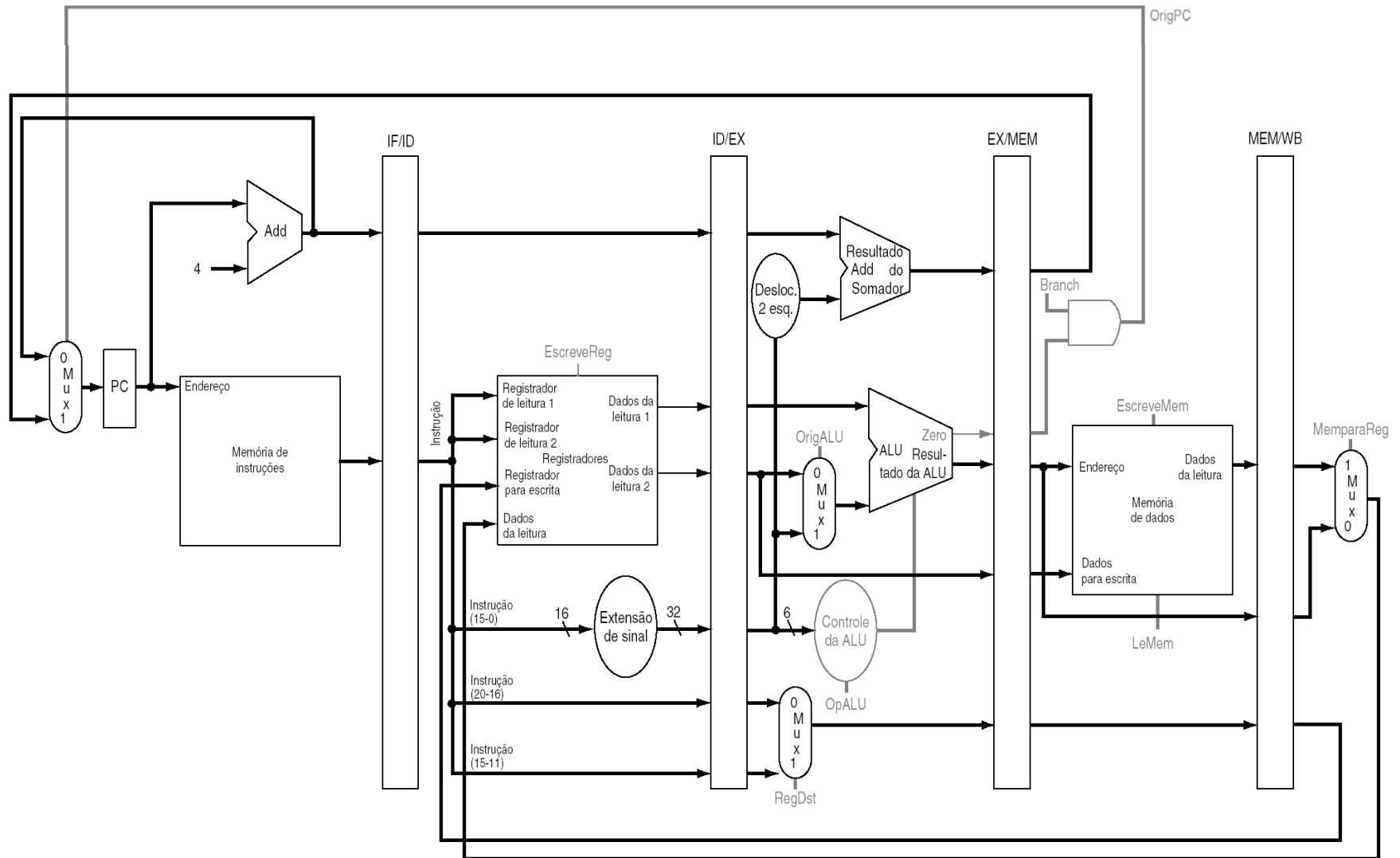


Corrigindo o BUG!!!





Controle do pipeline – Sinais de Controle





Controle do pipeline

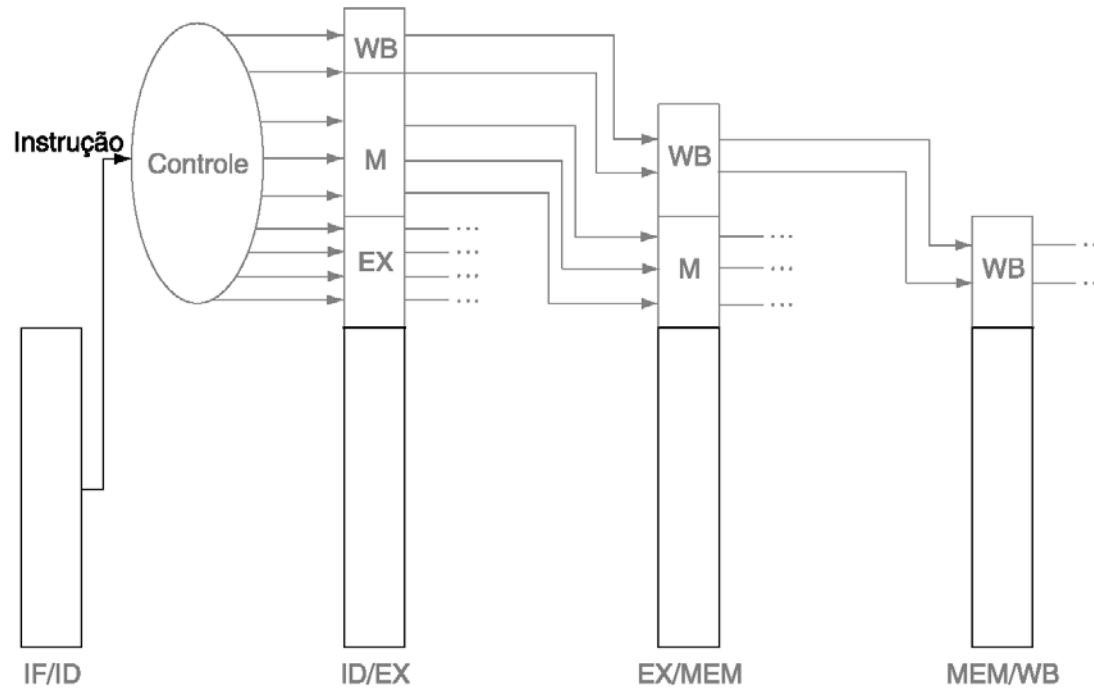
- Temos 5 estágios. O que precisa ser controlado em cada estágio?
 - Busca da instrução:
 - Nenhum sinal de controle
 - Decodificação da instrução / leitura do banco de regs.:
 - Nenhum sinal de controle
 - Execução / cálculo de endereço:
 - RegDest, OpALU, OrigALU
 - Acesso a memória:
 - Branch, LeMem, EscreveMem
 - Escrita do resultado
 - MemparaReg, EscreveReg



