Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Informática e Estatística
Ciências da Computação & Engenharia Eletrônica

Sistemas Digitais

INE 5406

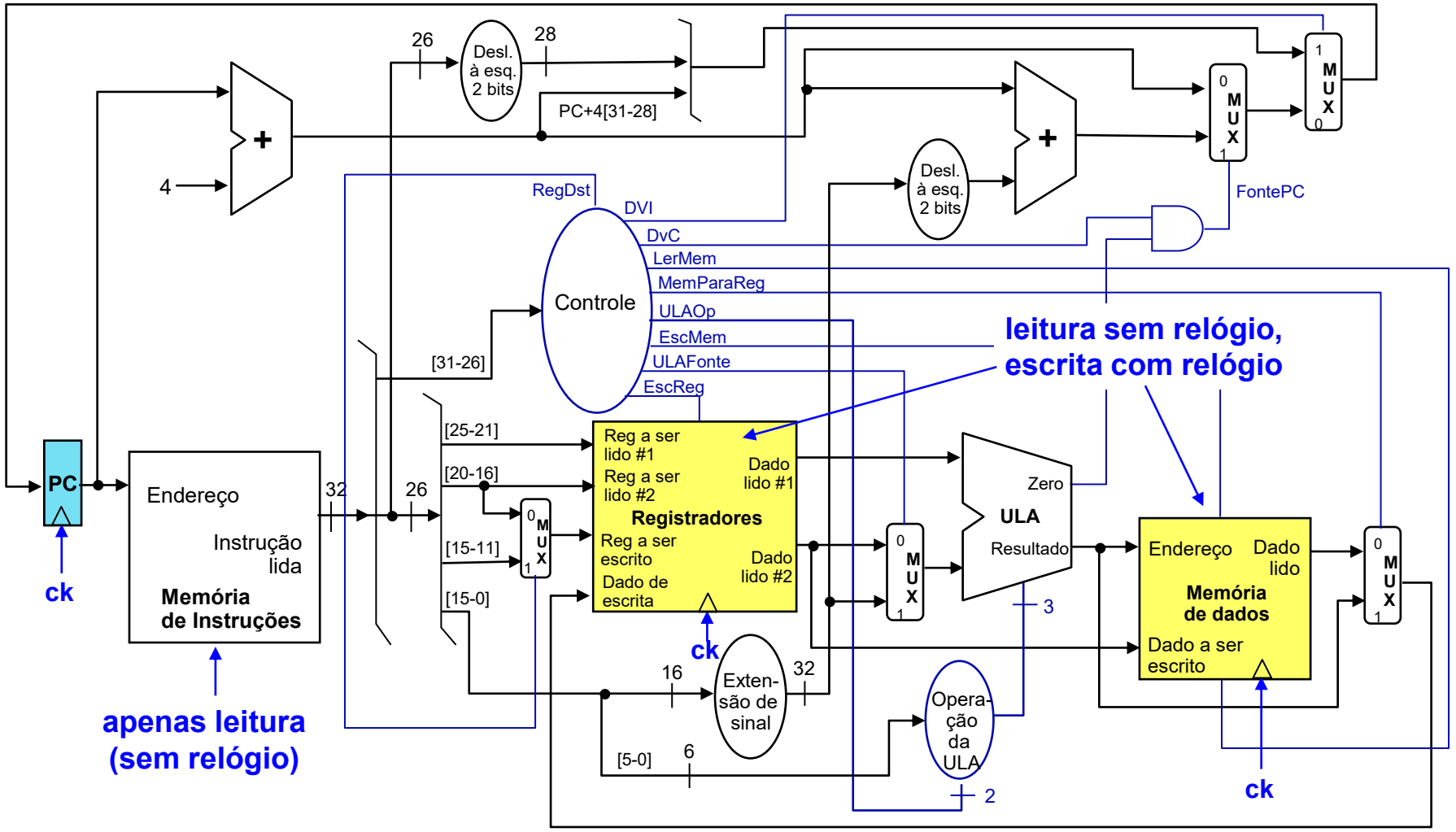
Aula 10-T

4. O Processador MIPS multiciclo: construção do bloco operativo e execução das instruções.

Profs. José Luís Güntzel e Cristina Meinhardt
{j.guntzel, cristina.meinhardt}@ufsc.br

