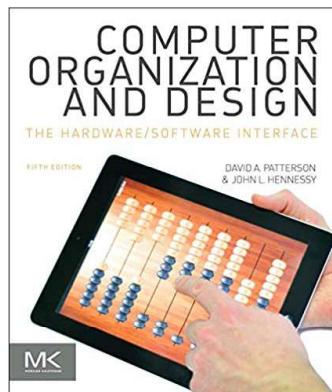


Aula 08 – Pipeline

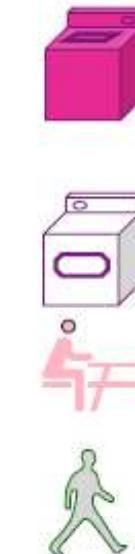
Prof. Dr. Clodoaldo A. de Moraes Lima

Material baseado no livro “Patterson, David A., Hennessy, J. L. - Computer Organization And Design: The Hardware/Software Interface”

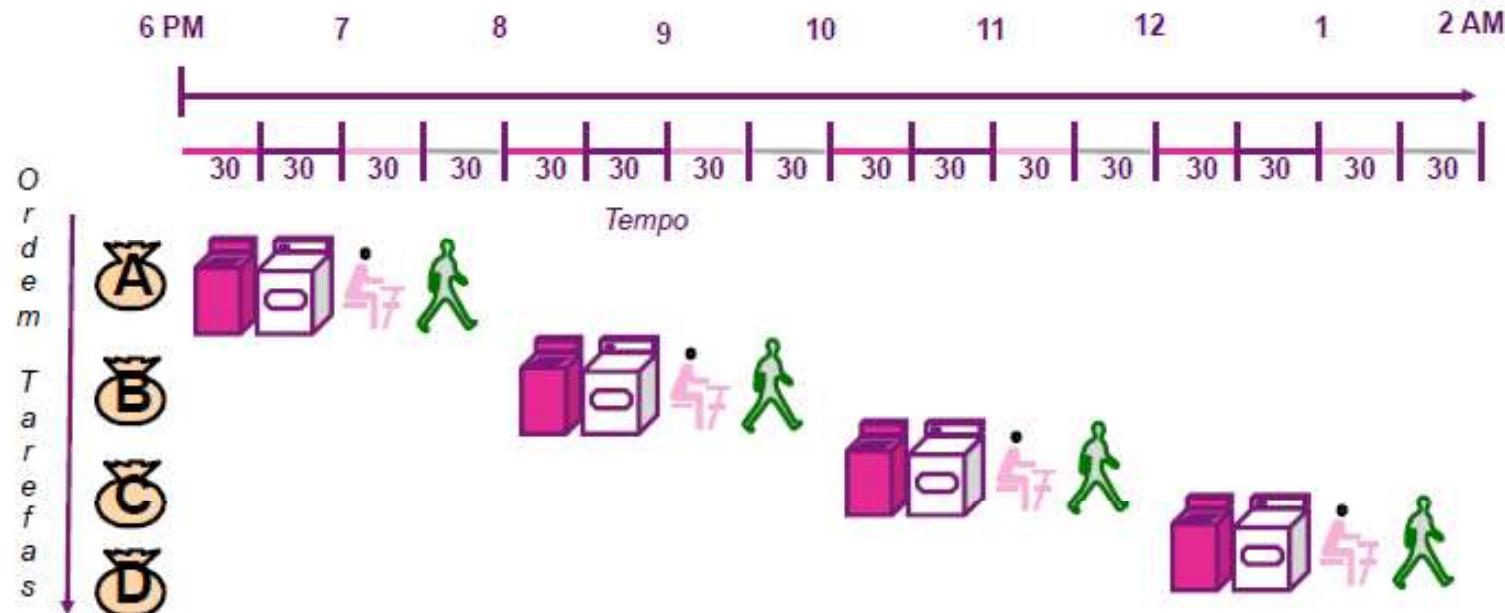


Processo de Pipelining (exemplo)

- Ana, Bruno, Carla, Luiz têm roupas sujas a serem lavadas, secadas, dobradas e guardadas
 - Lavadora leva 30 minutos
 - Secadora leva 30 minutos
 - “Dobrar” leva 30 minutos
 - “Guardar” leva 30 minutos

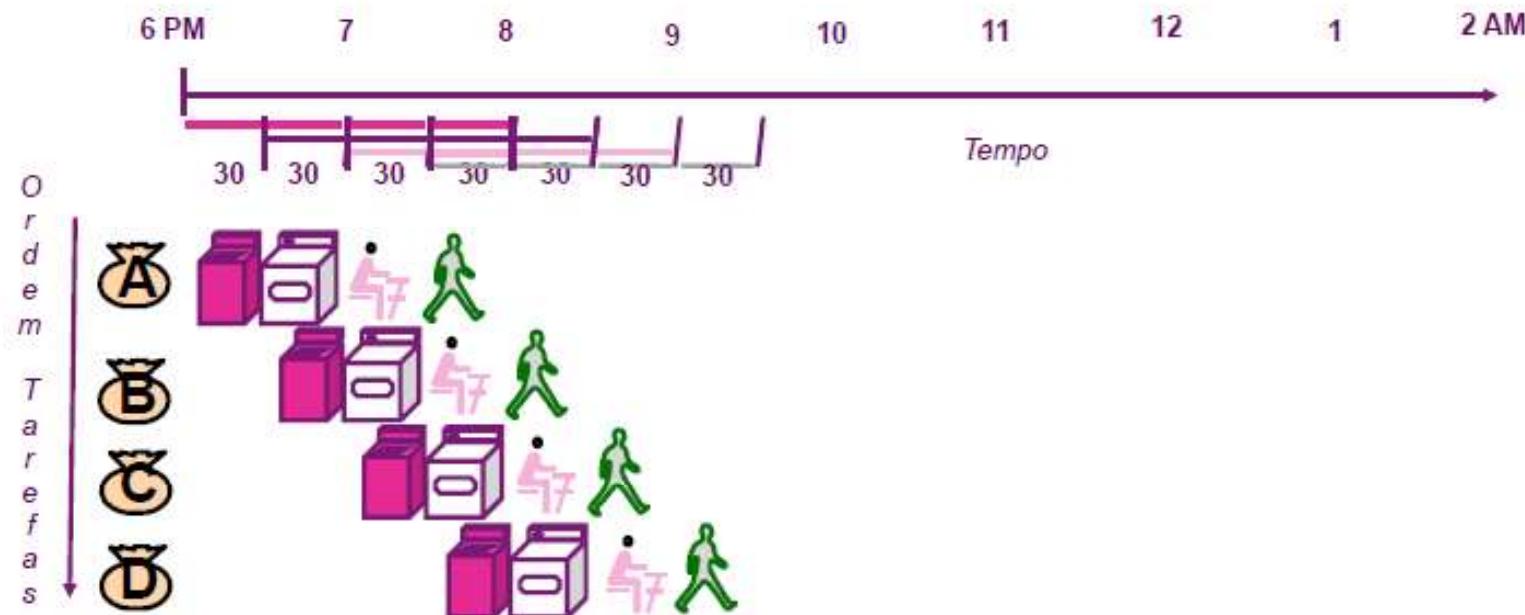


Pipelining (exemplo de lavanderia)



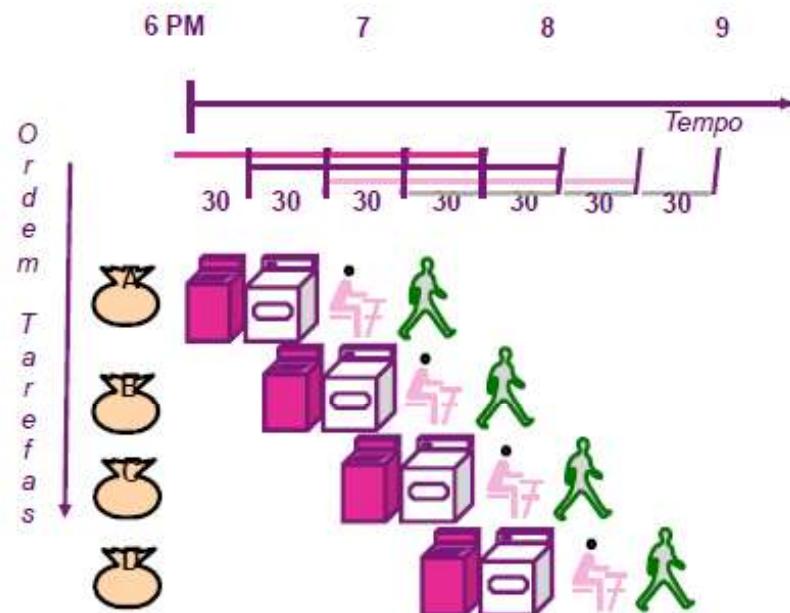
- Processo sequencial de lavagem leva oito horas para os quatro
- Quanto tempo levaria, utilizando-se pipelining?

Pipelining (exemplo de lavanderia)



- Utilizando-se a técnica de pipeline consome-se 3.5 horas no processo de lavagem !

Observações sobre Pipelining



- Pipelining não ajuda a melhorar a latência de uma atividade, mas aumenta o throughput
- Várias tarefas operando em paralelo utilizam recursos diversos
- Aceleração potencial = Número de estágios de pipe
- Taxa de pipeline limitada pelo estágio mais lento
- Desequilíbrios na duração dos estágios reduz a aceleração
- Tempo para “encher” o pipeline e para “esvaziá-lo” reduz a aceleração
- Pode parar por dependências

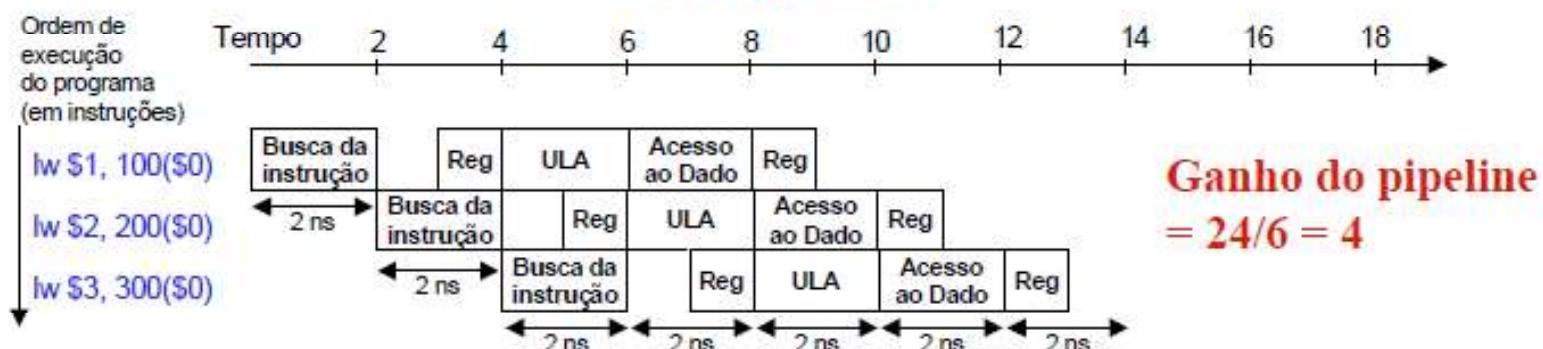
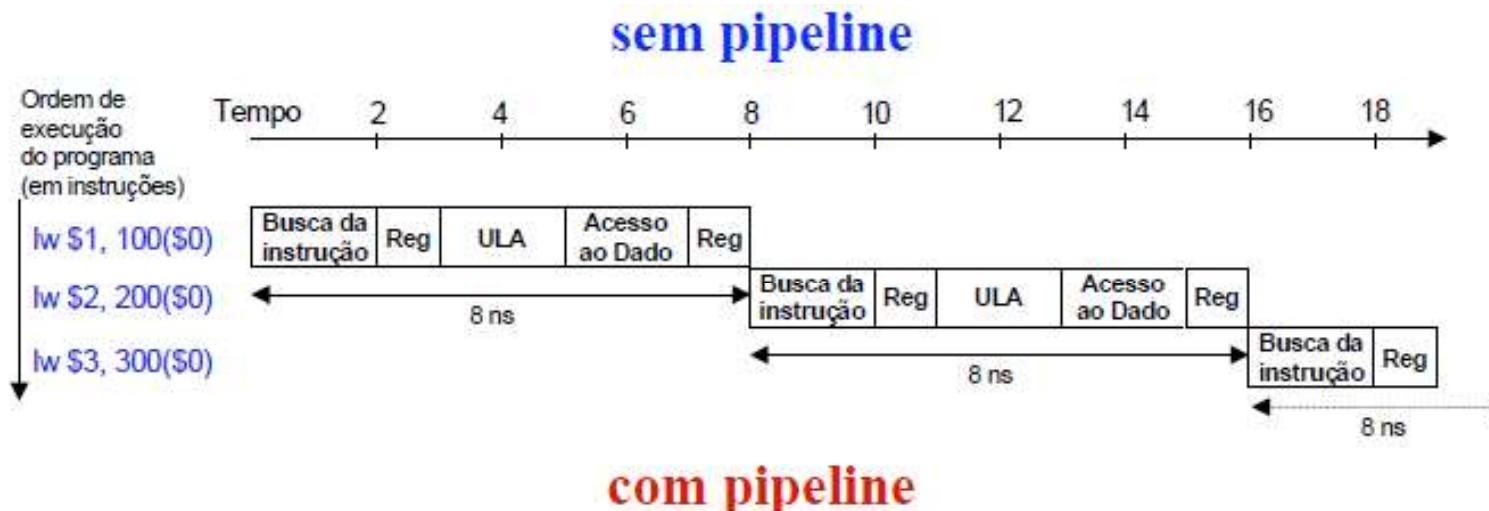
Pensando em termos de instruções MIPS

- Considere os tempos de resposta de cada um dos componentes da via de dados e os tempos para execução das instruções conforme a tabela abaixo.
- Assuma que o tempo de cada estágio é:
 - 1ns para ler ou escrever no registrador;
 - 2ns para os outros estágios;
- Podemos estimar que o período de clock será igual a 8ns para uma CPU MIPS na sua versão monociclo.

Classe da instrução	Busca da instrução	Leitura de registrador	Operação na ULA	Acesso ao dado	Escrita no registrador	Tempo total
Load word (ld)	2 ns	1 ns	2 ns	2 ns	1 ns	8 ns
Store word (sw)	2 ns	1 ns	2 ns	2 ns	---	7 ns
Formato R (add, sub, and, or, slt)	2 ns	1 ns	2 ns	---	1 ns	6 ns
Branch (beq)	2 ns	1 ns	2 ns	---	---	5 ns

- Pensando na execução de 3 loads (instrução mais lenta)

Execução sem e com pipeline



Características do Pipelining

- Se os estágios do pipeline não estiverem balanceados e um dos estágios for mais lento que os outros, a vazão do pipeline será afetada como um todo.
- Idealmente, cada estágio é balanceado (todos os estágios estão prontos para iniciar ao mesmo tempo e levam a mesma quantidade de tempo para serem concluídos).
- Assim, o tempo de execução por instrução no pipeline seria definido por:

$$\text{Tempo entre instruções}_{\text{pipeline}} = \frac{\text{Tempo entre instruções}_{\text{não-pipeline}}}{\text{Número de estágios do pipe}}$$

Características do Pipelining

- A expressão anterior corresponde ao ganho ideal (máximo).
 - Existe um overhead para preencher o pipeline e os estágios podem não estar perfeitamente平衡ados...
- Em termos de uma CPU, a implementação do pipeline tem efeito de reduzir o tempo médio por instrução e, portanto, o valor médio do CPI.
 - Ex: Se cada instrução num microprocessador leva 5 ciclos de clock (sem pipelined) e se houver um pipeline de 4 estágios, o CPI ideal teórico seria 1.25

O ISA no MIPS projetado para pipeline:

- Todas as instruções são de 32 bits:
 - Facilidade de buscar e decodificar em um ciclo;
 - X86: 1 a 17 bytes por instrução;
- Poucos e regulares formatos de instrução:
 - Pode decodificar e ler registradores em um único passo;
- Endereçamento de Load/Store:
 - Pode calcular o endereço no 3º estágio, acessar a memória no 4º estágio;
- Alinhamento dos operandos na memória:
 - Acesso à memória gasta apenas 1 ciclo.

Voltando ao exemplo dos 3 loads...

Sem pipeline: O tempo decorrido entre o início da primeira instrução e o início da quarta instrução é $3 \times 8 = 24$ ns

Com pipeline: O tempo decorrido entre o início da primeira instrução e o início da quarta instrução é 14 ($8 + 3 \times 2$) ns

- Melhora de desempenho com pipeline = $24/14 = 1.7$ vezes!
 - Mas o ganho máximo teórico não deveria ser $4x$, já que esse é o número de estágio do pipeline?
 - Onde está a diferença no exemplo?
- E se agora fossem realizadas 1.000.003 instruções load? Qual o tempo total de execução com e sem pipeline? E o ganho
- Sem pipeline: $1.000.003 \times 8 = 8.000.024$ ns
- Com pipeline: $8 + 1.000.003 \times 2 = 2.000.014$ ns
- Ganho = 4.0

Propriedades da arquitetura MIPS:

- Todos os operandos estão em registradores (32-bits).
- Acesso à Memória somente por operações load /store
- Instruções de tamanho único (32 bits)
- Poucos formatos de instruções, sempre com o registrador-fonte localizado na mesma posição
- Operandos do MIPS alinhados na memória

- Conjunto de instruções pensado para o pipeline!!!
- Instruções aritméticas (ULA), acesso a memória e mudança de fluxo de controle (desvios)

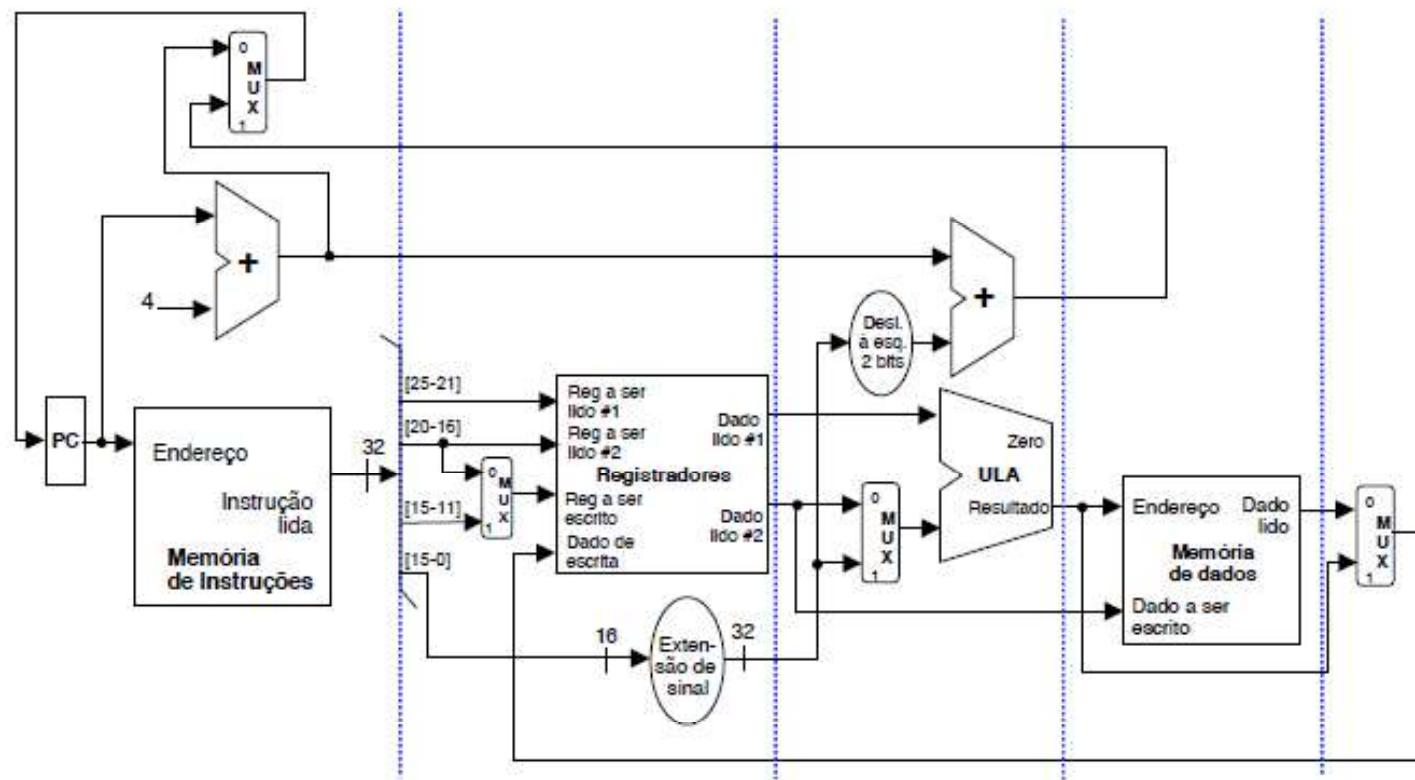
Tipos de Instruções MIPS

- ULA (ou tipo R)
 - Aritméticas com 2 registradores ou 1 registrador e um imediato como operandos. Resultado escrito num registrador destino
- Load/store
 - Um registrador (base) e um operando 16-bits como valor imediato. A soma deles define o endereço efetivo para o acesso à memória.
 - Um segundo registrador: destino para a operação de load e dado a ser armazenado na operação de stored.
- Branches e Jumps
 - Mudanças de fluxo de controle. Um branch resulta na soma de um valor imediato ao valor atual do PC.