Controle do *Pipeline*

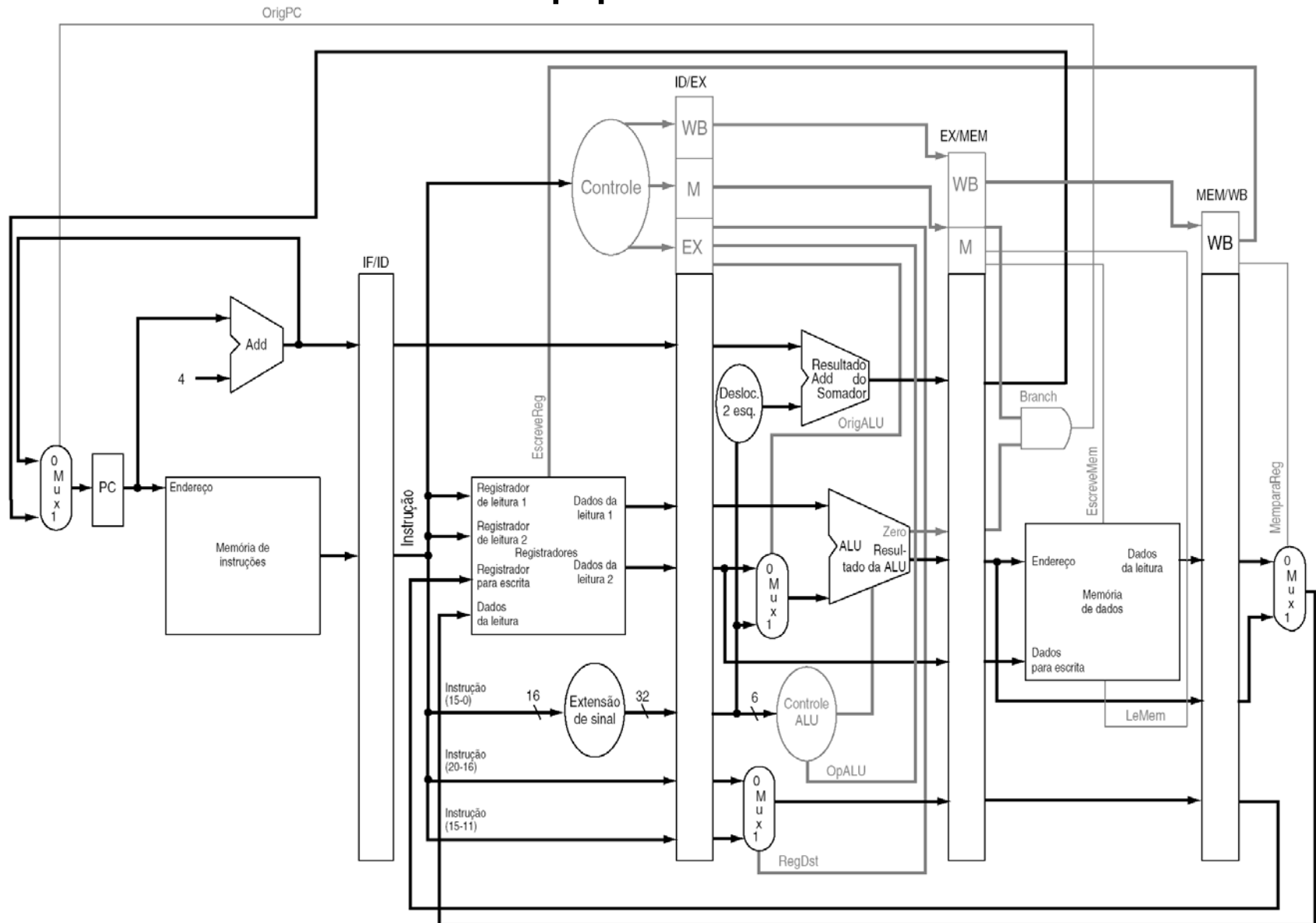
Sinais de controle são transmitidos da mesma forma que os dados

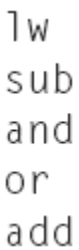
Instrução	Linhas de controle do estágio de cálculo de endereço/execução				Linhas de controle do estágio de acesso à memória			Linhas de controle do estágio de escrita do resultado	
	RegDst	OpALU1	OpALU0	OrigALU	Branch	LeMem	Escreve Mem	Escreve Reg	Mem para Reg
Formato R	1	1	0	0	0	0	0	1	0
lw	0	0	0	1	0	1	0	1	1
sw	X	0	0	1	0	0	1	0	X
beq	X	0	1	0	1	0	0	0	X