O Processador MIPS Multiciclo

MIPS Monociclo



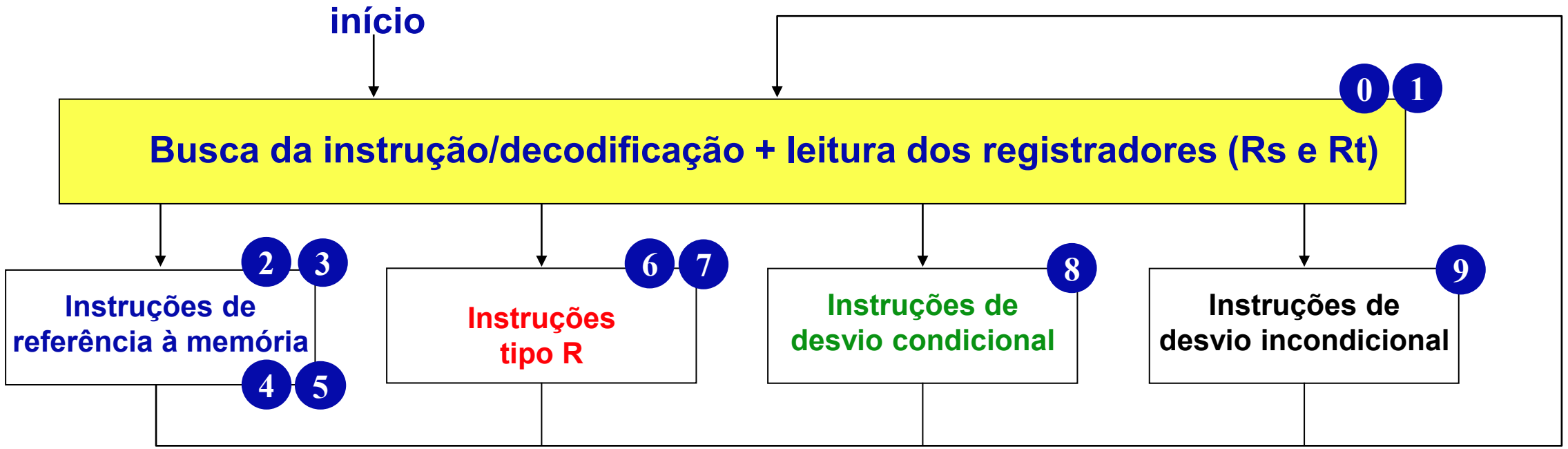
O Processador MIPS Multiciclo

Instruções Consideradas

Instrução	Formato	Linguagem de Montagem	Significado
Adição	R	`add $s1, $s2, $s3`	`$s1 ← $s2 + $s3`
Subtração	R	`sub $s1, $s2, $s3`	`$s1 ← $s2 - $s3`
AND bit a bit	R	`and $s1, $s2, $s3`	`$s1 ← $s2 and $s3`
OR bit a bit	R	`or $s1, $s2, $s3`	`$s1 ← $s2 or $s3`
Load word	I	`lw $s1, desl($s2)`	`$s1 ← Mem[$s2 + desl]`
Store word	I	`sw $s1, desl($s2)`	`Mem[$s2 + desl] ←$s1`
Salto condicional	I	`beq $s1, $s2, desl`	`if ($s1==$s2)` `then PC ← PC+4+(desl<<2)`
Salto incondicional	J	`j L`	`PC ← L onde` `L = ((PC+4)[31-28]) || (constante << 2)`

O Processador MIPS Multiciclo

Derivando uma Implementação Multiciclo



Cada caixa pode representar um ou mais passos e portanto, pode executar em um ou mais ciclos de relógio (dependendo da instrução).