Organizações do MIPS: pipeline

Bloco Operativo dos MIPS Monociclo

É possível identificar 5 etapas na execução



5 passos para execução de instruções MIPS

Assume-se que qualquer instrução MIPS pode ser executada em no máximo 5 ciclos de clock:

- ① Busca da instrução na memória (BI/IF - Instruction Fetch)
- ② Leitura dos registradores, enquanto a instrução é decodificada (ID/DI - Decode Instruction)
- ③ Execução de uma operação ou cálculo de um endereço (EX - Execution)
- ④ Acesso a um operando na memória (MEM - Memory Acess)
- ⑤ Escrita do resultado em um registrador (EM/WB - write back)

Considerando cada passo como um estágio, até cinco instruções podem estar em execução durante um dado ciclo de clock.

Pensando em ciclos de clock

Resumo dos ciclos para executar instruções numa CPU não pipeline:

- Jump, Branch - 3 clocks
- Formato R, Store - 4 ciclos
- Load - 5 ciclos

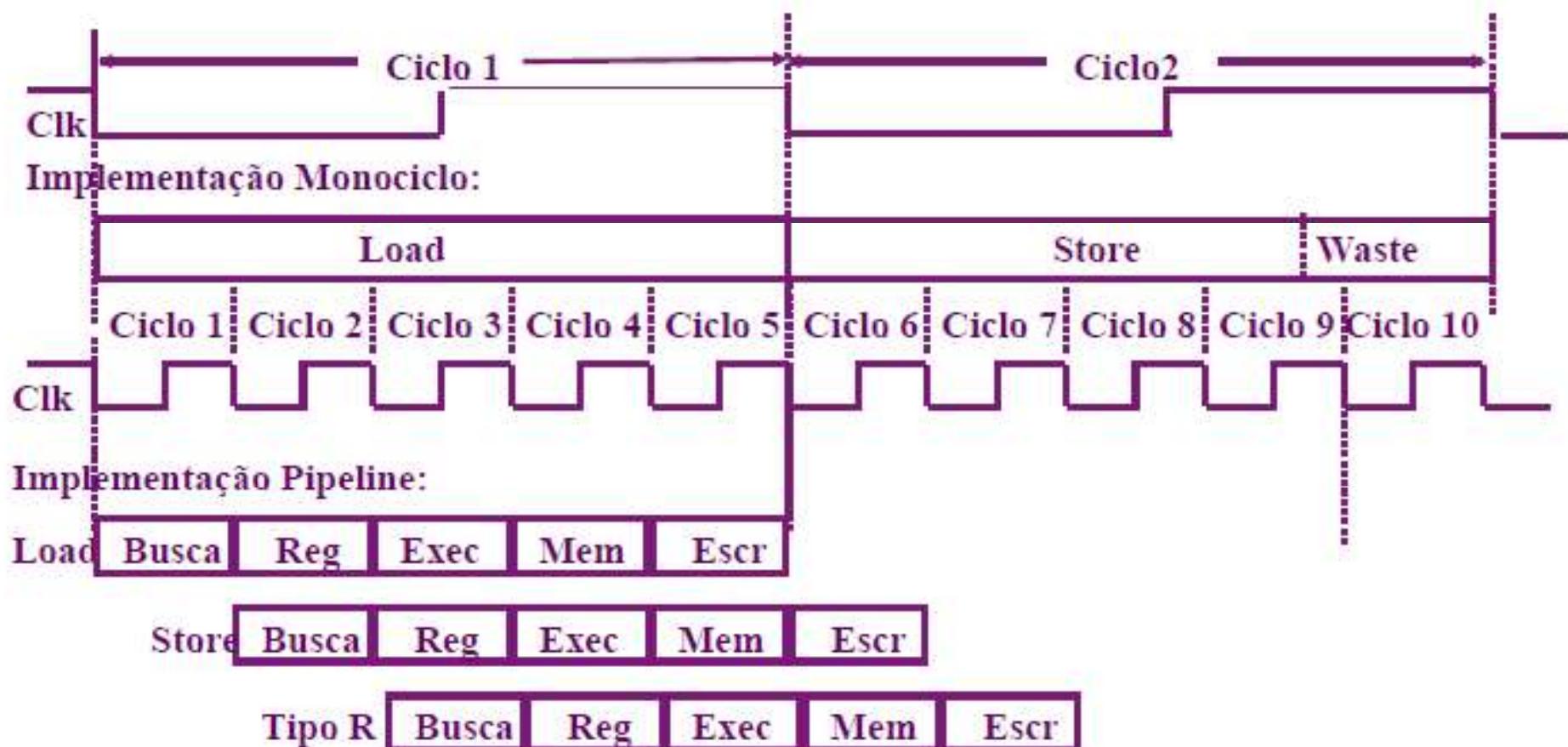
Assumindo que as instruções de desvio correspondem a 12% de todas as instruções e que stores a 10%, qual o CPI médio da CPU?

$$CPI_{\text{médio}} = 0.12 * 3 + 0.10 * 4 + 0.78 * 5 = 4.66$$

Pipeline RISC de 5 estágios

- Implementação clássica do conceito de pipeline proposta por Hennessy e Patterson para o DLX (MIPS)
- Para o caso ideal, o pipeline deveria começar uma nova instrução a cada novo ciclo de clock.
- Infelizmente isso nem sempre é possível:
 - Um ADD não gasta o mesmo tempo de ULA que um MULT
 - Dependências de dados entre instruções
 - ...
- Mas cada estágio pode ser tratado de forma independente e assim as fases de execução podem ser “superpostas”

Monociclo vs Pipeline



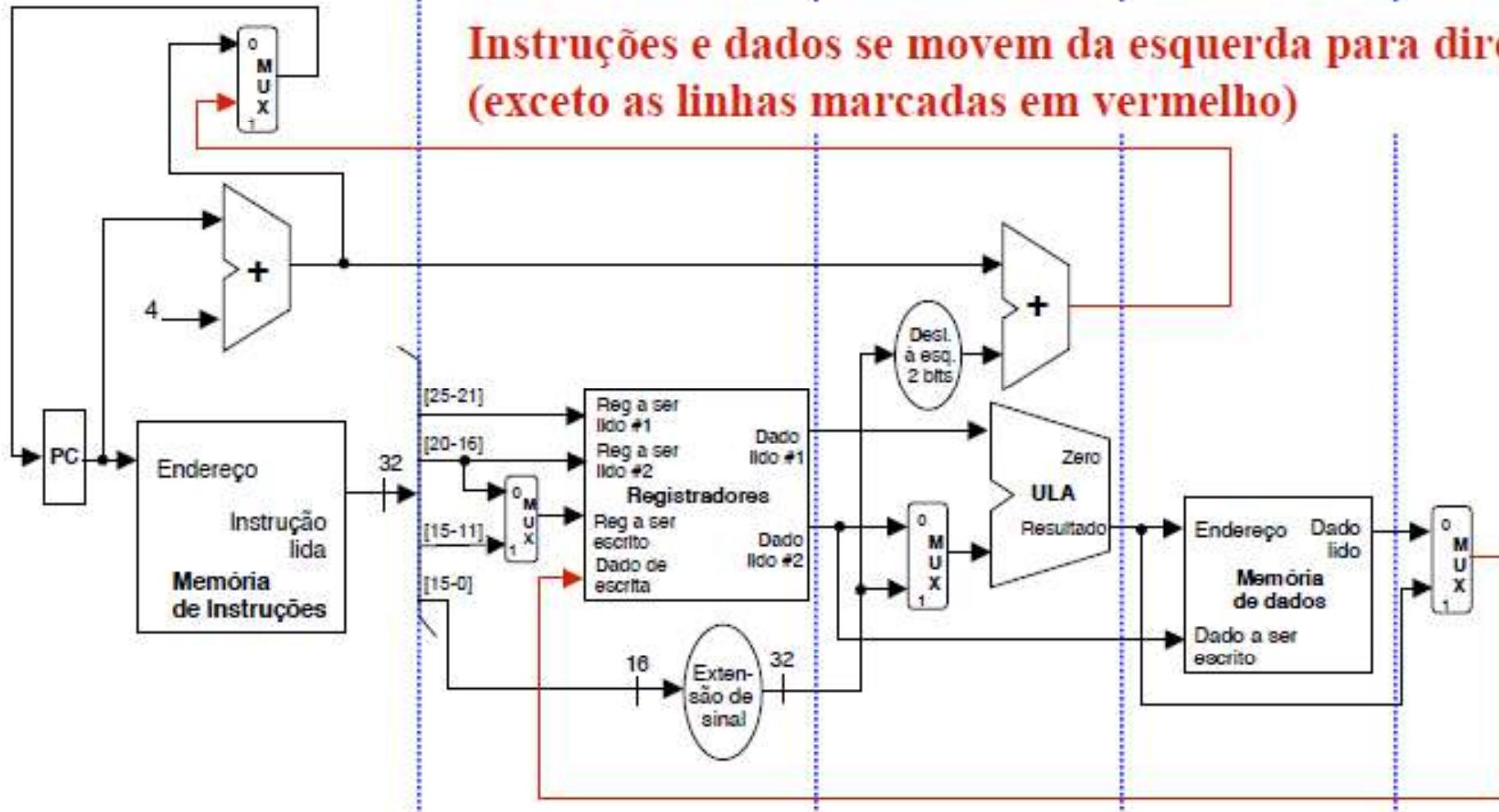
Organização do MIPS: Pipeline

BI: Busca da instrução

DI: Decodif. da instrução/
Leitura do banco de regs.

EX: Execução/
Cálculo do endereço

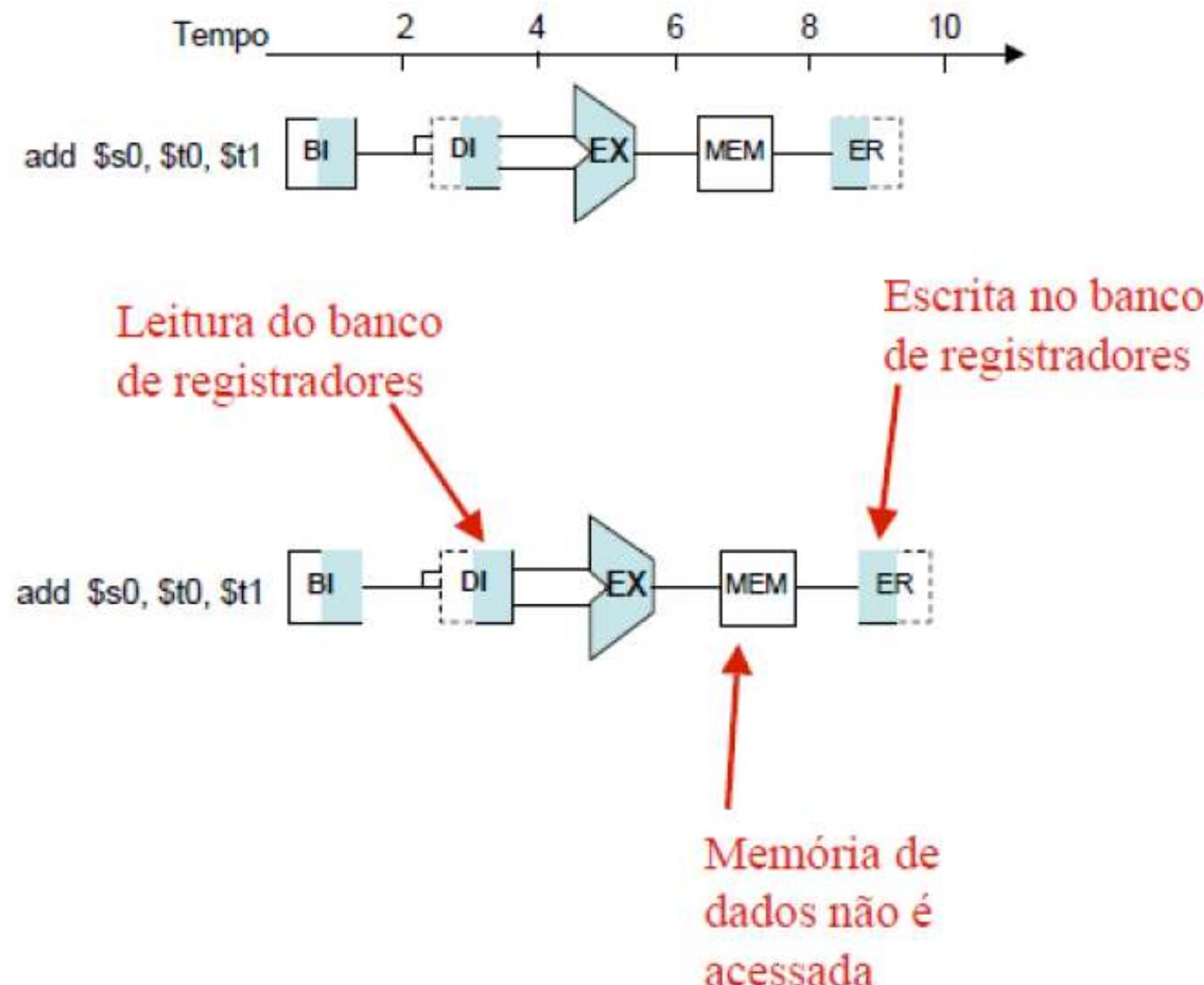
MEM: acesso à mem.
ER: escrita no banco
de registradores



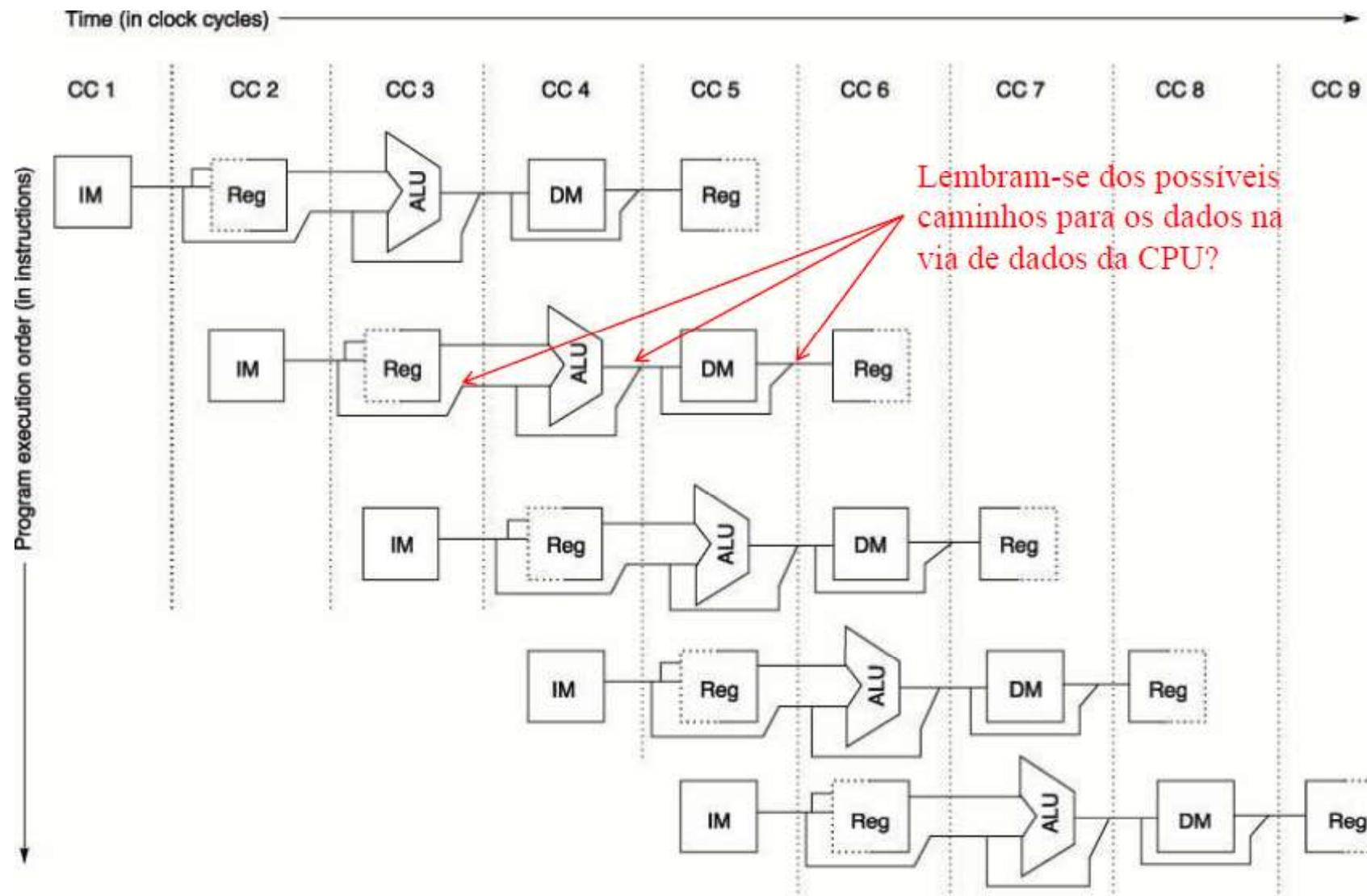
Bloco Operativo em Pipeline

- Um modo de mostrar a execução em pipeline é imaginar que cada instrução executa em seu próprio bloco operativo
- Os blocos operativos são colocados deslocados uns em relação aos outros, a fim de mostrar a relação entre as instruções