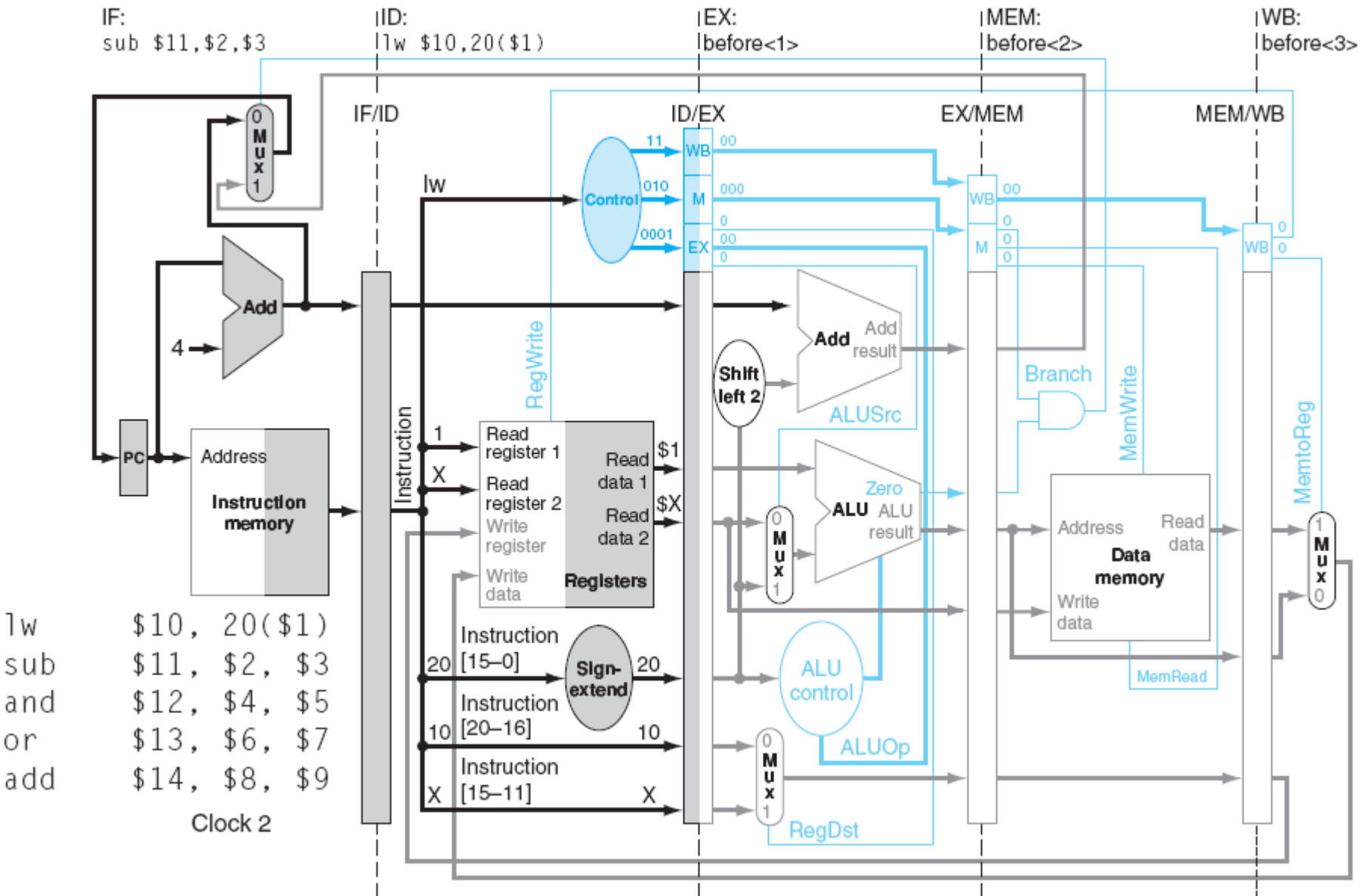


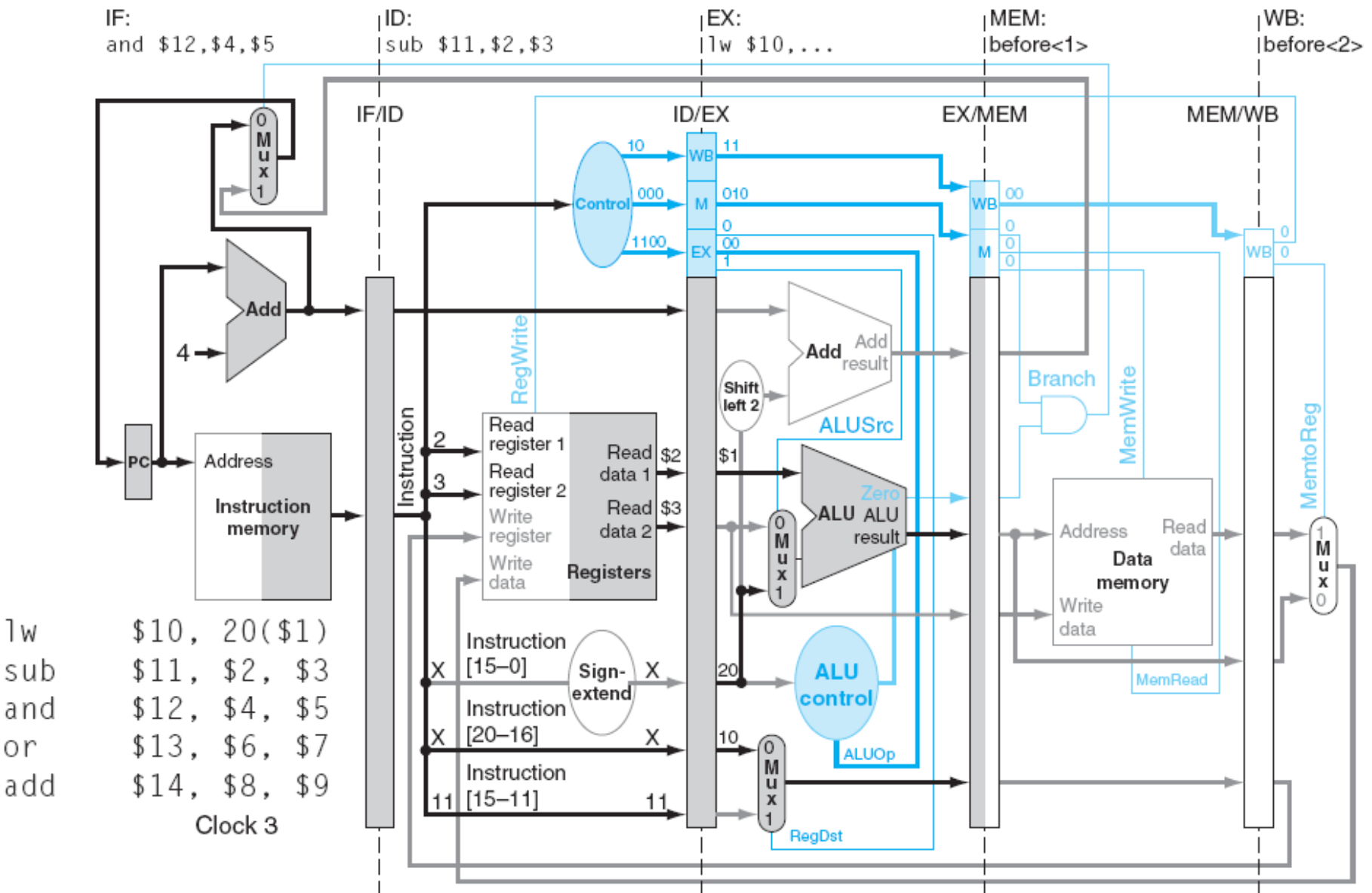


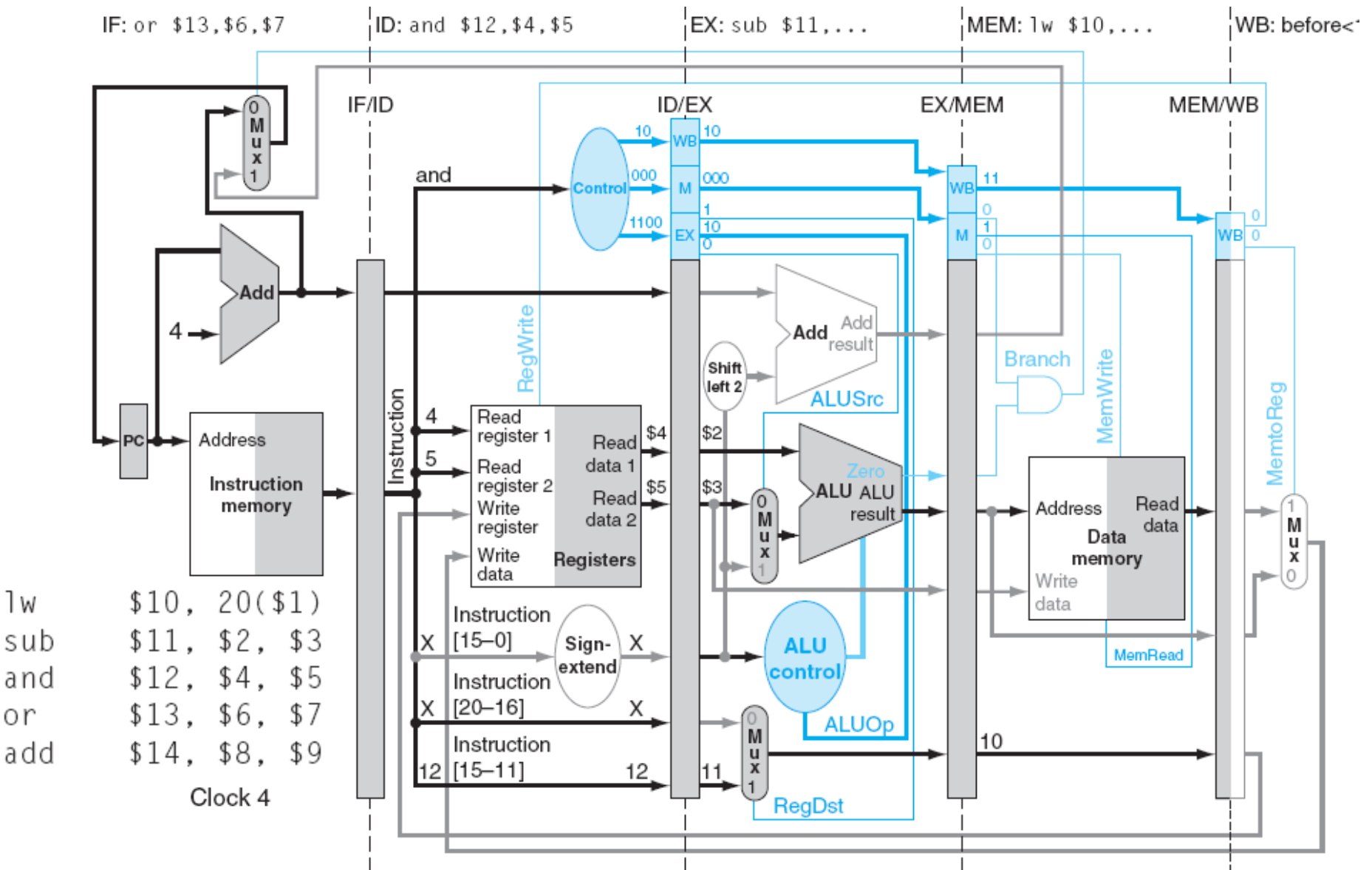
Caminho de dados pipeline com controle

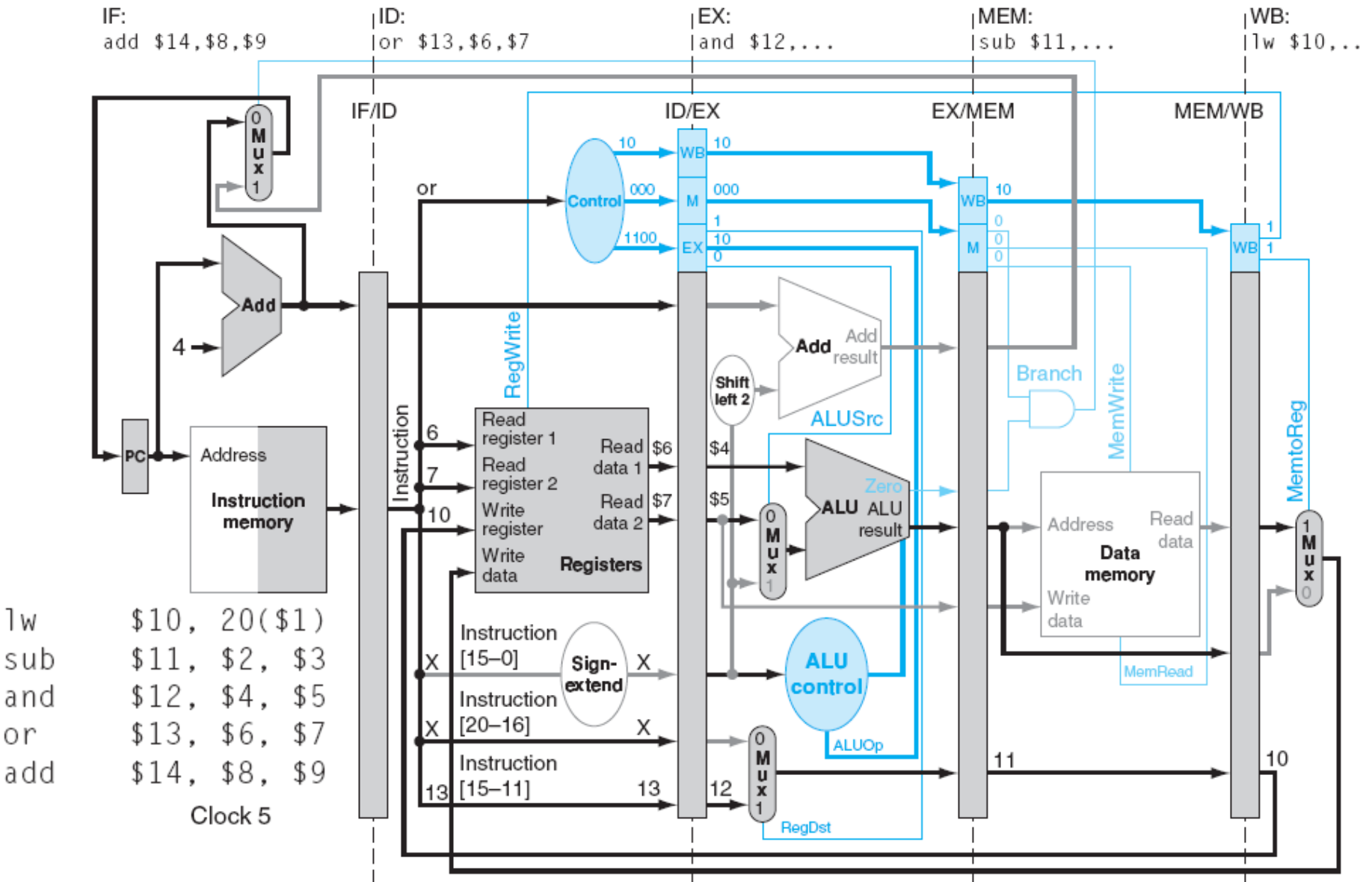


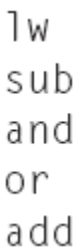