O Processador MIPS Multiciclo

Derivando uma Implementação Multiciclo

Princípio: Cada etapa (abaixo) terá exatamente um ciclo de relógio para executar (e assim, o período do relógio será reduzido)

instrução	Etapa 1	Etapa 2	Etapa3	Etapa 4	Etapa 5
Tipo R	Busca da instrução	Lê registrador(es) & decodifica instrução	ULA	Escreve registrador	
lw	Busca da instrução	Lê registrador(es) & decodifica instrução	ULA	Lê memória	Escreve registrador
sw	Busca da instrução	Lê registrador(es) & decodifica instrução	ULA	Escreve na memória	
beq	Busca da instrução	Lê registrador(es) & decodifica instrução	ULA		
jump	Busca da instrução	decodifica instrução *			

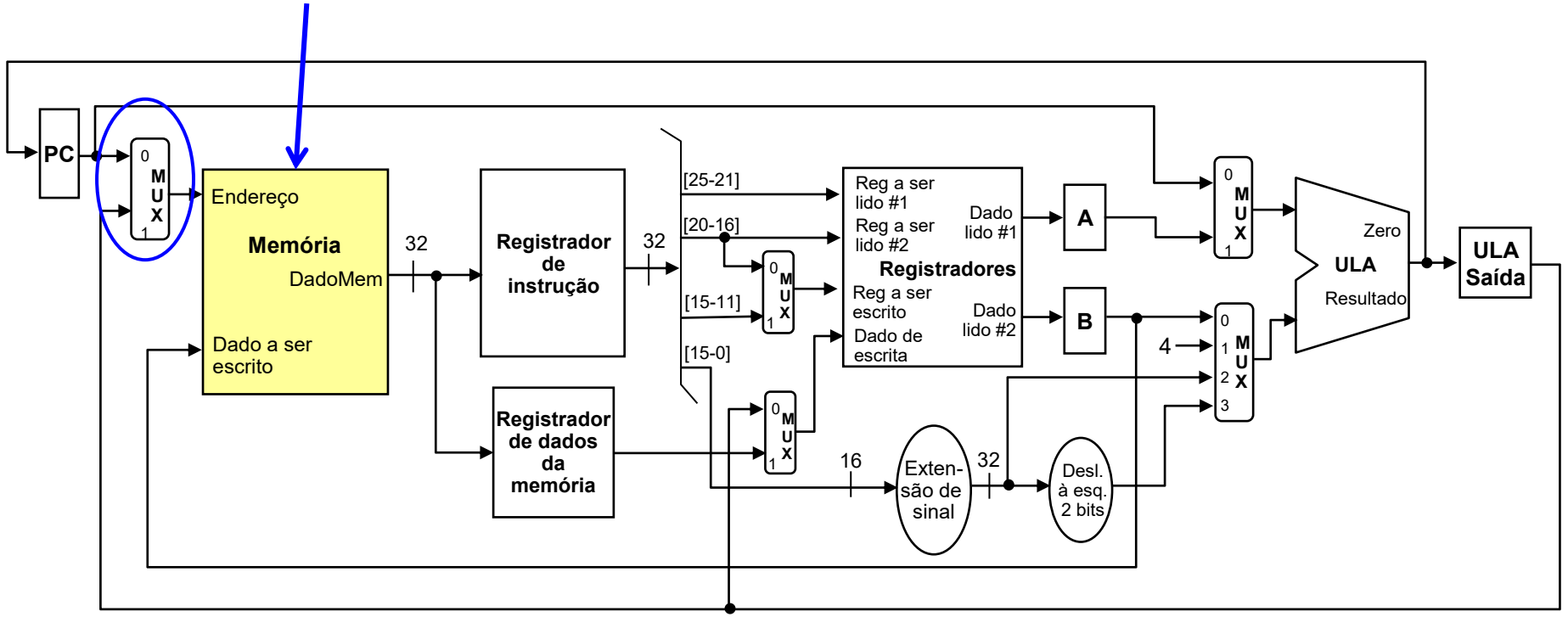
Implementação:

- Inserir registrador(es) entre os elementos que executam as etapas
- Possibilidade de reaproveitar recursos (principalmente a ULA)