Representação Gráfica do Pipeline de Instrução



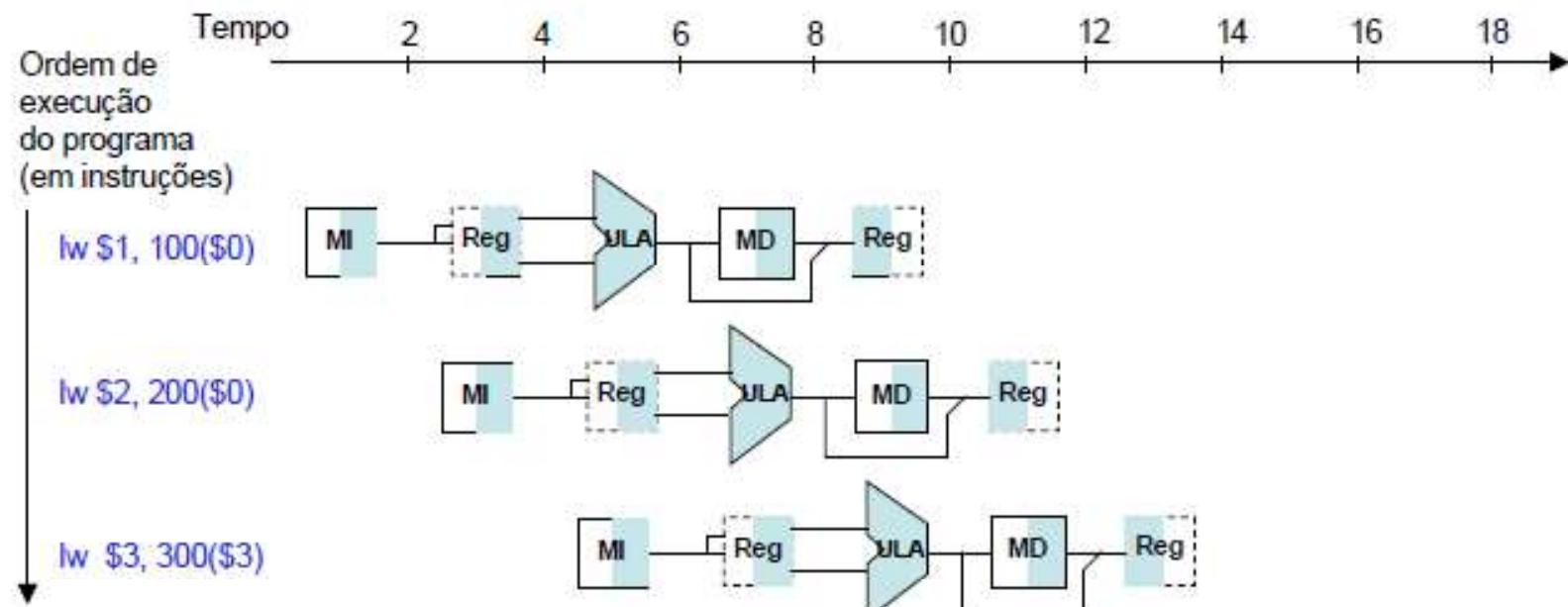
Representação Gráfica do Pipeline de Instrução



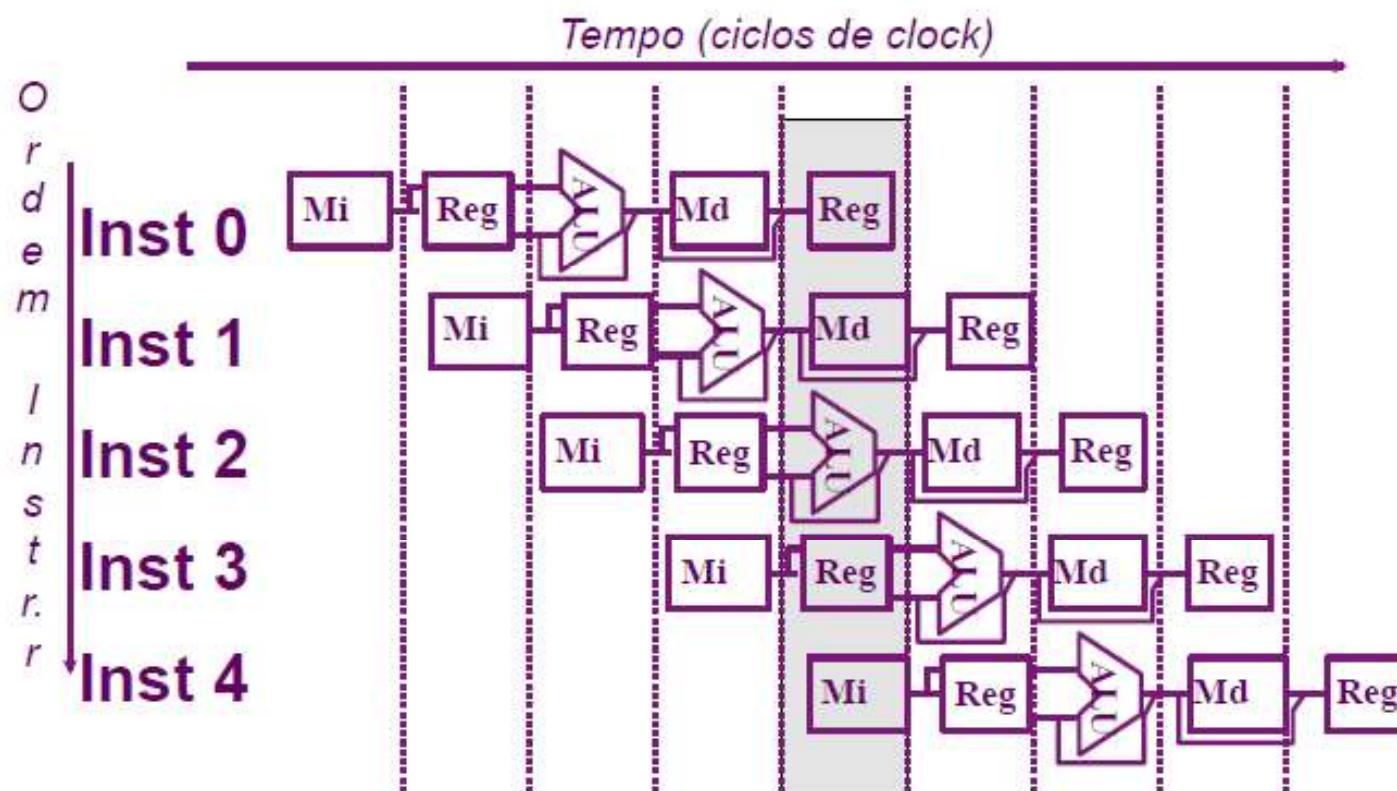
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline

Execução das 3 instruções `Iw` pressupondo o uso de pipeline

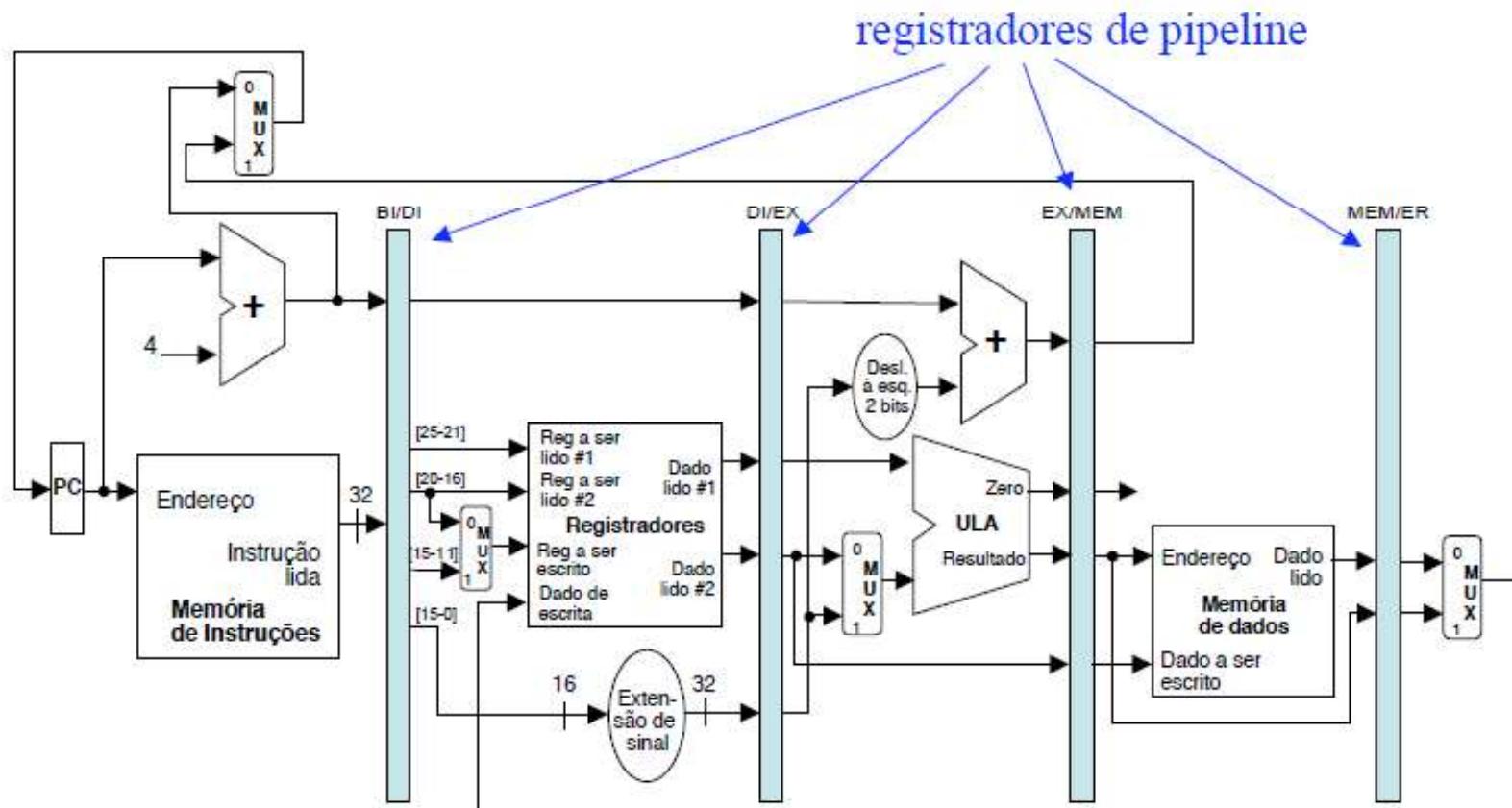


Pipeline - Recursos disponíveis



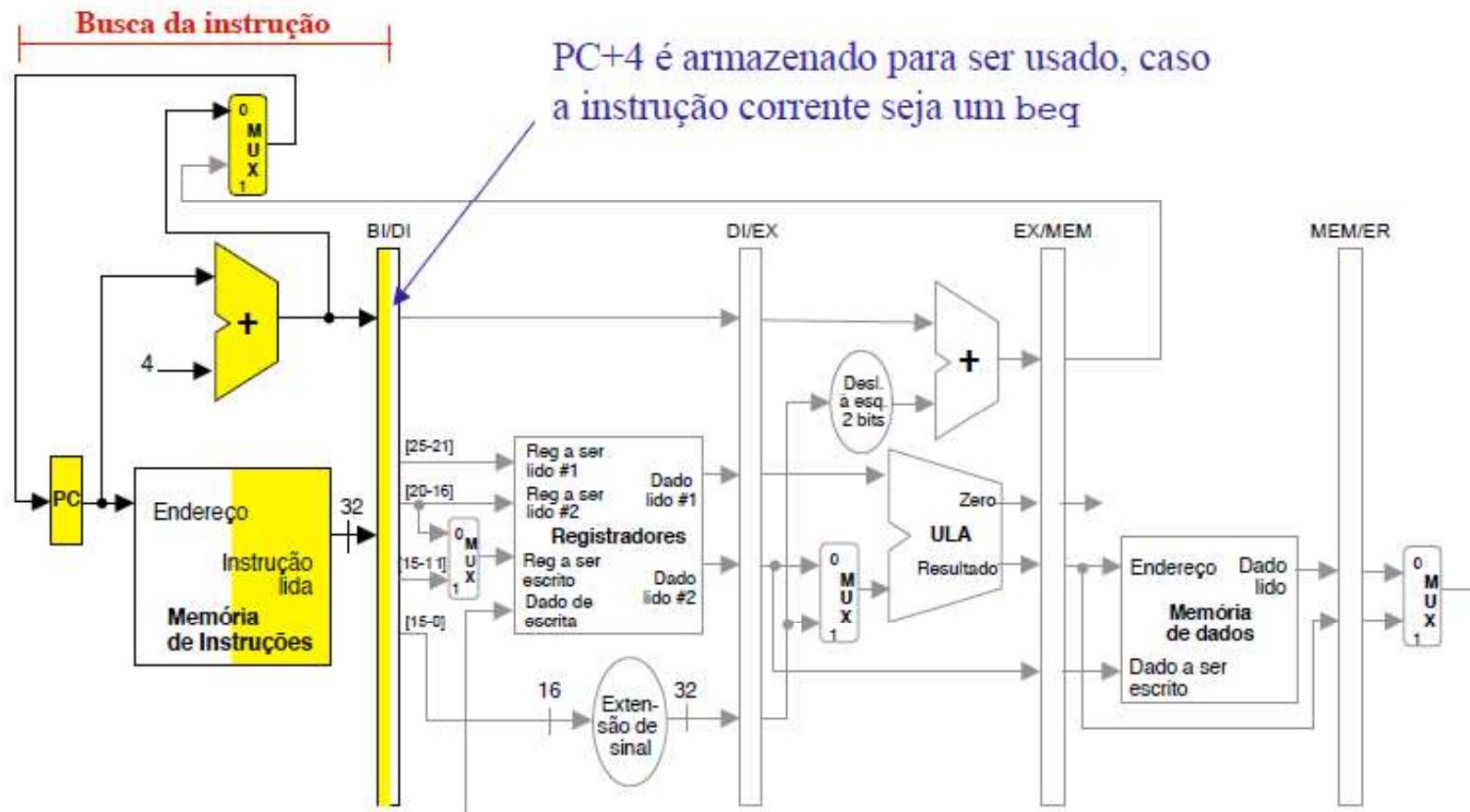
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline



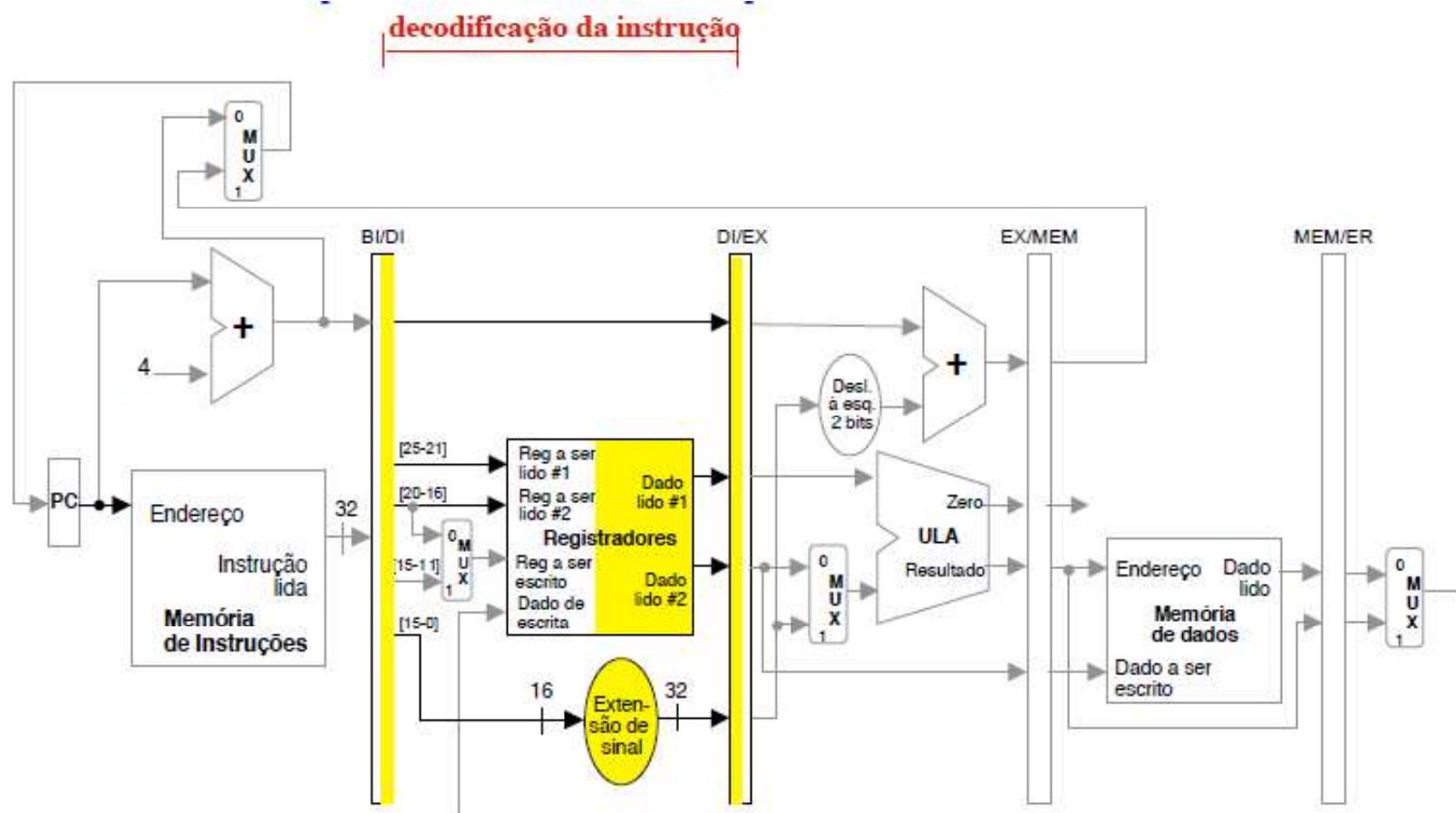
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando lw