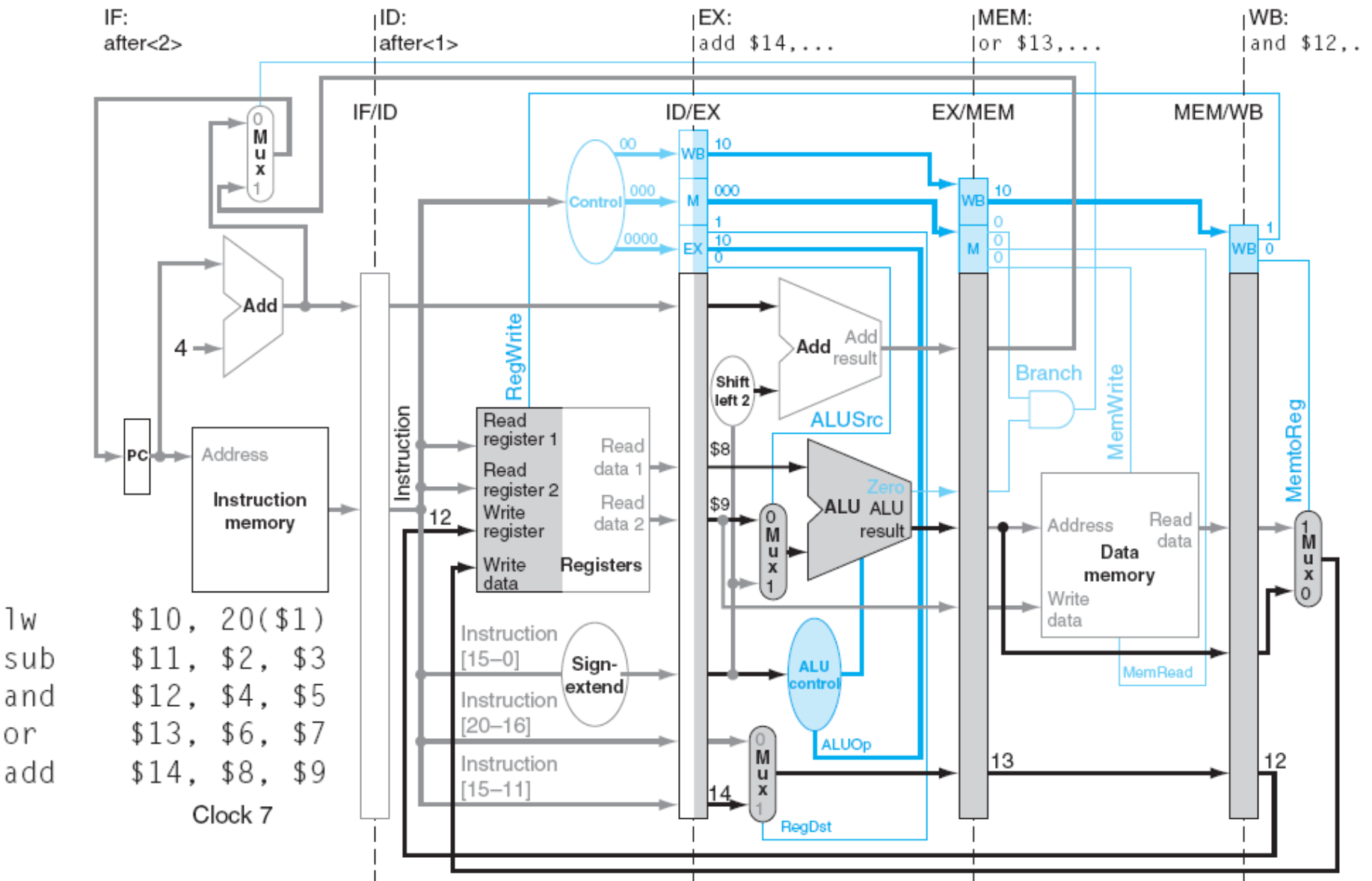


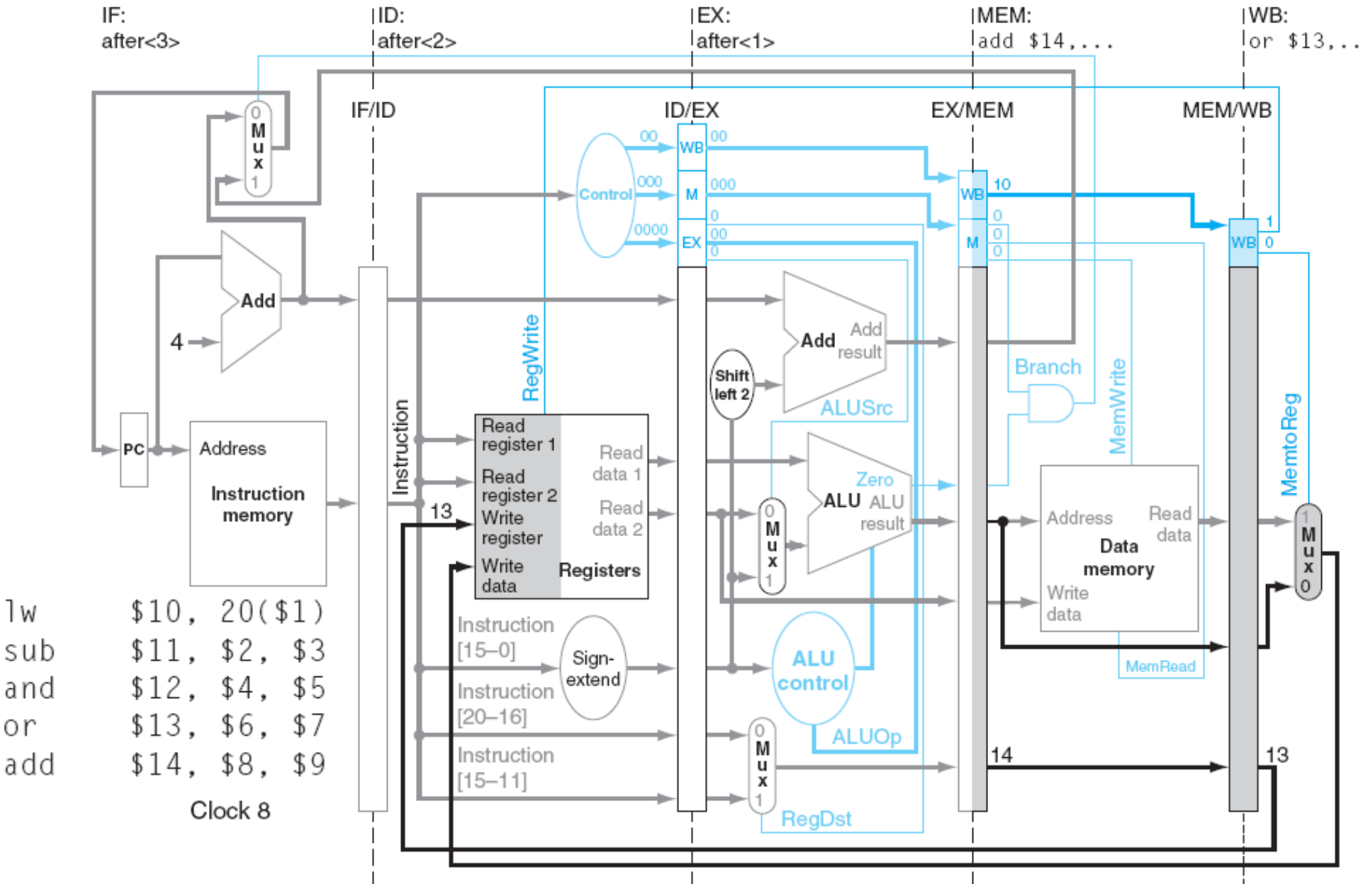


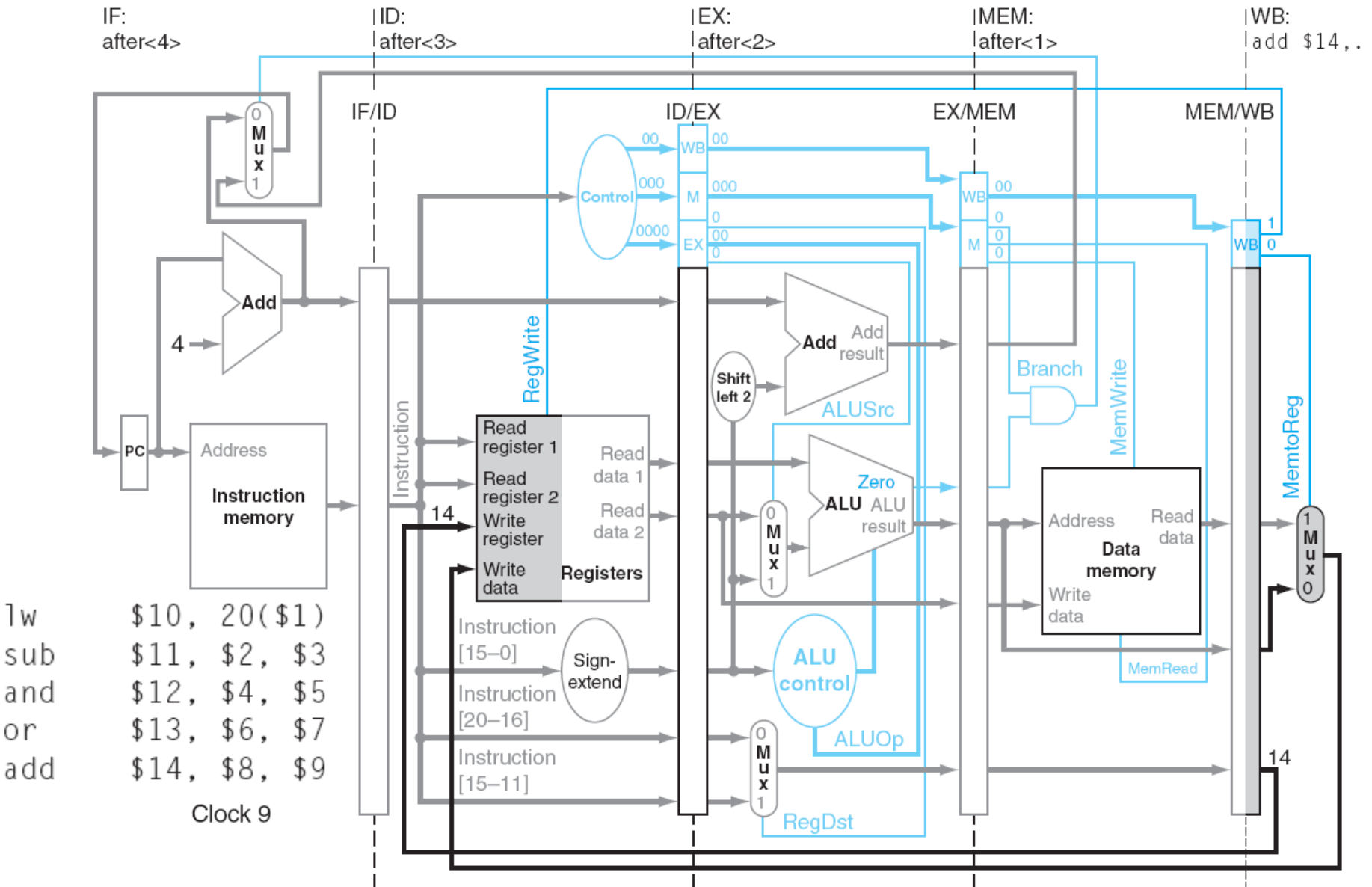














HAZARDS

■ Hazard de Dados: Forwarding

	Tempo (em ciclos de clock) →								
	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6	CC 7	CC 8	CC 9
Valor do registrador \$2:	10	10	10	10	10/-20	-20	-20	-20	-20
Valor de EX/MEM:	X	X	X	-20	X	X	X	X	X
Valor de MEM/WB:	X	X	X	X	-20	X	X	X	X

Ordem de
execução
do programa
(em instruções)