O Processador MIPS Multiciclo

Bloco Operativo Multiciclo: Características

Uma única memória para armazenar dados e instruções

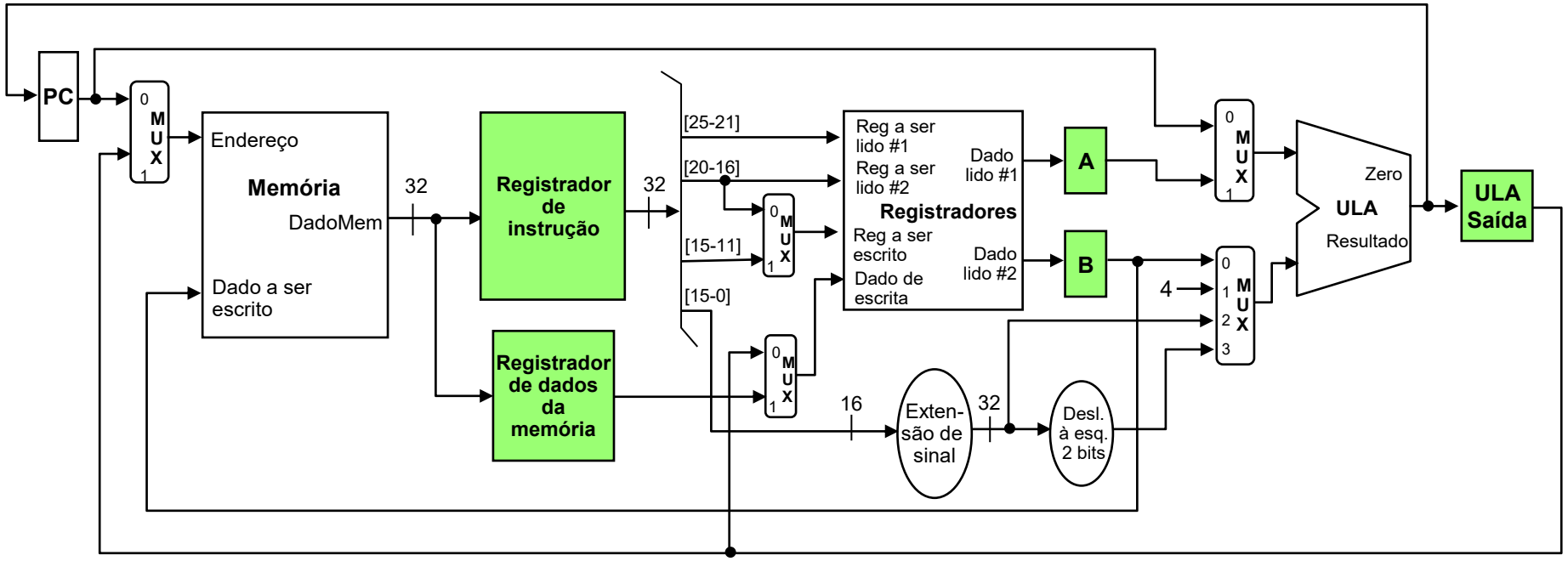


Consequência:

- **endereço de leitura vem do PC; endereço de escrita vem de ULASaída. Logo, usar mux 2:1**