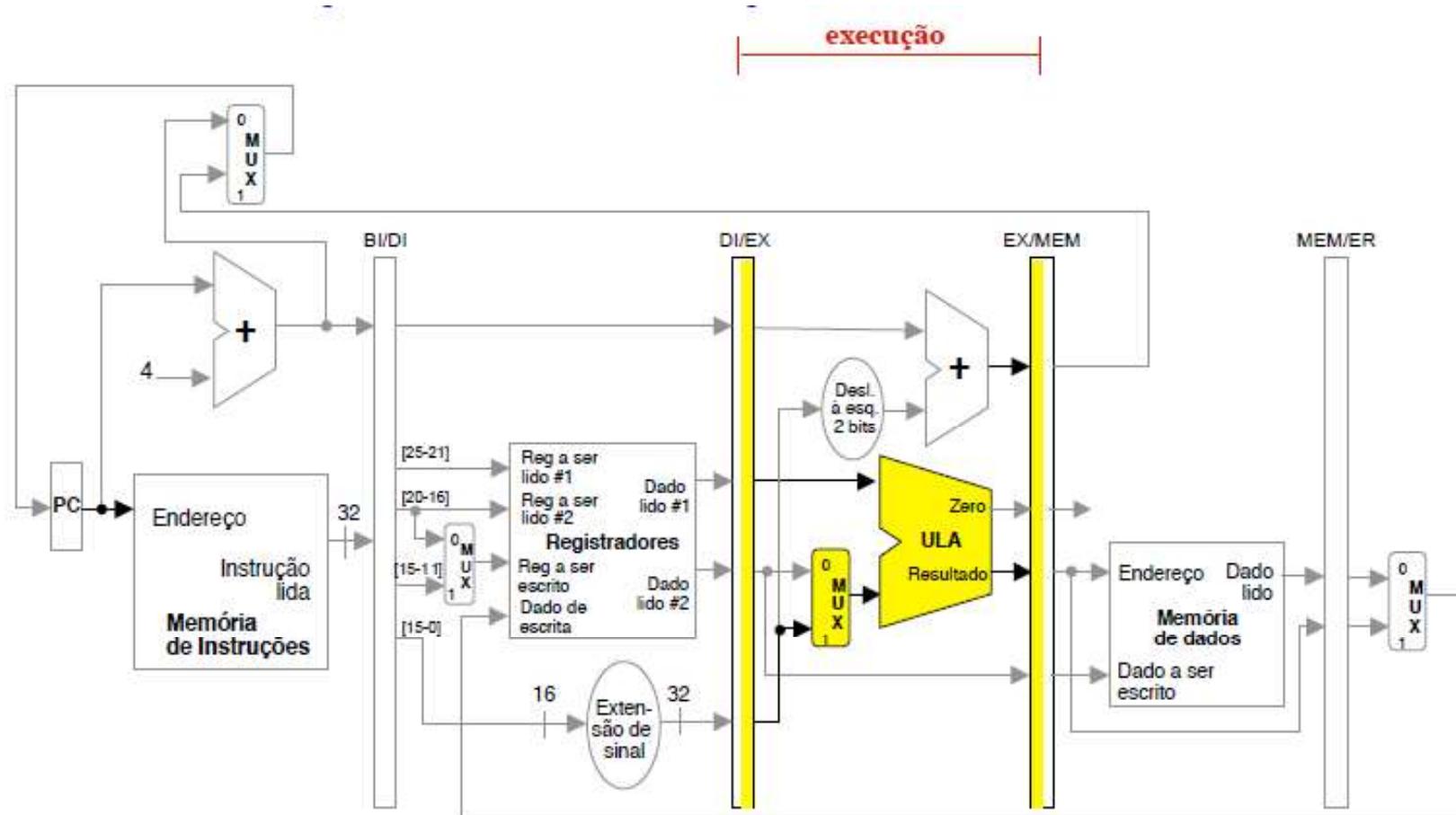
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando lw



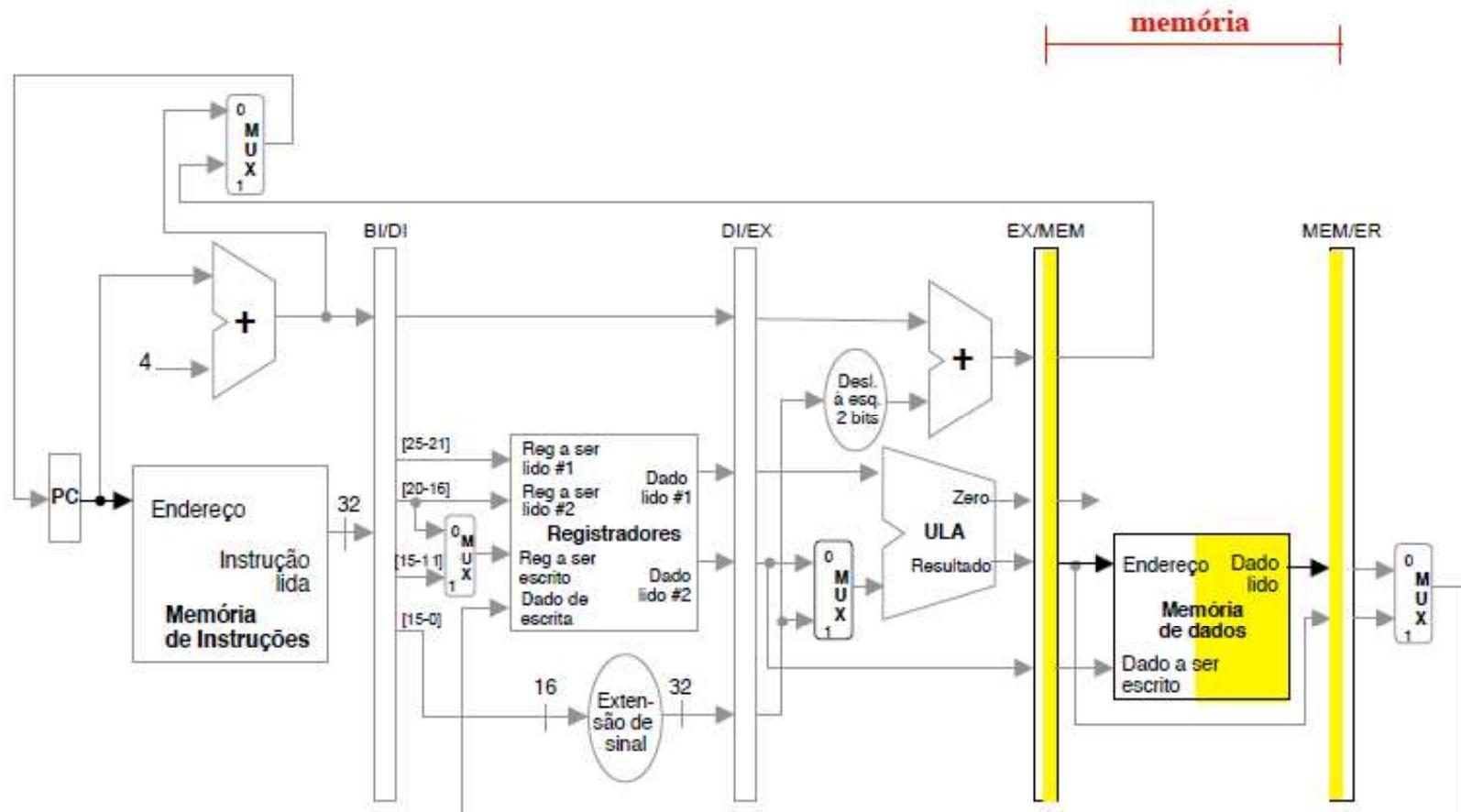
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando lw



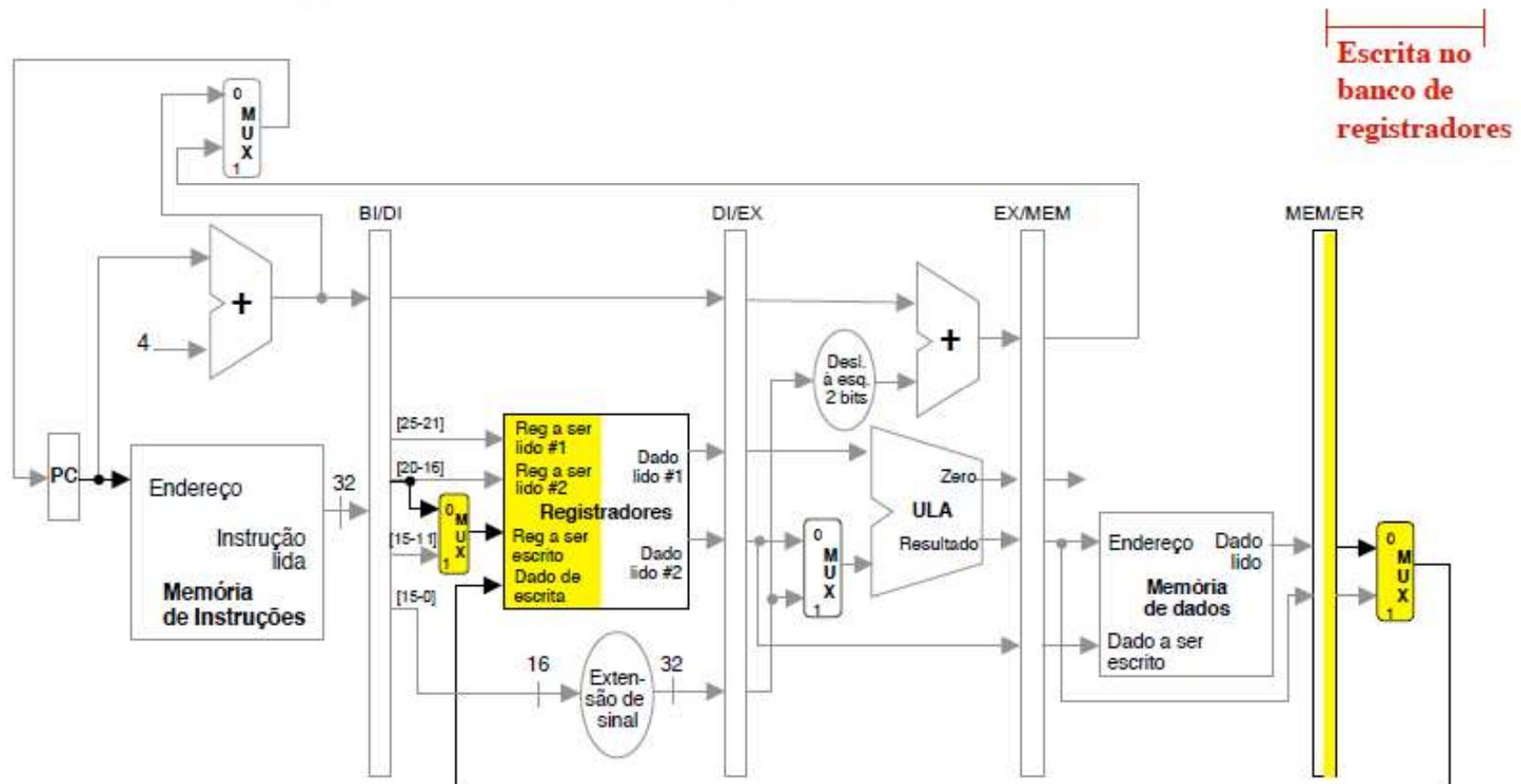
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando lw



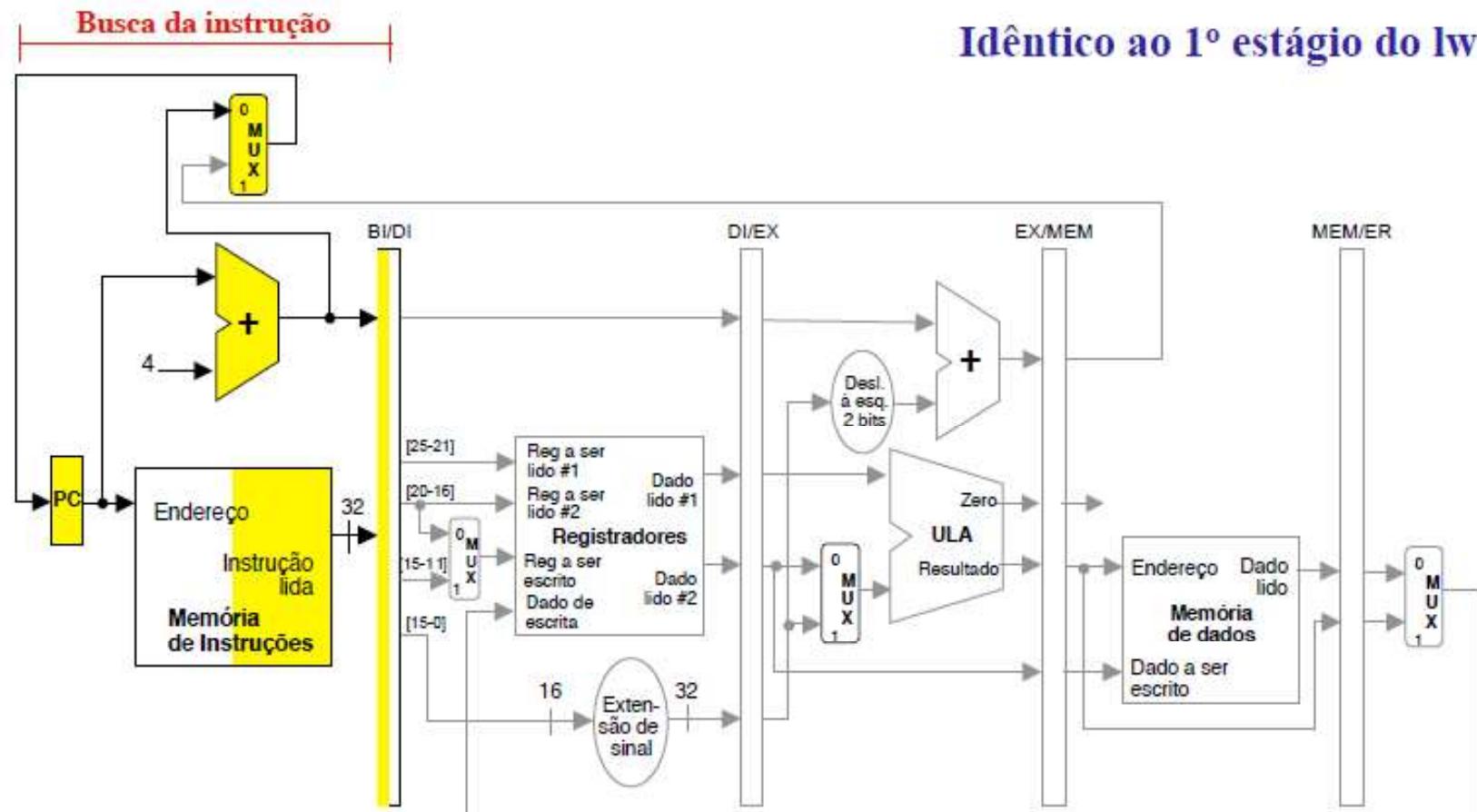
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando lw



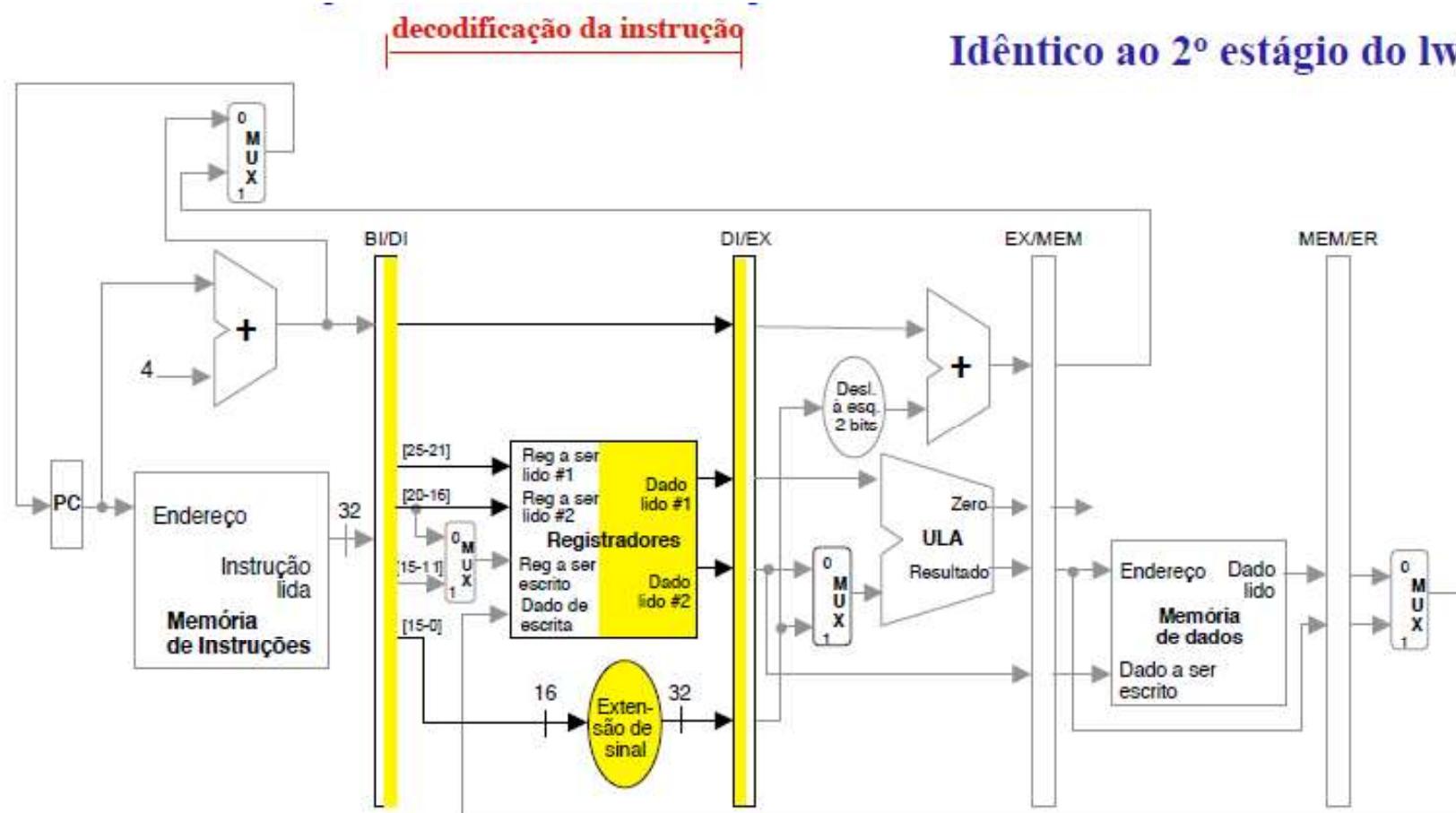
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando sw