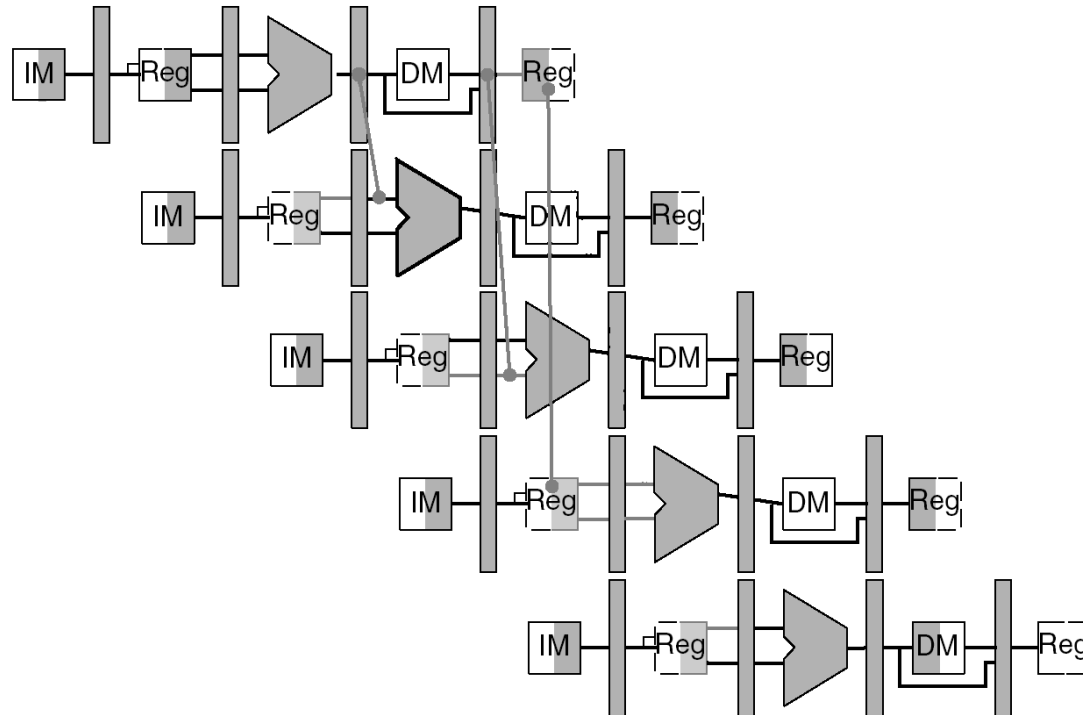
sub \$2, \$1, \$3

and \$12, \$2, \$5

or \$13, \$6, \$2

add \$14, \$2, \$2

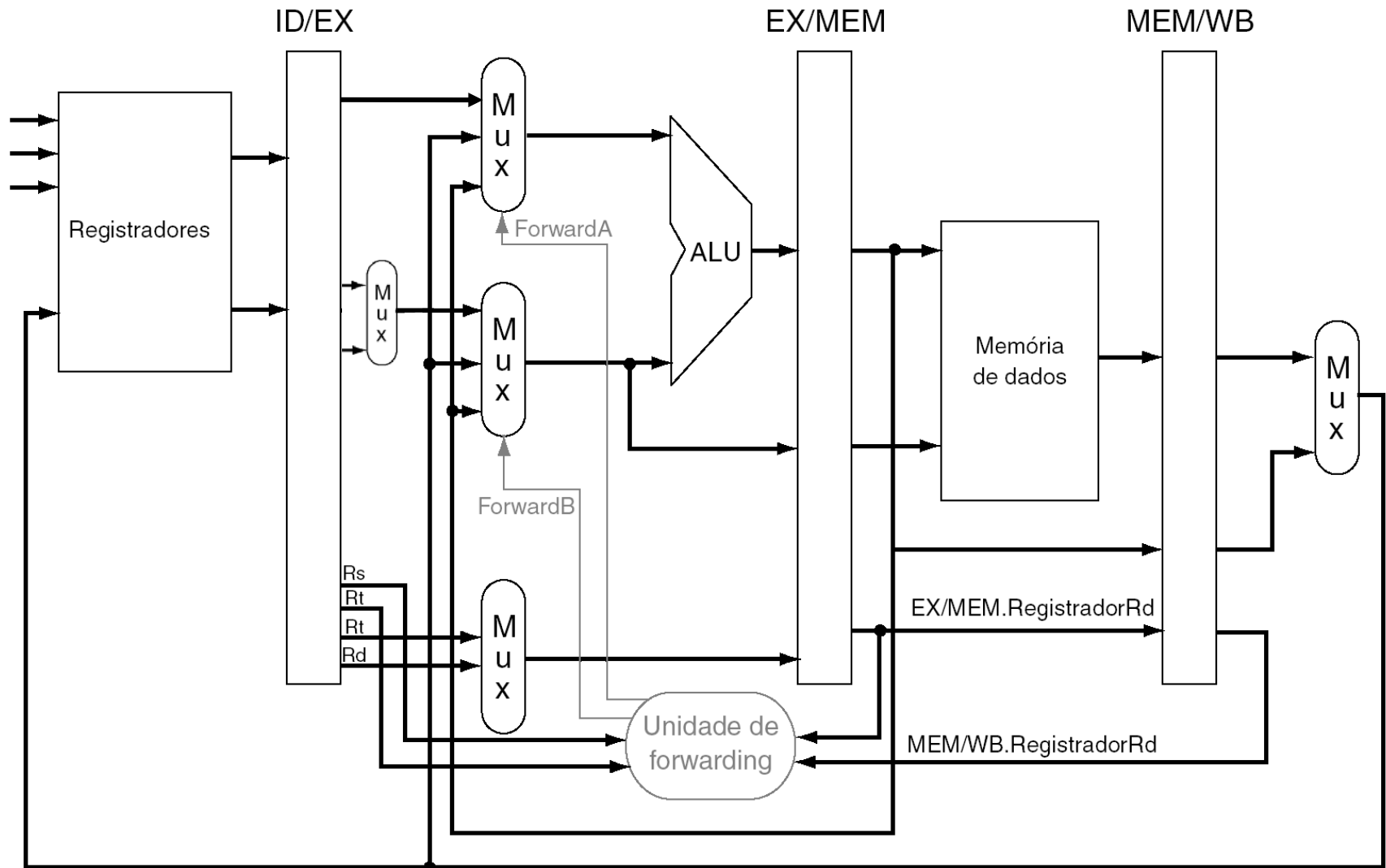
sw \$15, 100(\$2)



E se este \$2 fosse \$13?