O Processador MIPS Multiciclo

Bloco Operativo Multiciclo — Novos Registradores Não-Visíveis ao Programador

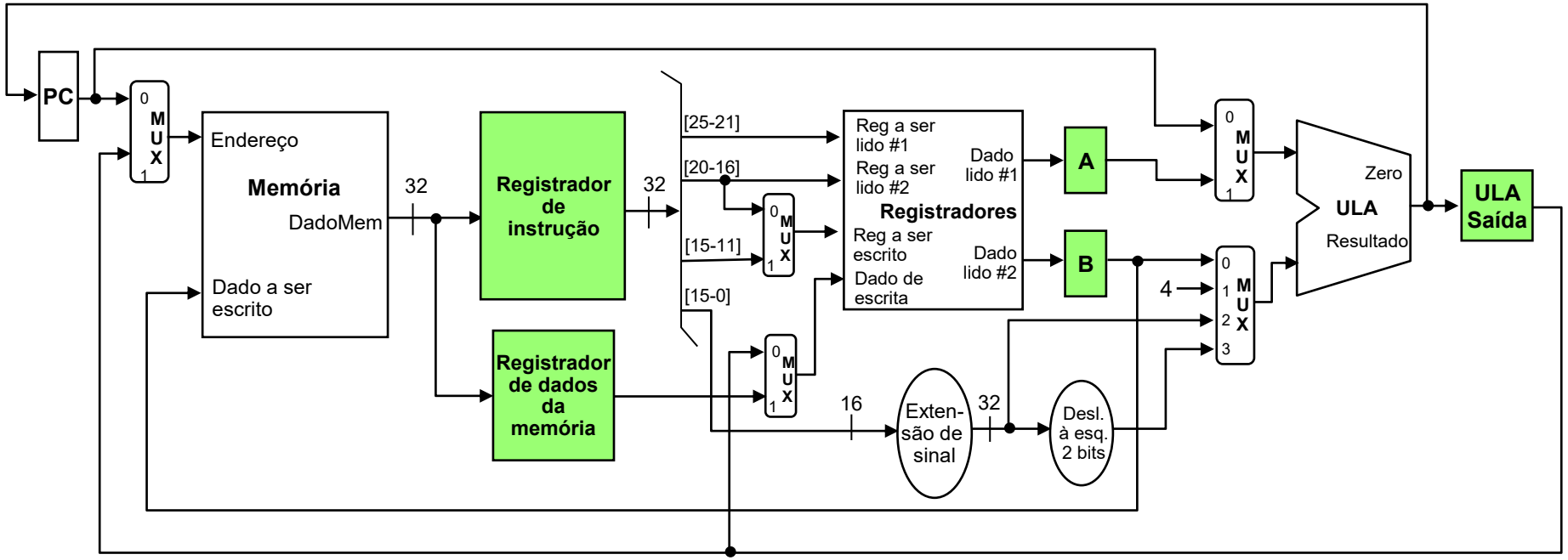


- **Registrador de Instrução (IR ou RI)**
- **Registrador de Dados da Memória (MDR ou RDM)**
- **Registradores A e B,** para guardar os dados lidos do banco de registradores
- **Registrador ULASaída,** para guardar o resultado de operação feita c/ a ULA