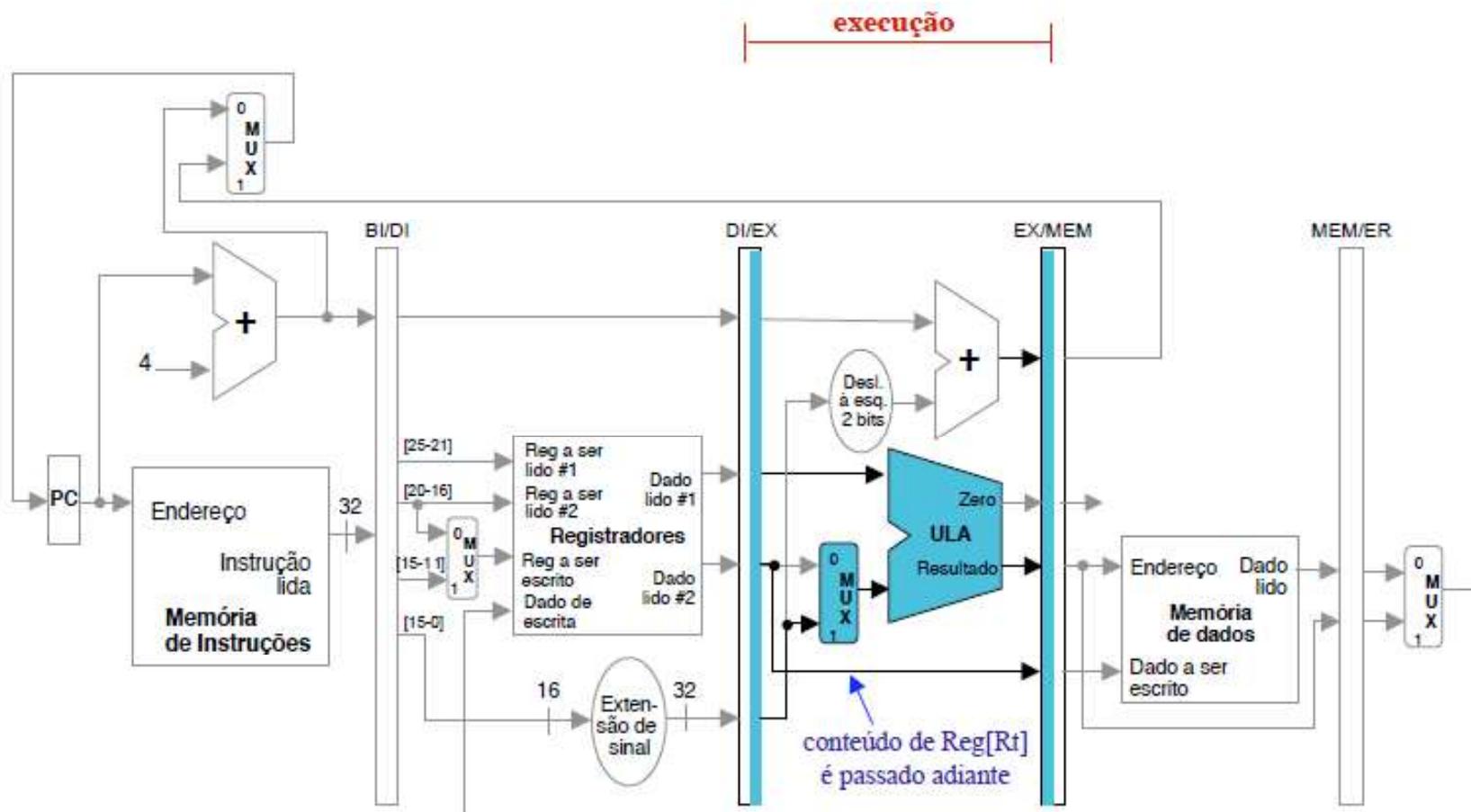
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando sw



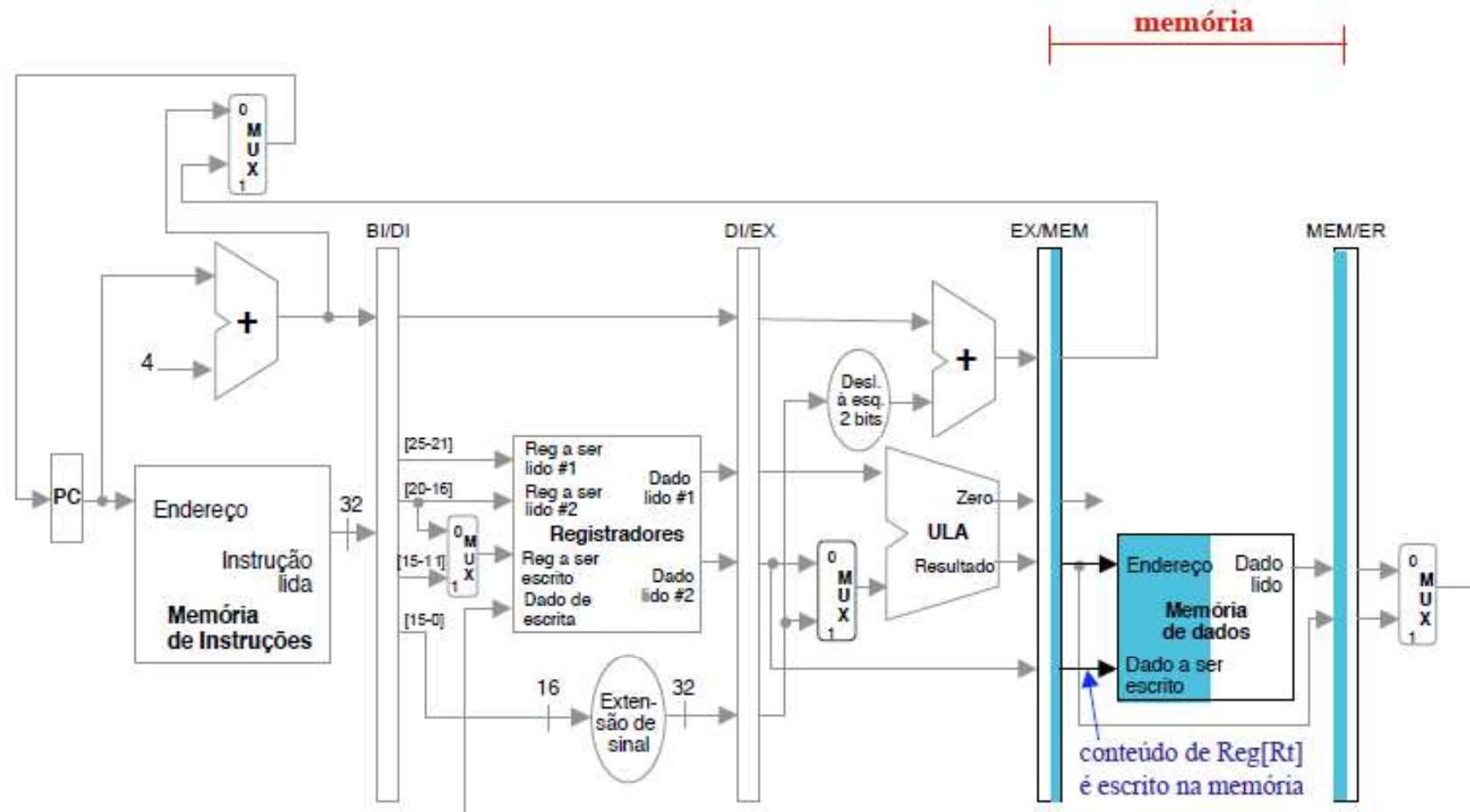
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando sw



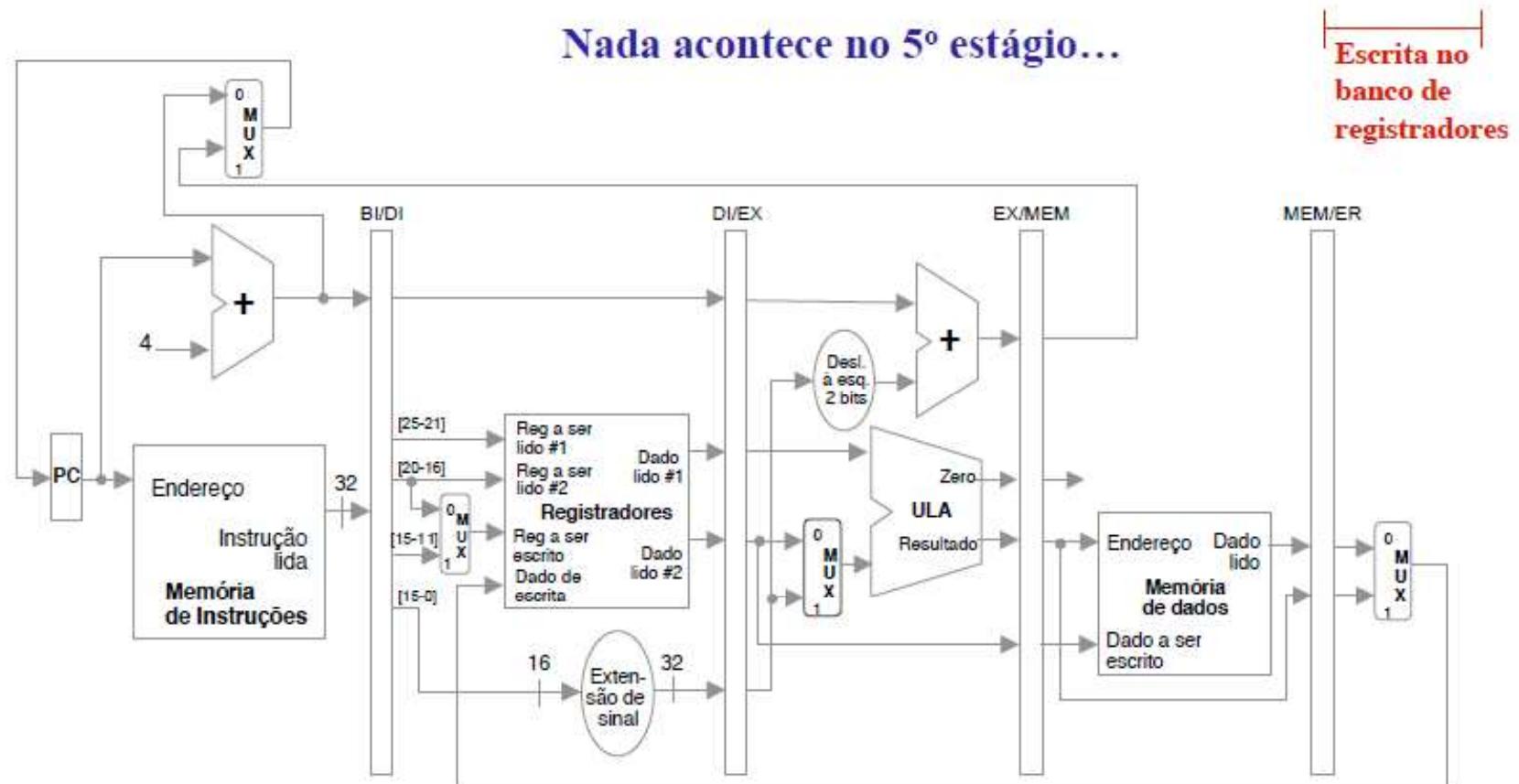
Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando sw



Organizações do MIPS: pipeline

Bloco Operativo em Pipeline: executando sw

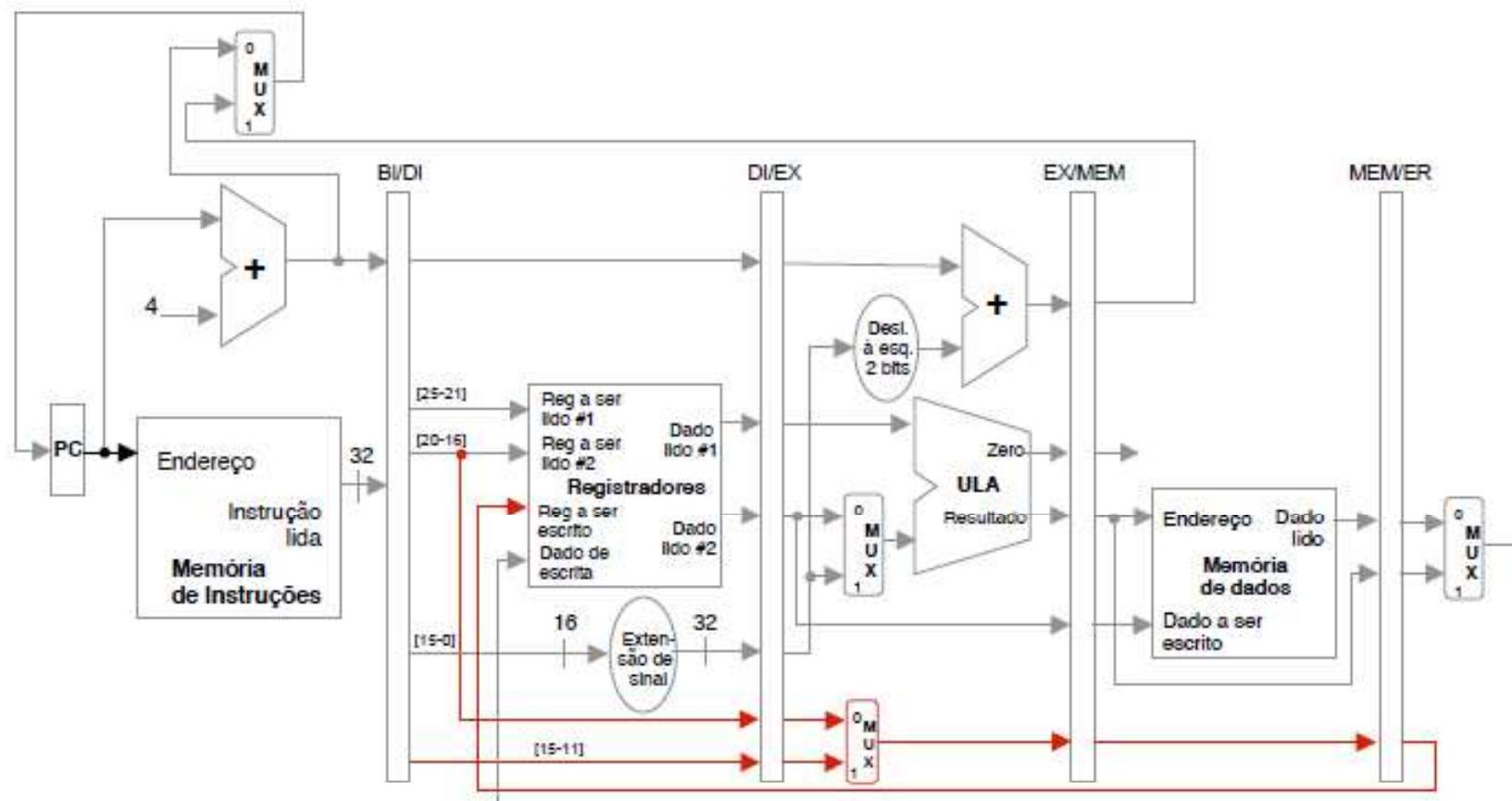


Bloco Operativo em Pipeline: executando sw

- Cada componente no bloco operativo só pode ser usado em um único estágio do pipeline
- Componentes:
 - Memória de instruções
 - Portas de leitura do banco de registradores
 - ULA
 - Memória de dados
 - Porta de escrita do banco de registradores