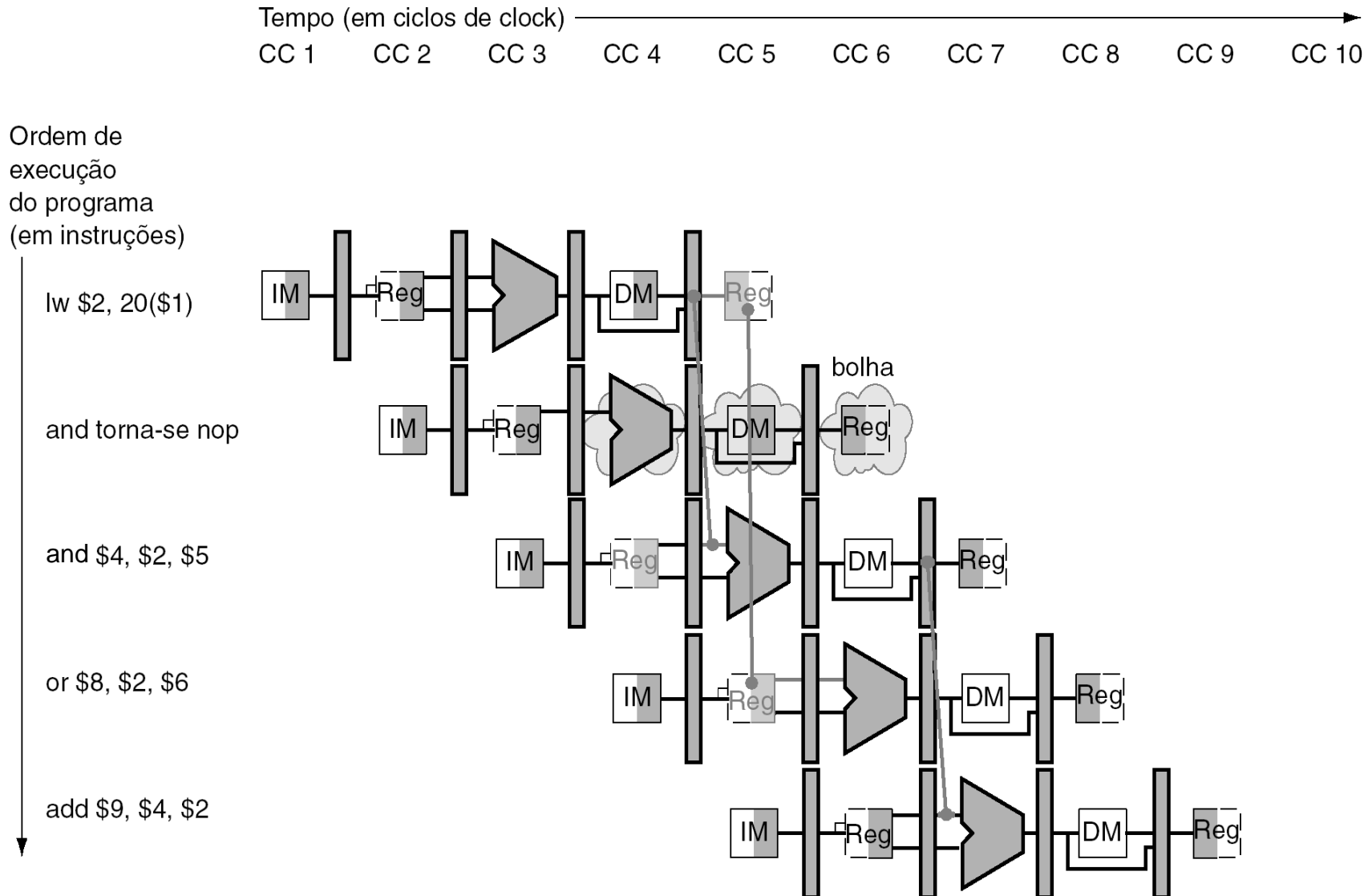


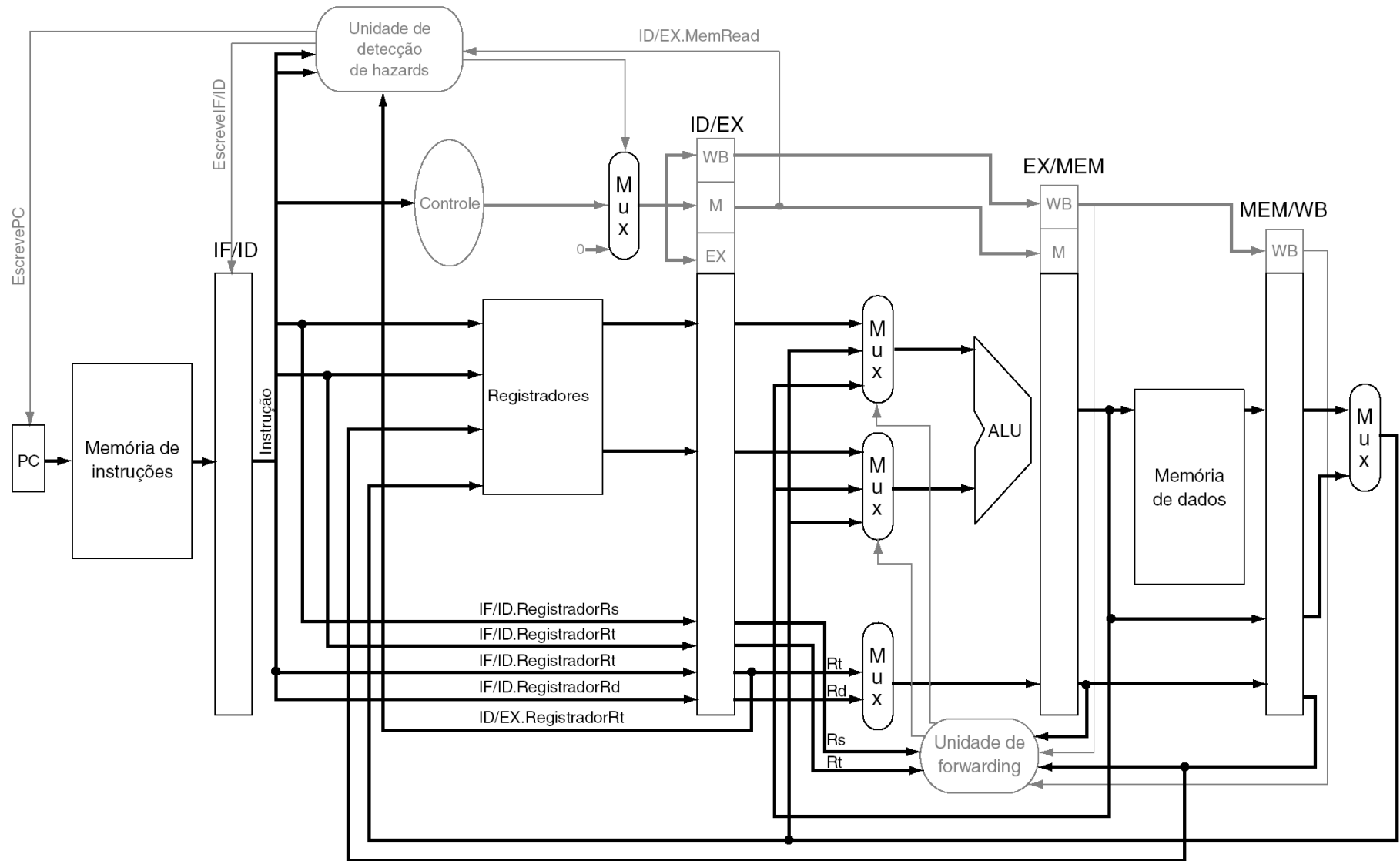
Hazard de dados: Forwarding





Quando forward não é possível





Obs.: Falta o mux OrigALU!



Deteção de Hazards

- Comparar índices dos registradores a serem lidos e escritos, armazenados nos registradores do *pipeline*:
- Deve-se comparar igualmente se a instrução escreve em um registrador ($\text{EscreveReg} = 1$) ou lê da memória ($\text{LeMem} = 1$)



Detecção de Hazard de dados

- 1. Hazard EX
 - IF (EX/MEM.EscreveReg
 - and (EX/MEM.RegistradorRd \neq 0)
 - and (EX/MEM.RegistradorRd = ID/EX.RegistradorRs)) ForwardA = 10
 - IF (EX/MEM.EscreveReg
 - and (EX/MEM.RegistradorRd \neq 0)
 - and (EX/MEM.RegistradorRd = ID/EX.RegistradorRt)) ForwardB = 10