O Processador MIPS Multiciclo

Bloco Operativo Multiciclo

Novos Registradores Não-Visíveis ao Programador

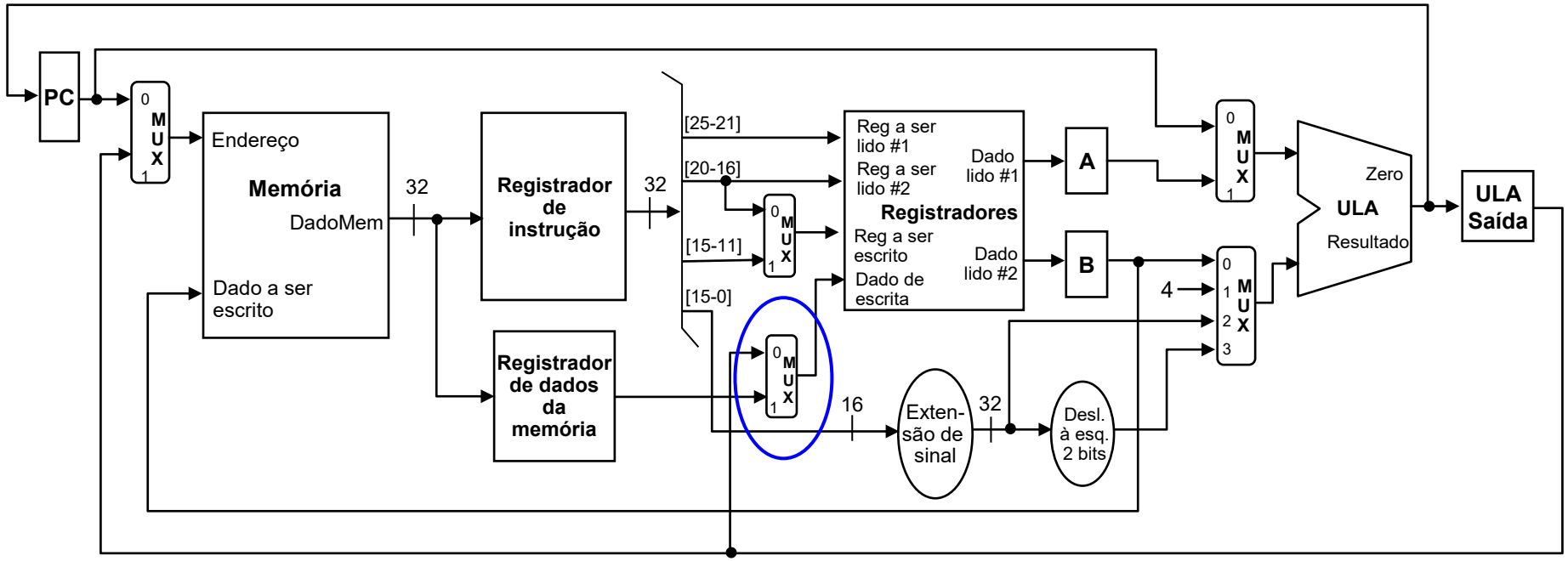


Todos os registradores novos (**exceto o RI**) devem armazenar dados somente entre duas bordas de relógio consecutivas e portanto, não necessitam sinal de carga (controle de escrita), apenas do sinal de relógio (omitido nos diagramas)

O Processador MIPS Multiciclo

Bloco Operativo Multiciclo

Dado a ser Escrito em Registrador pode vir da ULA (via ULASaída) ou da Memória (via RDM)



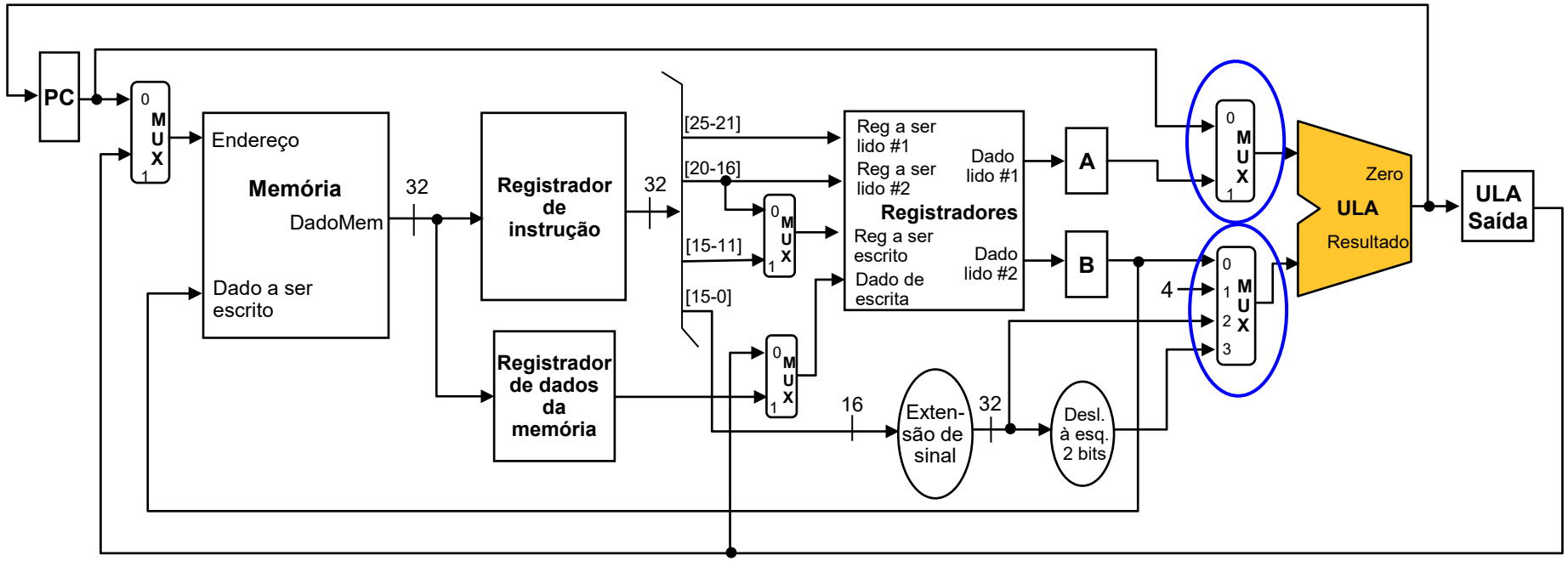
Consequência:

- **Reposicionamento do mux 2:1 na entrada de dado a ser escrito (do banco de registradores)**