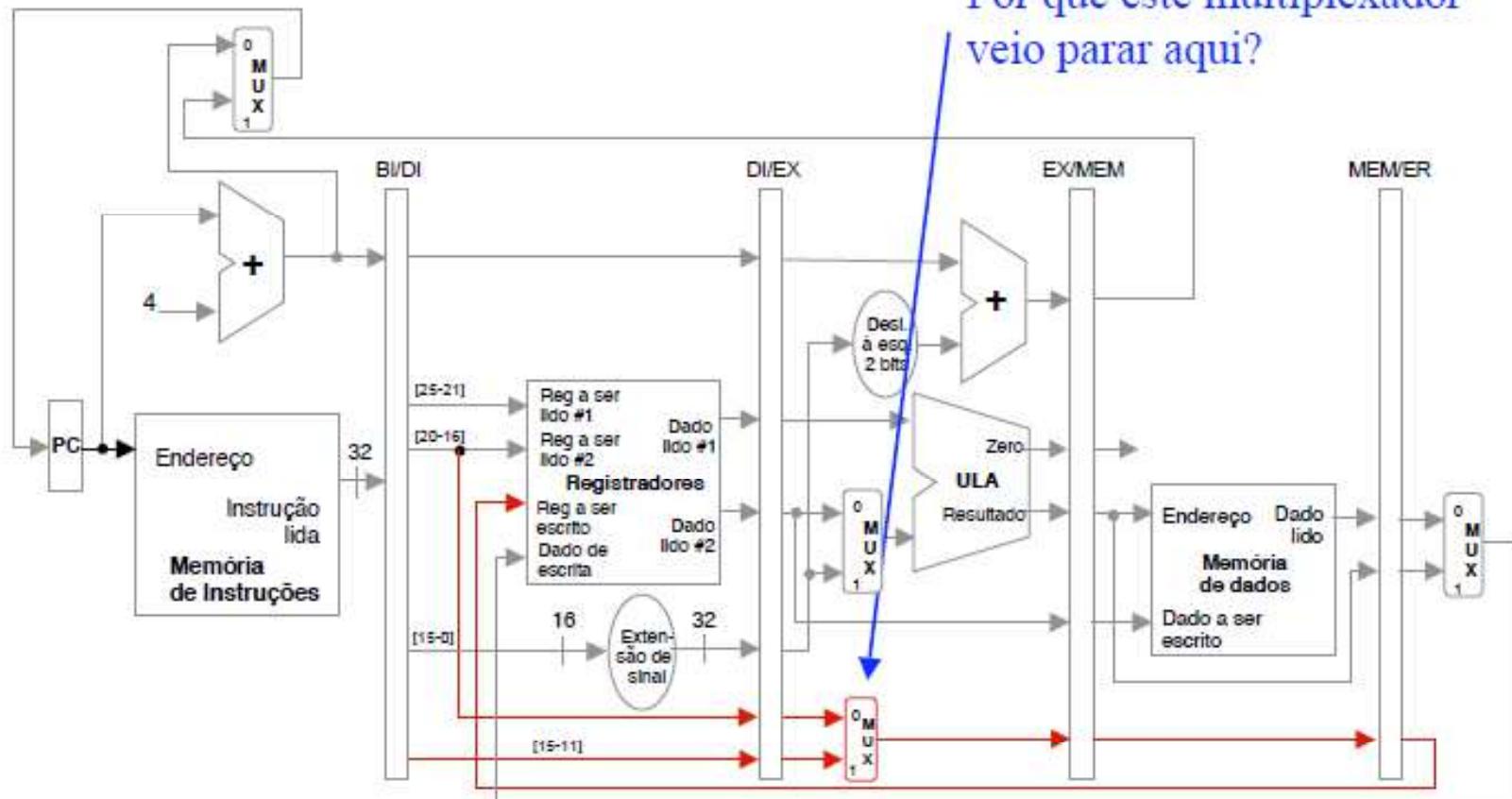
Organizações do MIPS: pipeline

Bloco Operativo Pipeline Corrigido



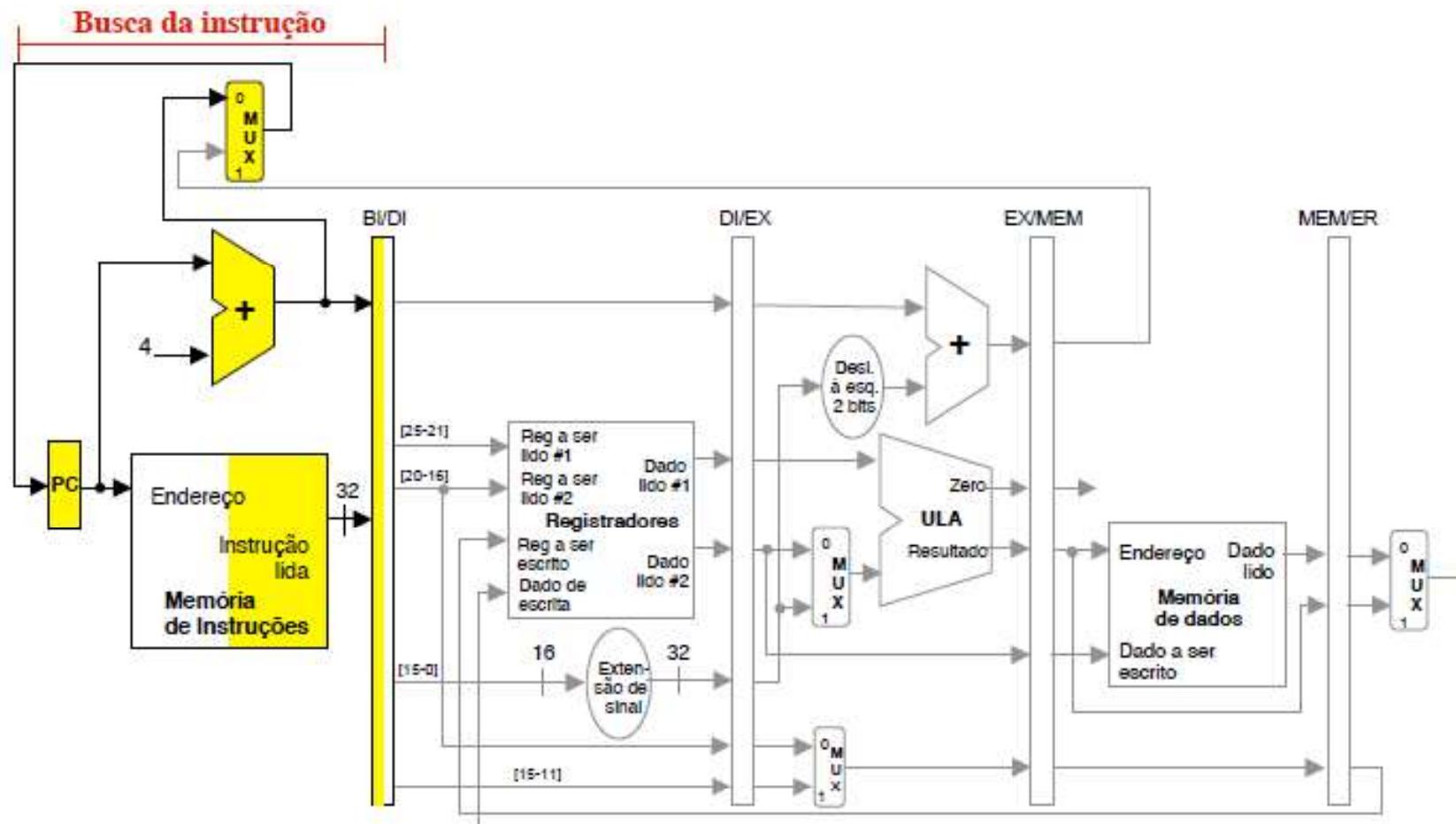
Organizações do MIPS: pipeline

Bloco Operativo Pipeline Corrigido



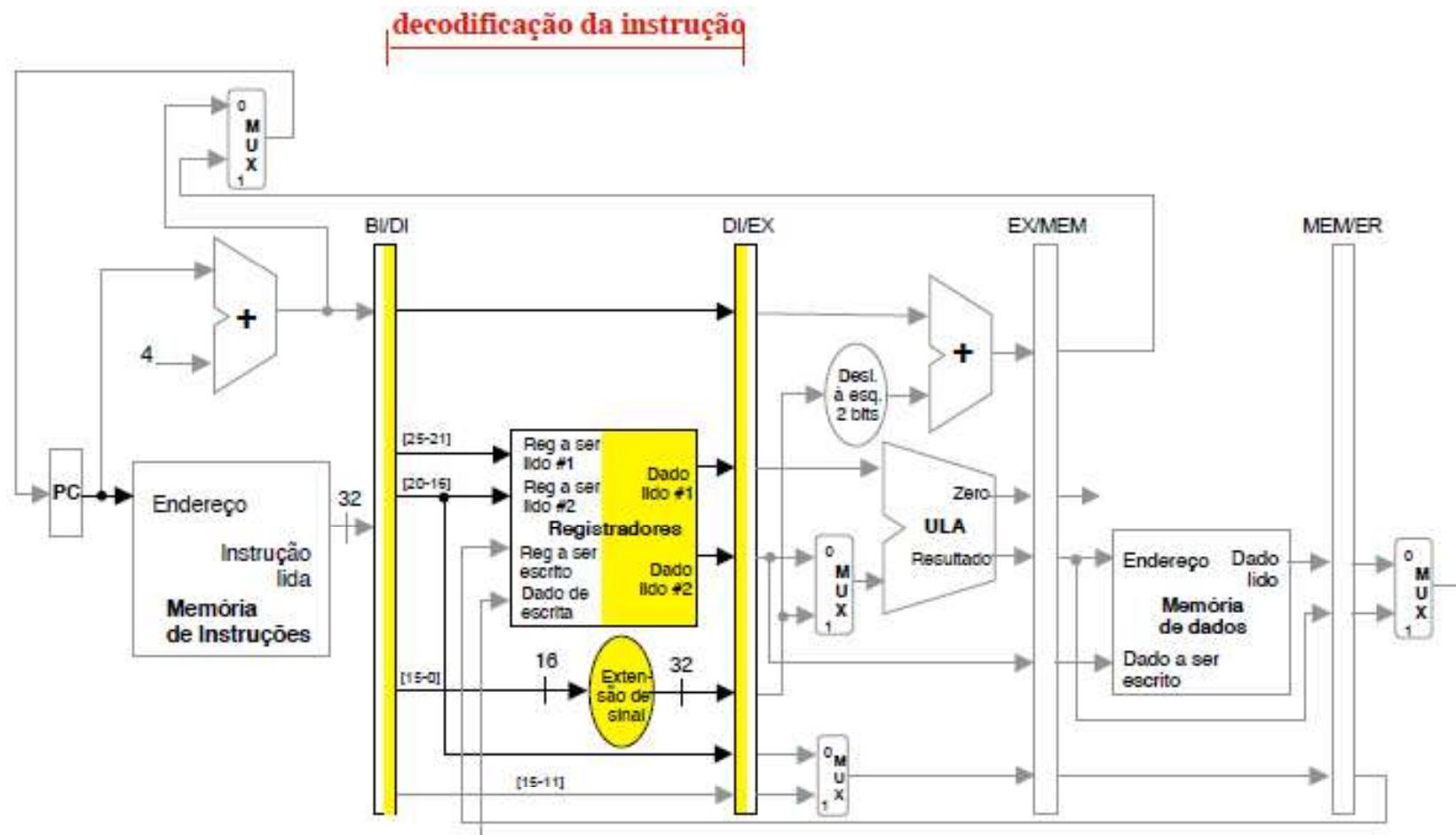
Organizações do MIPS: pipeline

Bloco Operativo Pipeline Corrigido: executando lw



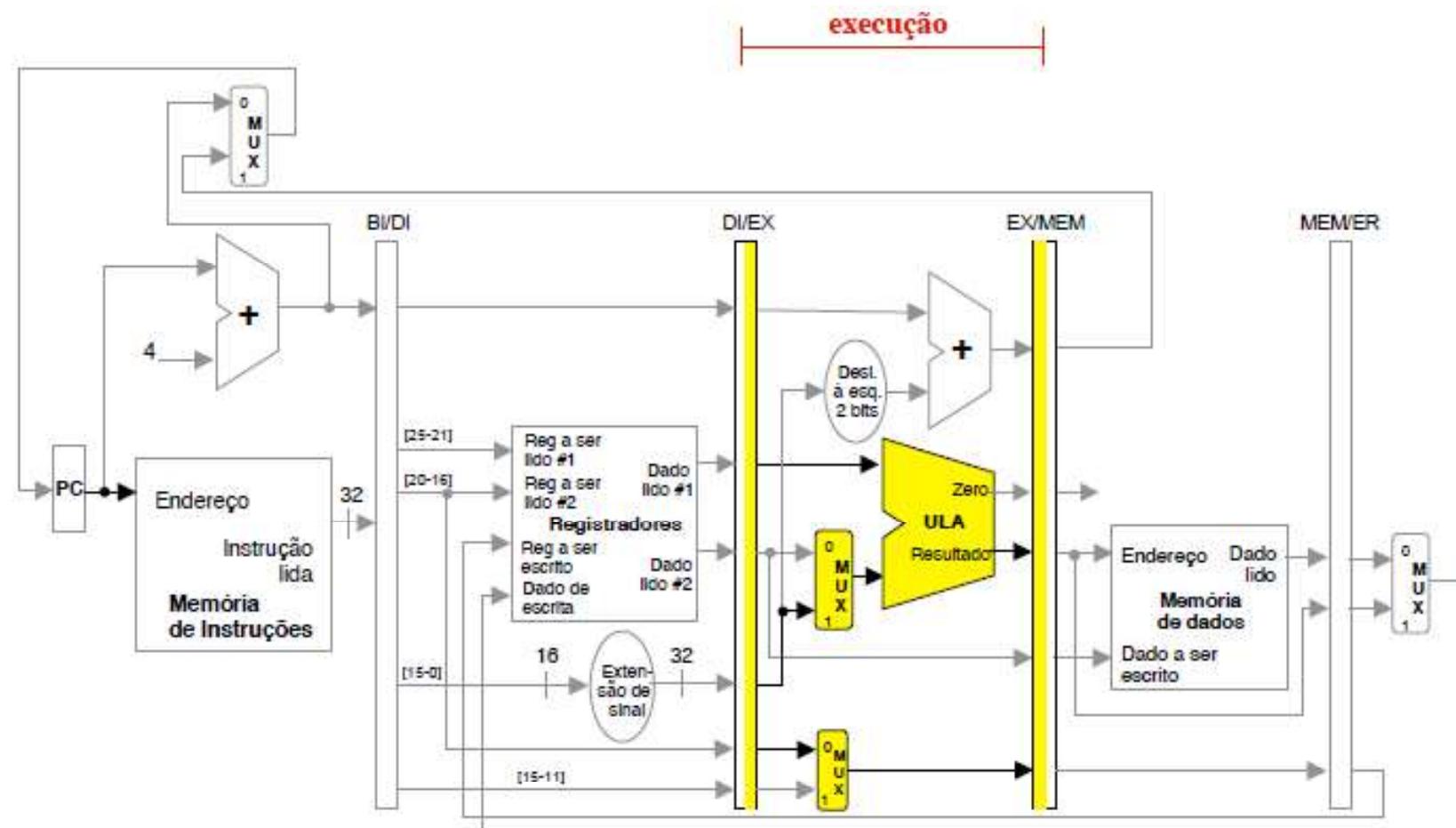
Organizações do MIPS: pipeline

Bloco Operativo Pipeline Corrigido: executando lw



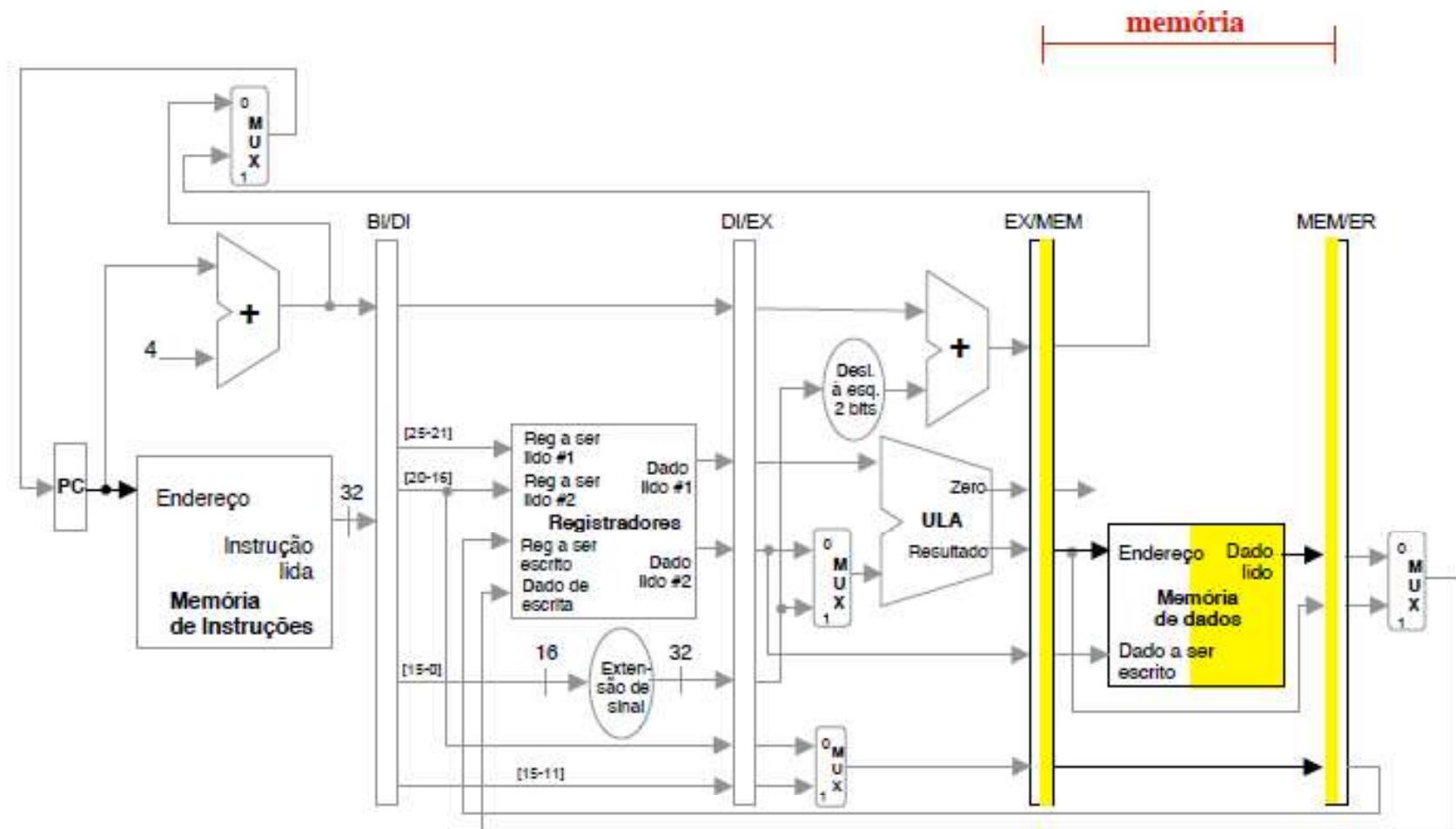
Organizações do MIPS: pipeline

Bloco Operativo Pipeline Corrigido: executando lw



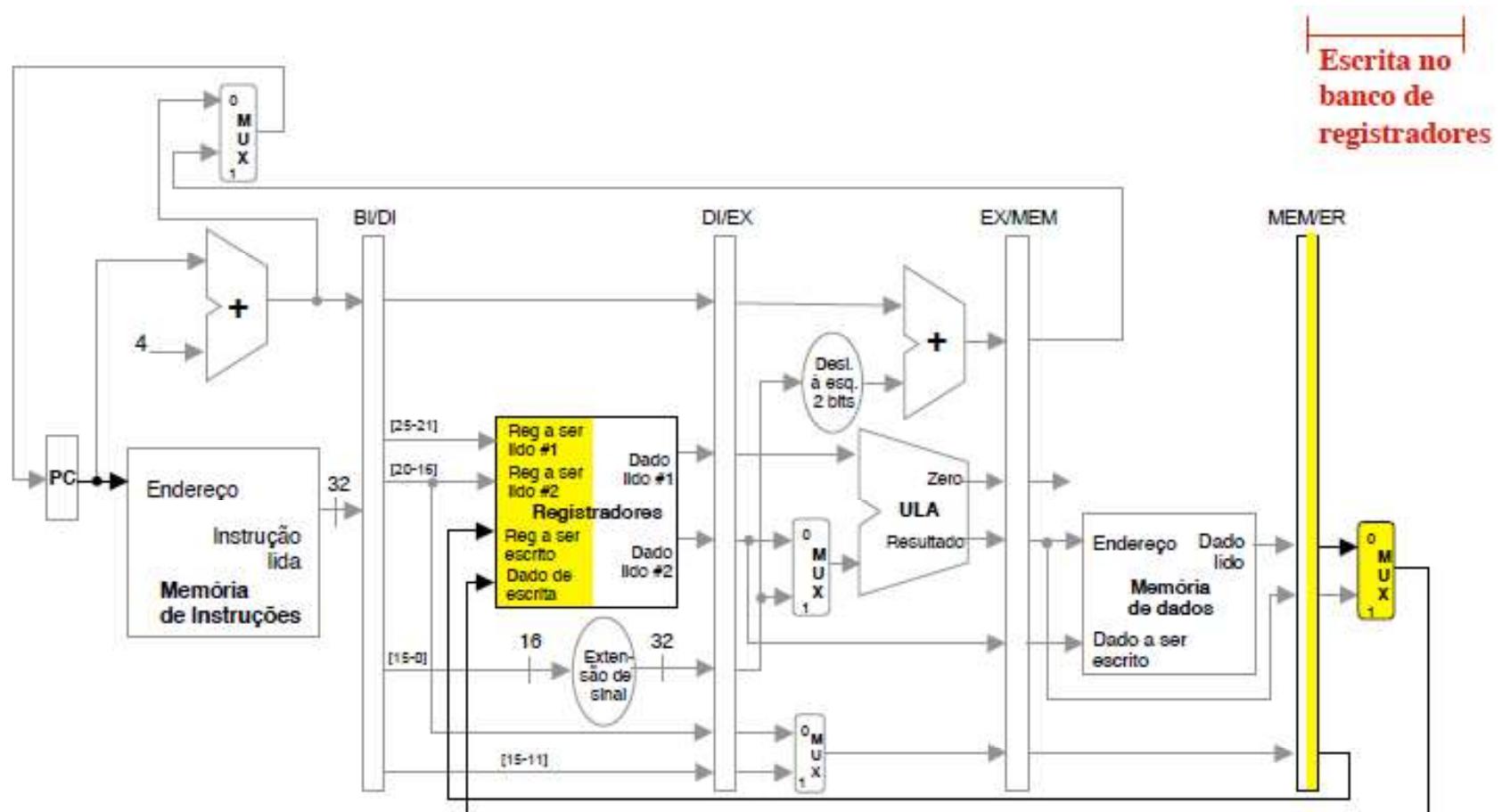
Organizações do MIPS: pipeline

Bloco Operativo Pipeline Corrigido: executando lw



Organizações do MIPS: pipeline

Bloco Operativo Pipeline Corrigido: executando lw



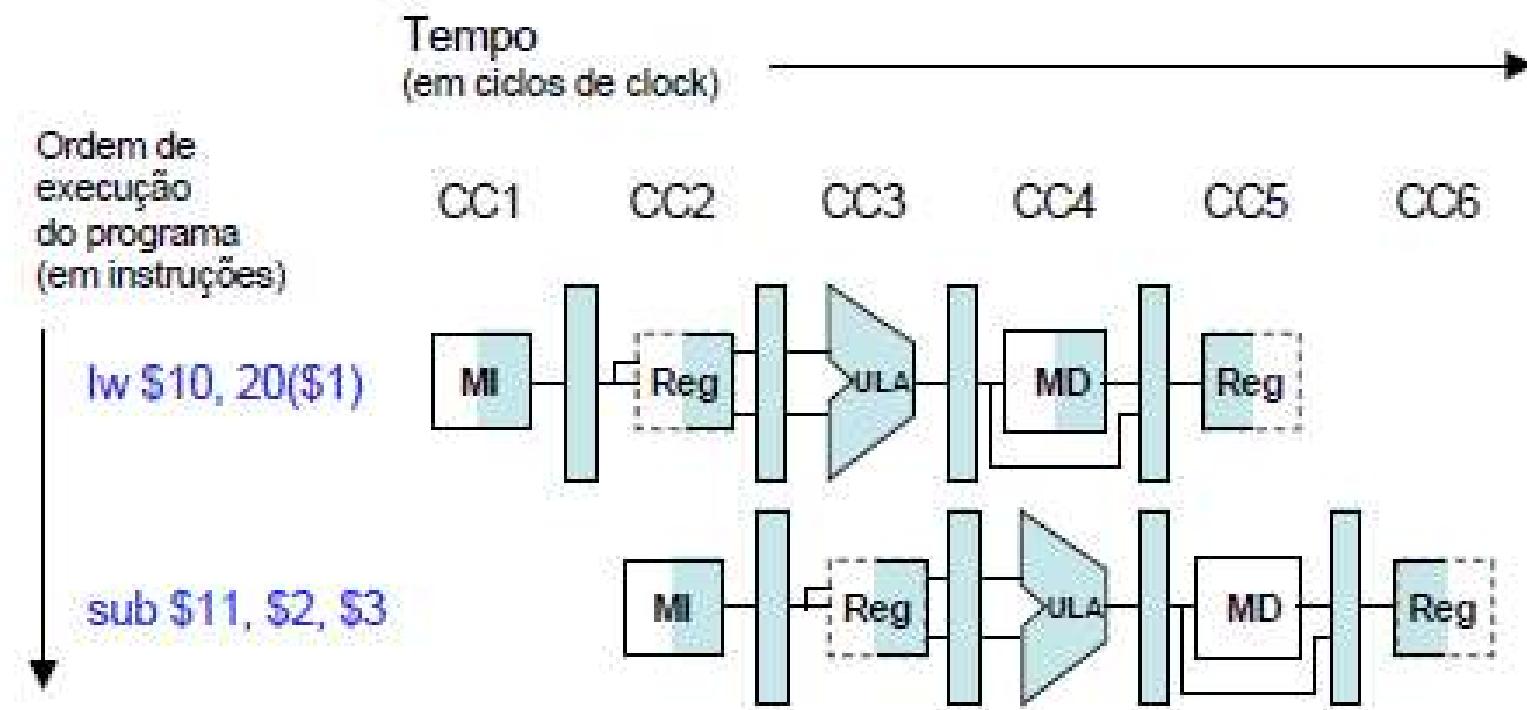