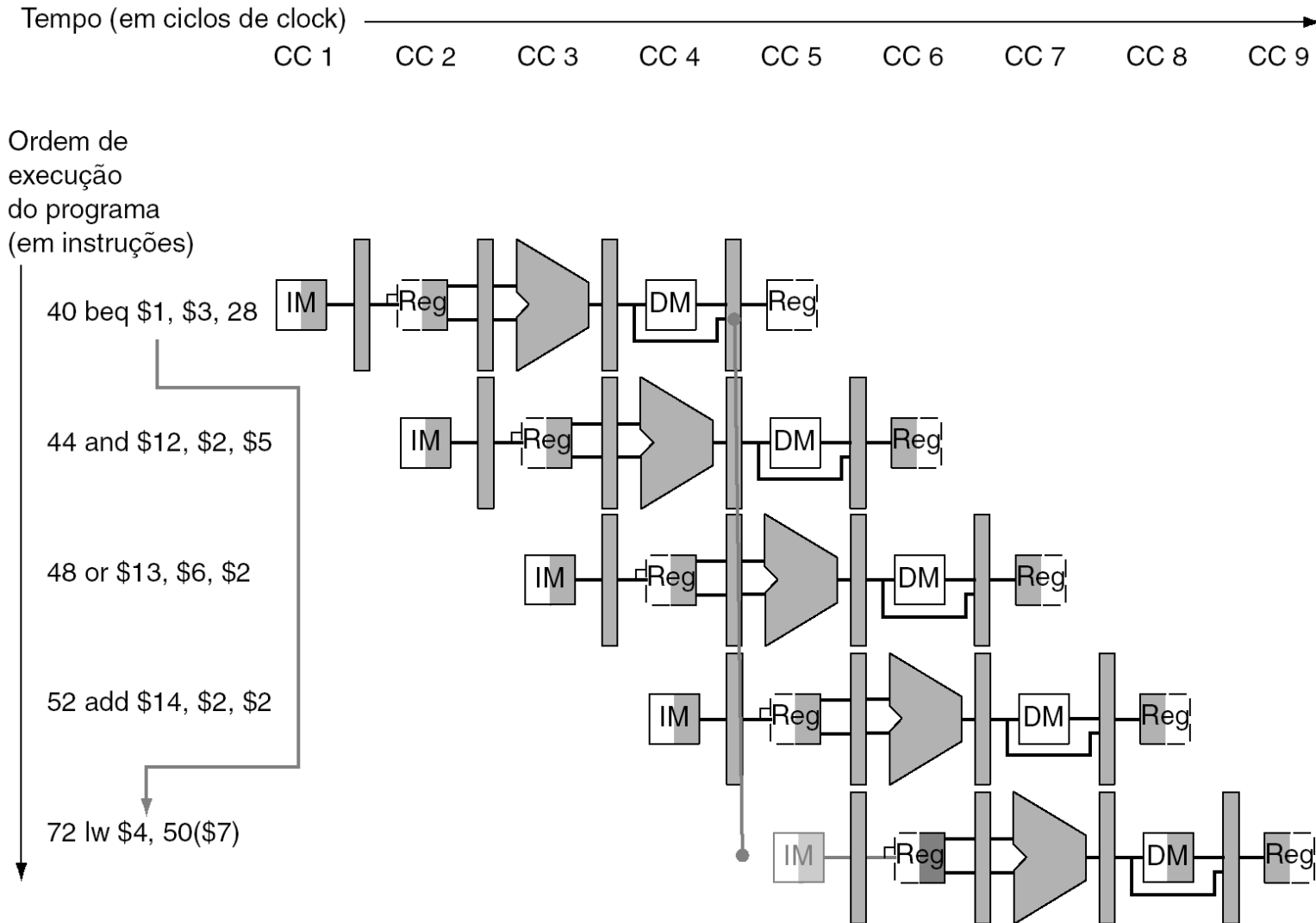
- 2. Hazard MEM
 - IF (MEM/WB.EscreveReg
 - and (MEM/WB.RegistradorRd \neq 0)
 - and (MEM/WB.RegistradorRd = ID/EX.RegistradorRs)) ForwardA = 01
 - IF (MEM/WB.EscreveReg
 - and (MEM/WB.RegistradorRd \neq 0)
 - and (MEM/WB.RegistradorRd = ID/EX.RegistradorRt)) ForwardB = 01

- 3. LW
 - IF (ID/EX.LeMem
 - and ((ID/EX.RegistradorRt = IF/ID.RegistradorRs)
 - or (ID/EX.RegistradorRt = IF/ID.RegistradorRt))) Ocasiona stall no pipeline



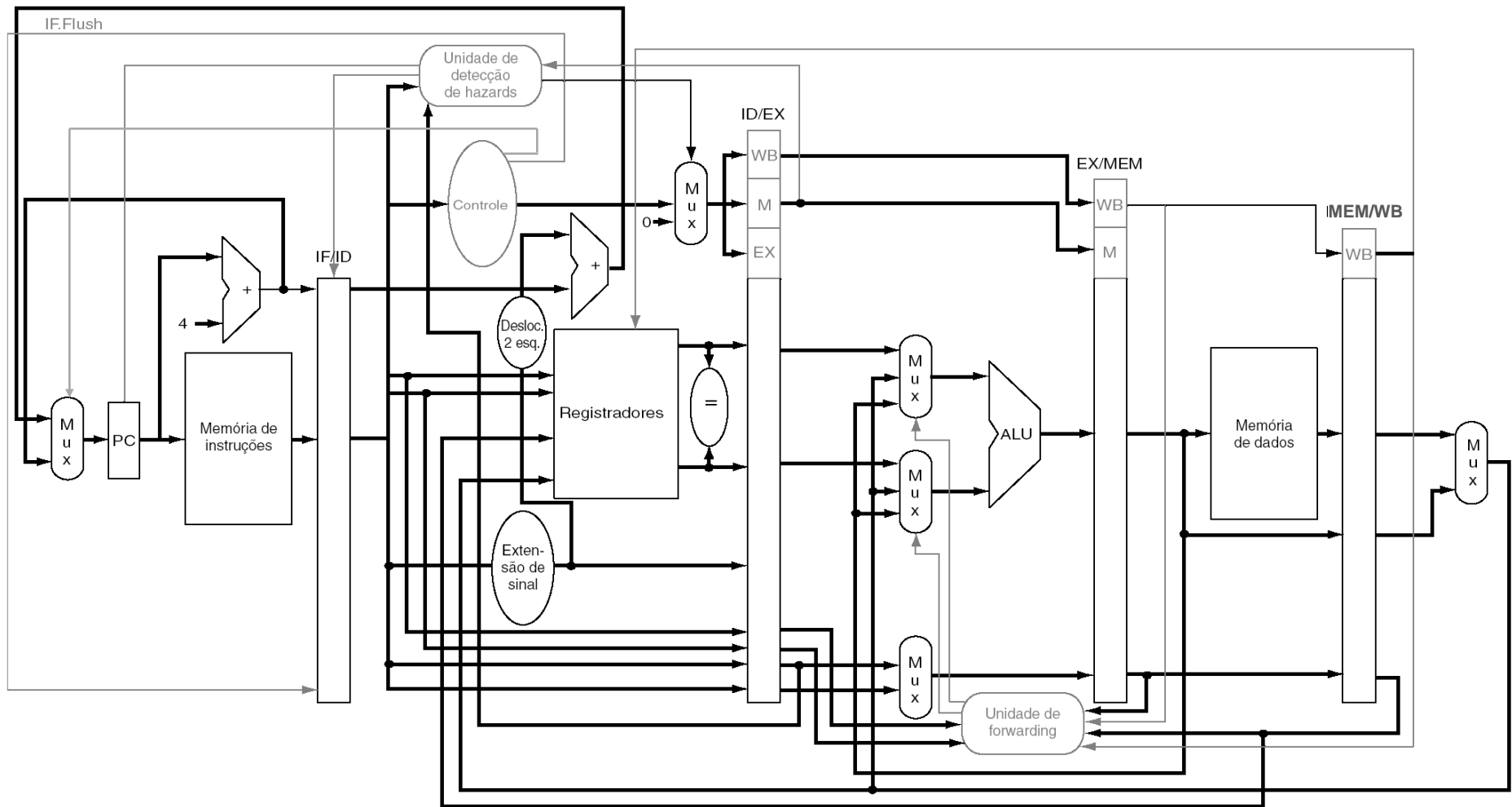
Hazard de Controle

■ Prevendo: Desvio não-tomado





Hazard de Controle otimizado (1 bolha)



Buscar formas mais inteligentes de previsão de desvios!



Comparação: Uniciclo x Multiciclo x Pipeline

■ Suponha os seguintes tempos das unidades operativas:

- ☐ Acesso a memória: 200ps
- ☐ Operação da ULA: 100ps
- ☐ Acesso ao banco de registradores: 50ps

■ Dado o workload composto de:

- ☐ 25% loads
- ☐ 10% stores
- ☐ 11% branches
- ☐ 2% jumps
- ☐ 52% operações com a ULA

Compare o desempenho, considerando para o pipeline

50% dos loads é seguido de uma operação que requer o argumento

25% dos desvios são previstos erradamente e que o atraso da previsão errada é 1 ciclo de clock

jumps usam 2 ciclos de clock

ignore outros hazards.



Solução

Ciclo único:

período de clock: $200+50+100+200+50 = 600\text{ps}$

$\text{CPI}=1$

Tempo de execução médio da instrução: $600 \times 1 = 600\text{ps}$

Multiciclo:

período de clock: 200ps

$\text{CPI} = .25 \times 5 + .1 \times 4 + .11 \times 3 + .02 \times 3 + 0.52 \times 4 = 4.12$

Tempo de execução médio da instrução: $200 \times 4.12 = 824\text{ps}$

Pipeline:

período de clock: 200ps

$\text{CPI} = .25 \times 1.5 + .1 \times 1 + .11 \times 1.25 + .02 \times 2 + 0.52 \times 1 = 1.17$

Tempo de execução médio da instrução: $200 \times 1.17 = 234\text{ps}$