O Processador MIPS Multiciclo

Bloco Operativo Multiciclo

ULA usada também para calcular PC+4 e endereço de desvio

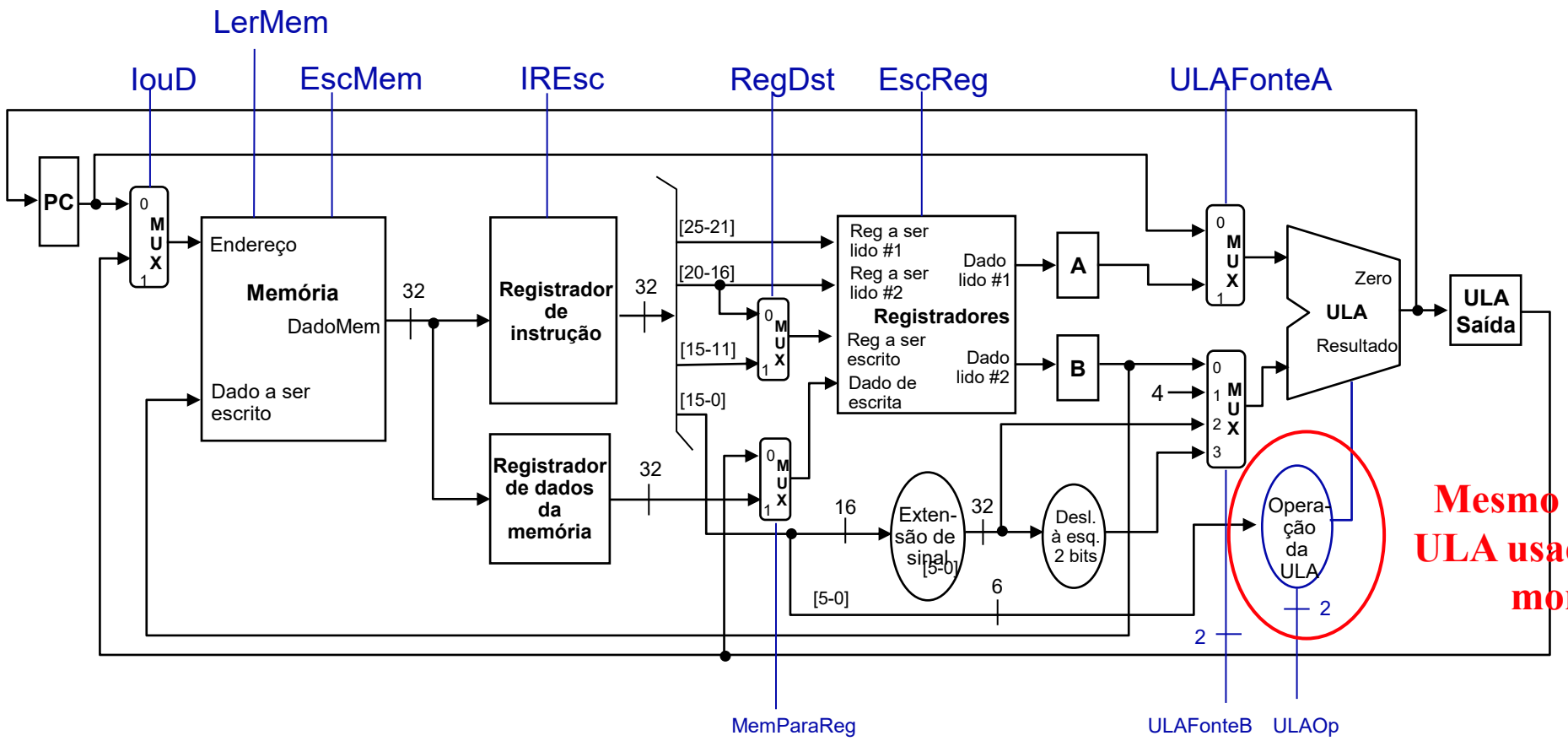


Consequência:

- **Necessidade de mux 2:1 na entrada superior**
- **Entrada inferior passa a ter mux 4:1 (ao invés de mux 2:1)**