Executando uma sequência de instruções

- Considere a seguinte sequência de instruções:
lw \$10, 20(\$1)
sub \$11, \$2, \$3
- Representá-la usando diagrama de pipeline com múltiplos ciclos de clock (relógio)

Organizações do MIPS: pipeline

Executando uma sequência de instruções

- Diagrama de Pipeline com Múltiplos Ciclos de Clock: versão 1

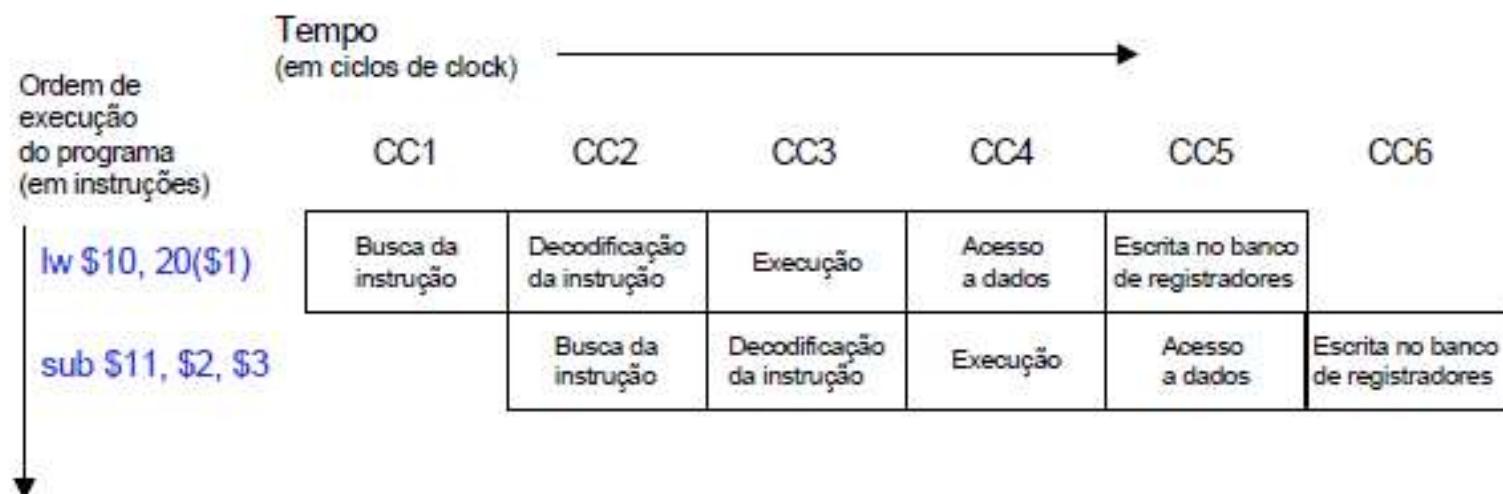


- Diagrama não-convencional
- Os recursos usados em cada estágio estão identificados

Organizações do MIPS: pipeline

Executando uma sequência de instruções

- Diagrama de Pipeline com Múltiplos Ciclos de Clock: versão 2



- Diagrama tradicional
- Identifica cada estágio pelo nome

Executando uma sequência de instruções

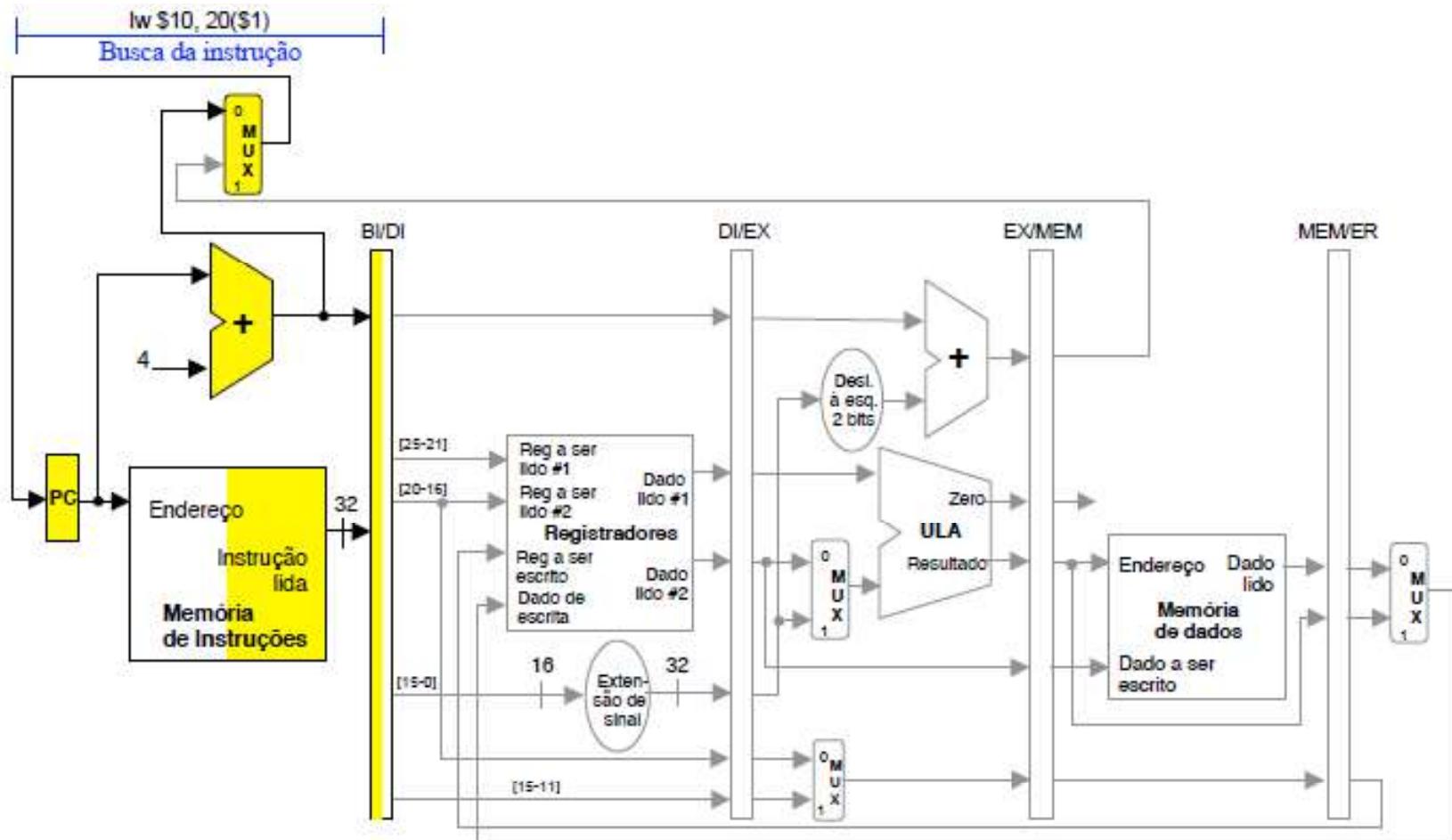
- Representa a sequência de instruções abaixo usando diagrama de pipeline com único ciclo de clock (relógio)

lw \$10, 20(\$1)

sub \$11, \$2,\$3

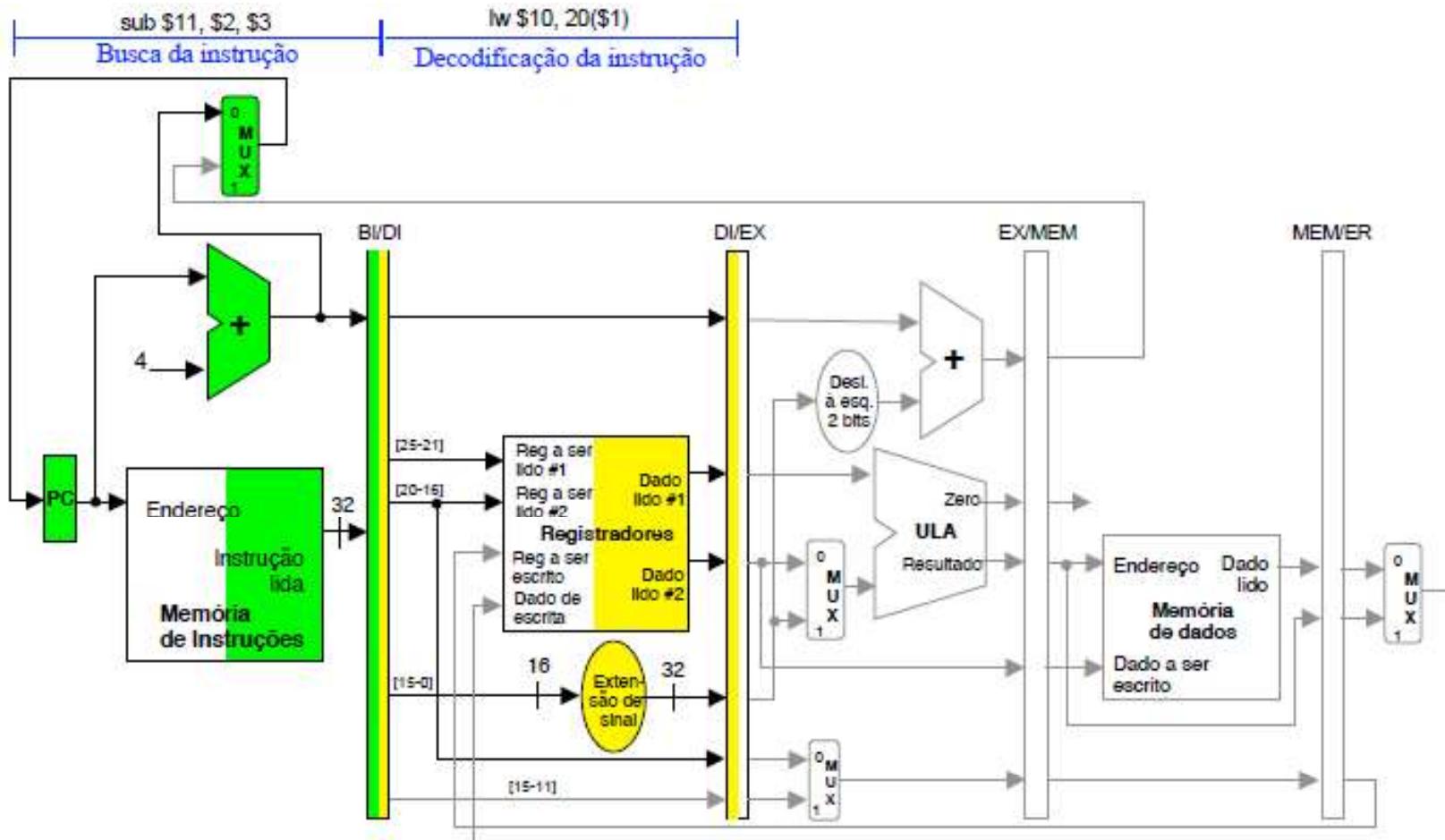
Organizações do MIPS: pipeline

Executando uma sequência de instruções



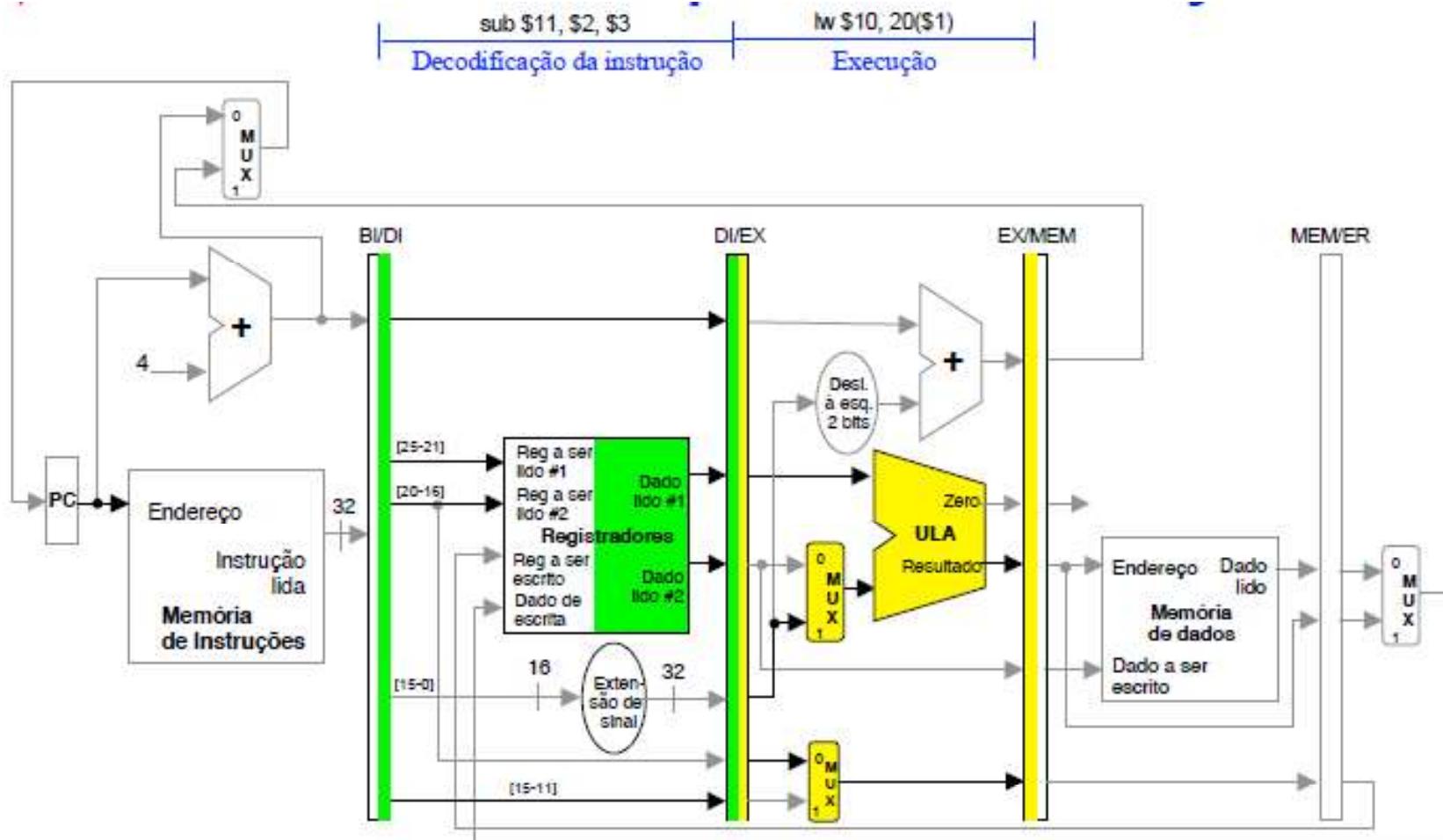
Organizações do MIPS: pipeline

Executando uma sequência de instruções



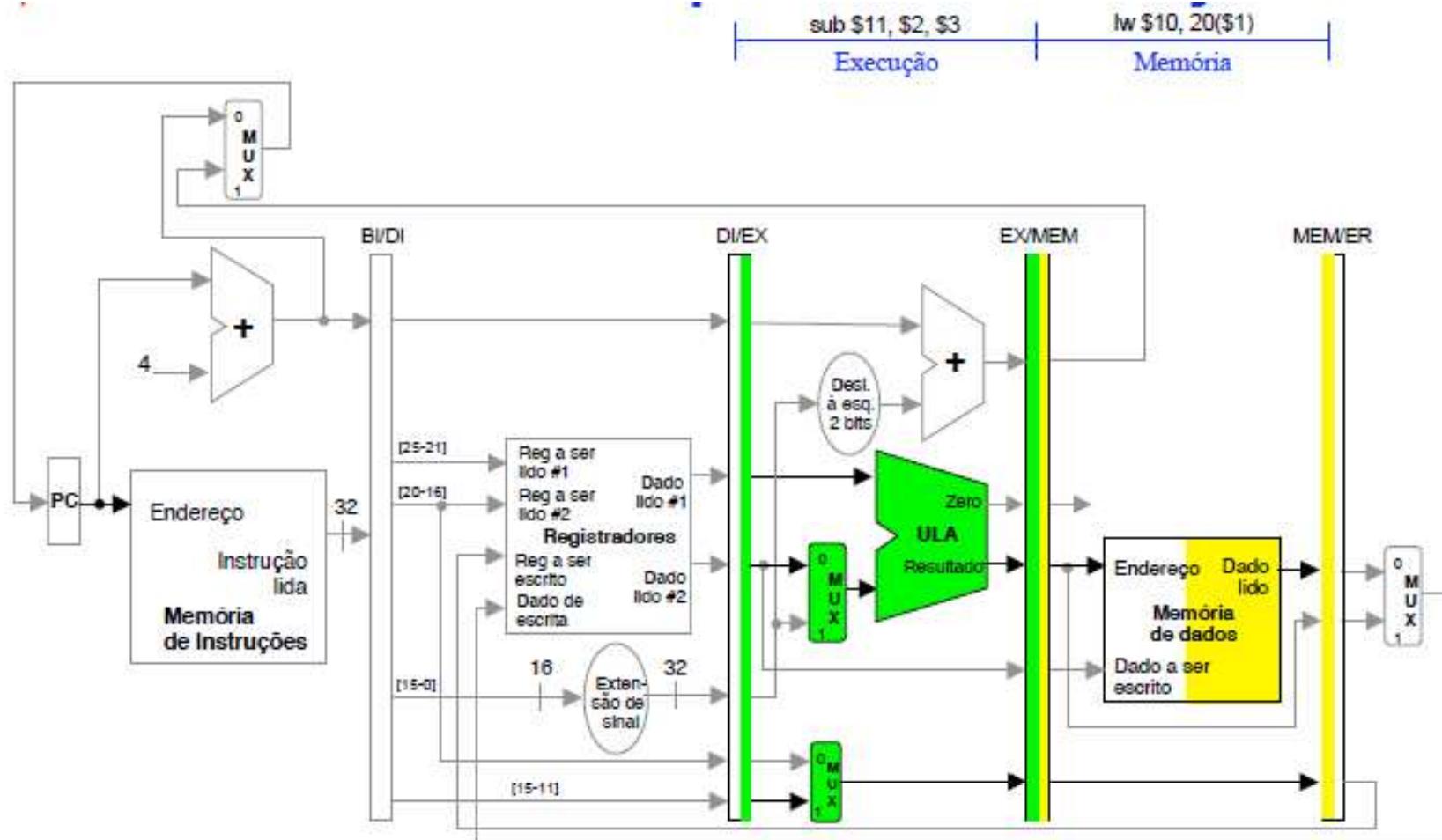
Organizações do MIPS: pipeline

Executando uma sequência de instruções



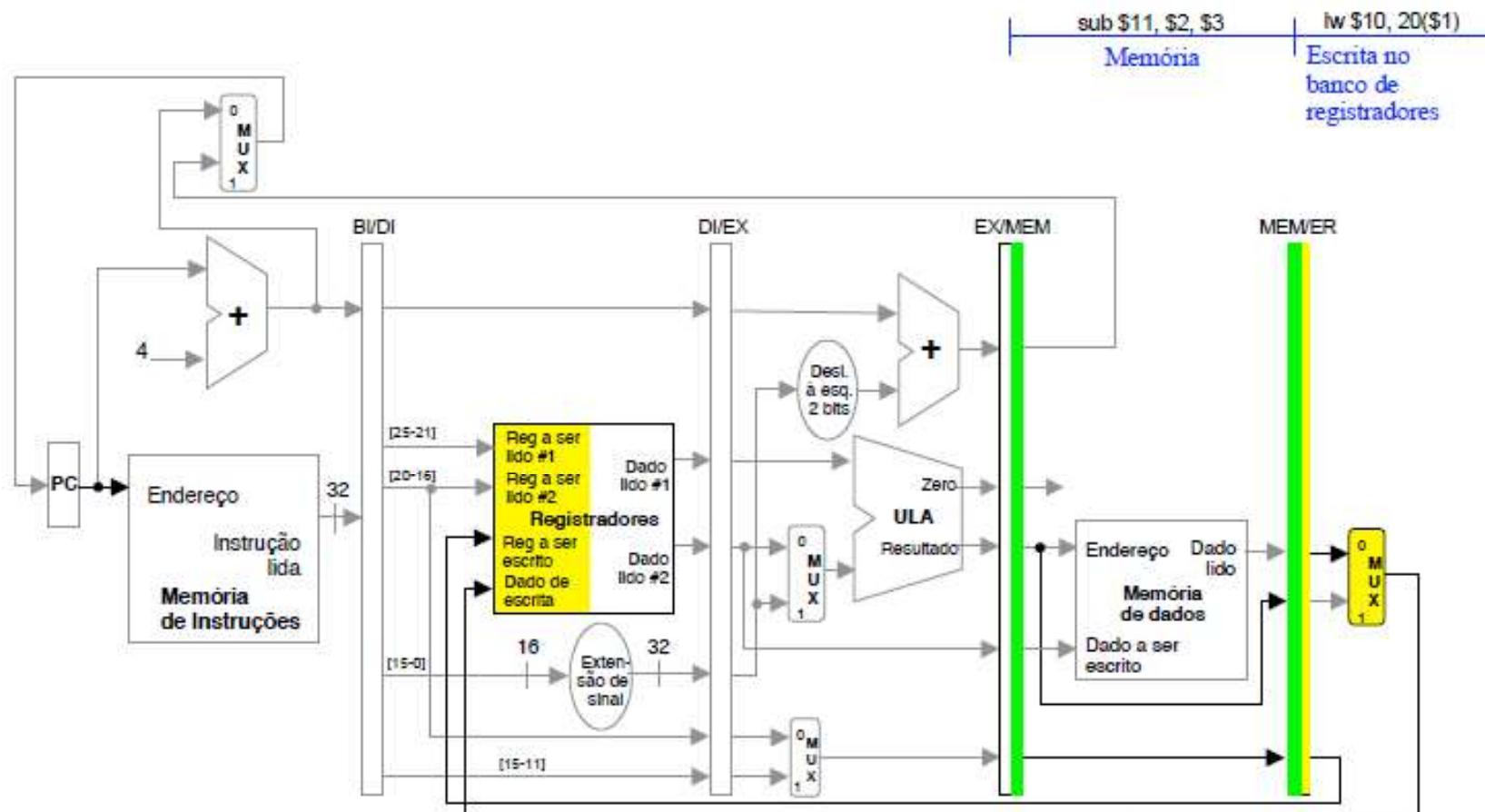
Organizações do MIPS: pipeline

Executando uma sequência de instruções



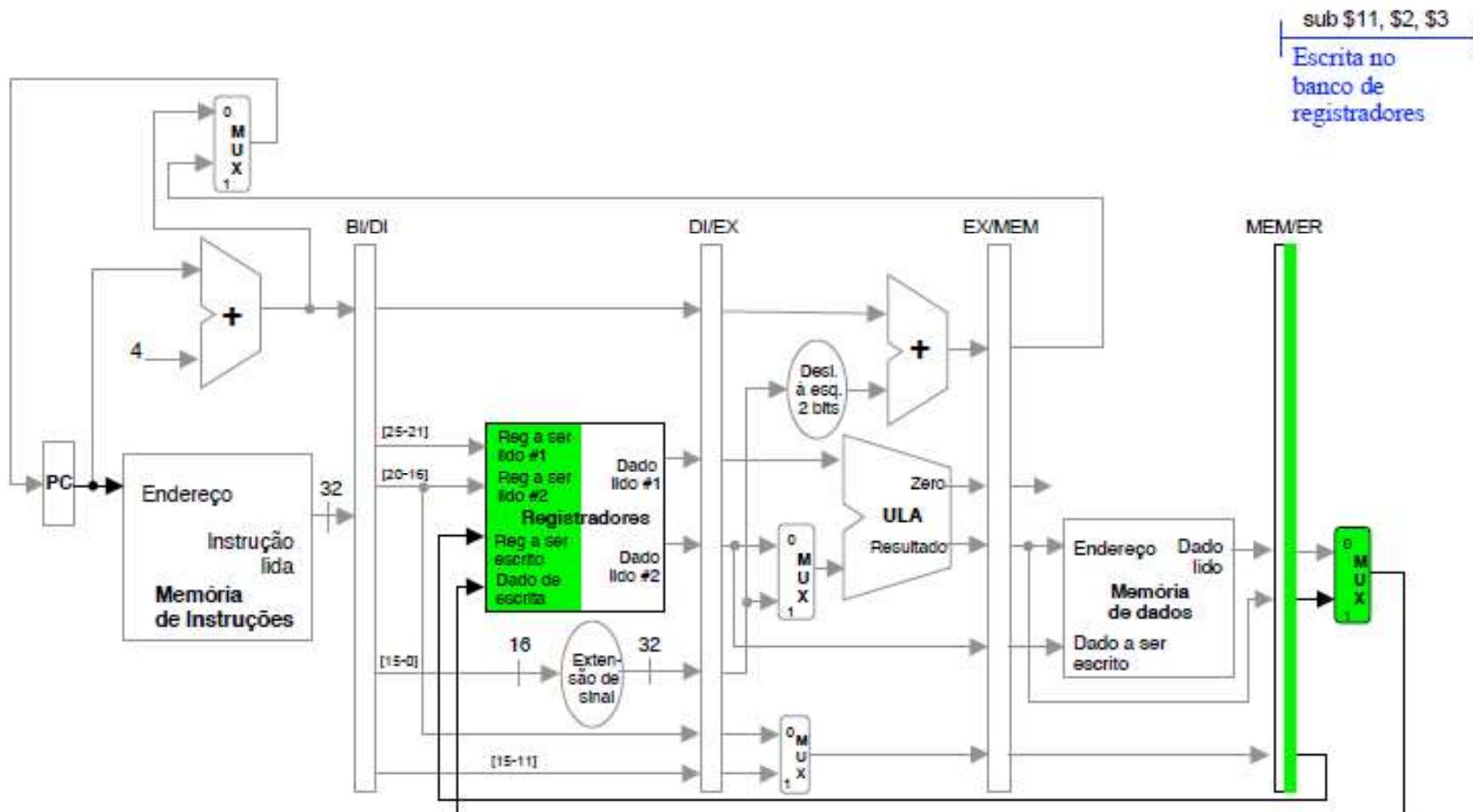
Organizações do MIPS: pipeline

Executando uma sequência de instruções



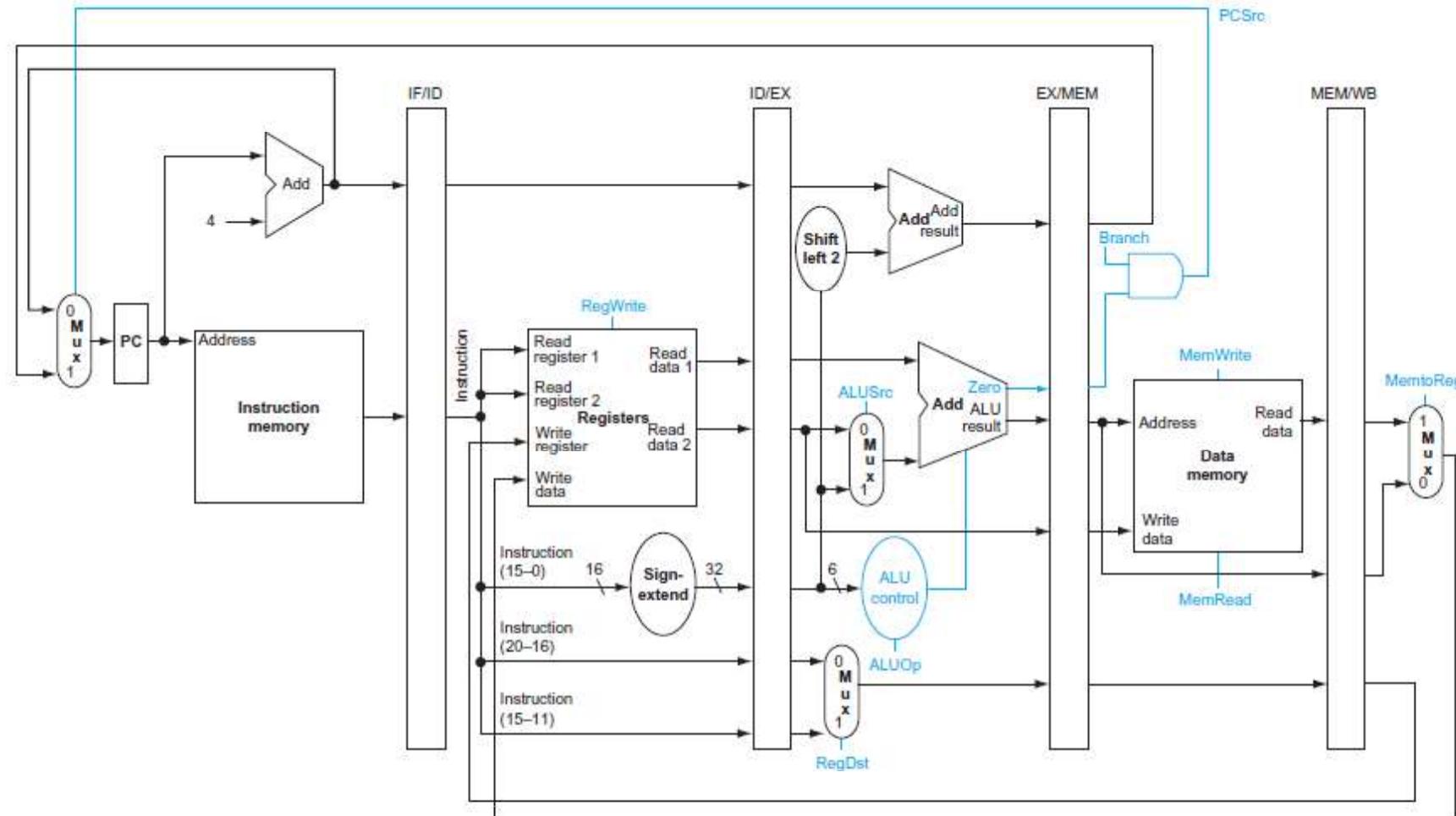
Organizações do MIPS: pipeline

Executando uma sequência de instruções



Organizações do MIPS: pipeline

Bloco Operativo em Pipeline com Sinais de Controle



Projeto do Bloco de Controle

- Iremos aproveitar ao máximo os sinais de controle do MIPS monociclo
- Isto inclui utilizar a mesma lógica de controle para:
 - A ULA
 - O desvio condicional
 - O multiplexador que controla a fonte do dado do registrador destino
 - E demais linhas de controle mostradas na transparência anterior...

Organizações do MIPS: pipeline

Relembrando o Controle da ULA

- Correspondência entre “funct” & “ULAOp” com “controle da ULA”

Instruction opcode	ALUOp	Instruction operation	Function code	Desired ALU action	ALU control input
LW	00	load word	XXXXXX	add	0010
SW	00	store word	XXXXXX	add	0010
Branch equal	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
R-type	10	subtract	100010	subtract	0110
R-type	10	AND	100100	AND	0000
R-type	10	OR	100101	OR	0001
R-type	10	set on less than	101010	set on less than	0111

Conclusões:

- Apenas algumas das 64 combinações possíveis a partir dos 6 bits do campo “funct” são de interesse
- O campo “funct” somente interessa quando ULAOP = 10

Organizações do MIPS: pipeline

Projeto do Bloco de controle

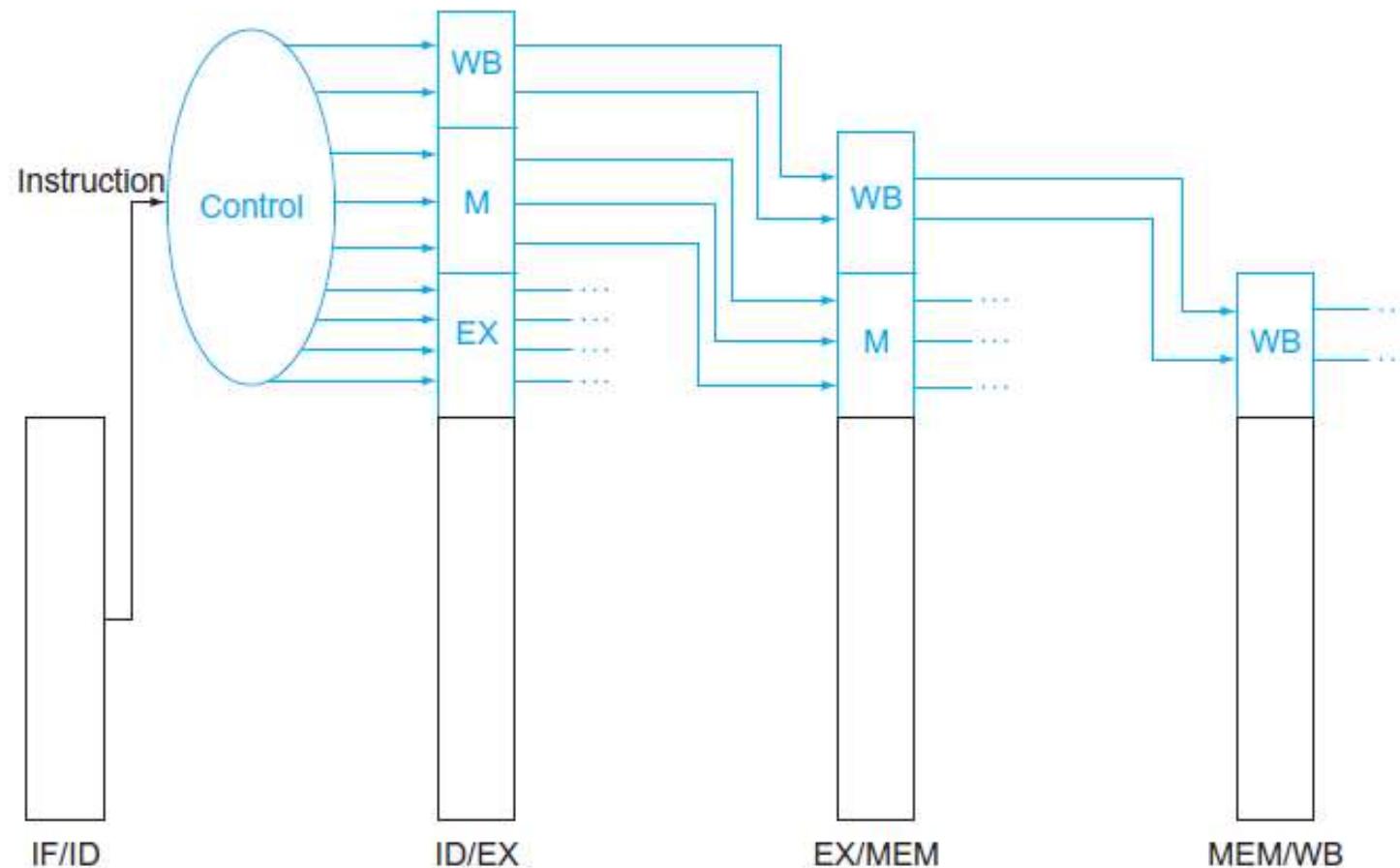
- Reagrupando os Sinais de Controle do MIPS monociclo, a fim de reaproveitá-los...

Instruction	Execution/address calculation stage control lines				Memory access stage control lines			Write-back stage control lines	
	RegDst	ALUOp1	ALUOp0	ALUSrc	Branch	Mem-Read	Mem-Write	Reg-Write	Memto-Reg
R-format	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

- Conforme pode-se perceber, os sinais de controle são essencialmente os mesmos do MIPS monociclo
- A única particularidade é que eles precisam “viajar” pelos estágios juntamente com a instrução

Organizações do MIPS: pipeline

Projeto do Bloco de controle



Organizações do MIPS: pipeline

Projeto do Bloco de controle