O Processador MIPS Multiciclo

Bloco Operativo Multiciclo + Sinais de Controle



Mesmo controle da ULA usado na versão monociclo

O Processador MIPS Multiciclo

MIPS Multiciclo

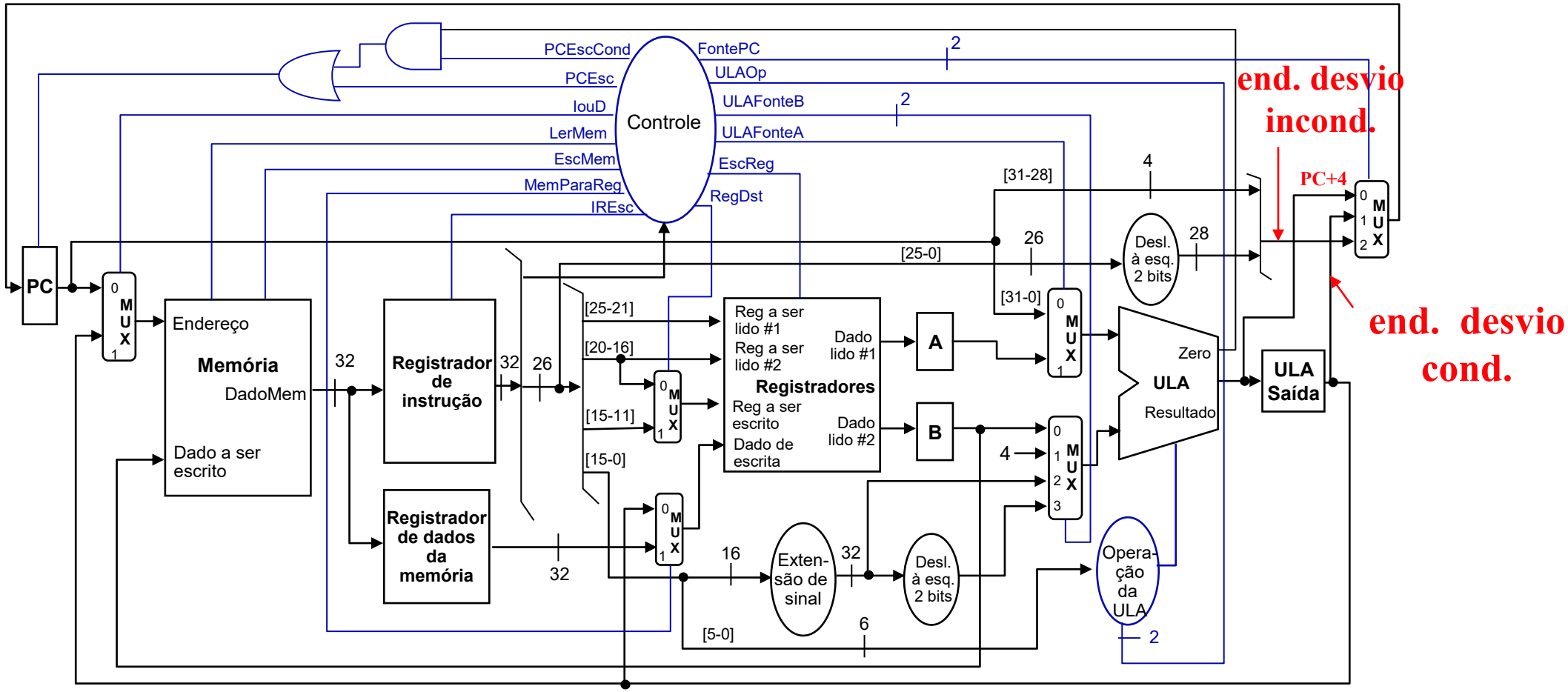
Acréscimos de Recursos para Suportar jump e branch on equal

Existem 3 possíveis fontes para o PC:

1. O resultado de PC+4, disponível na saída da ULA: este valor sempre será armazenado no PC
2. O conteúdo de ULASaída: este registrador armazena o endereço-alvo do desvio condicional, após este ter sido calculado pela ULA (beq)
3. Os 26 bits menos significativos do IR, deslocados à esquerda e concatenados com os 4 bits mais significativos do PC incrementado (jump)

O Processador MIPS Multiciclo

MIPS Multiciclo (incluso Bloco de Controle)



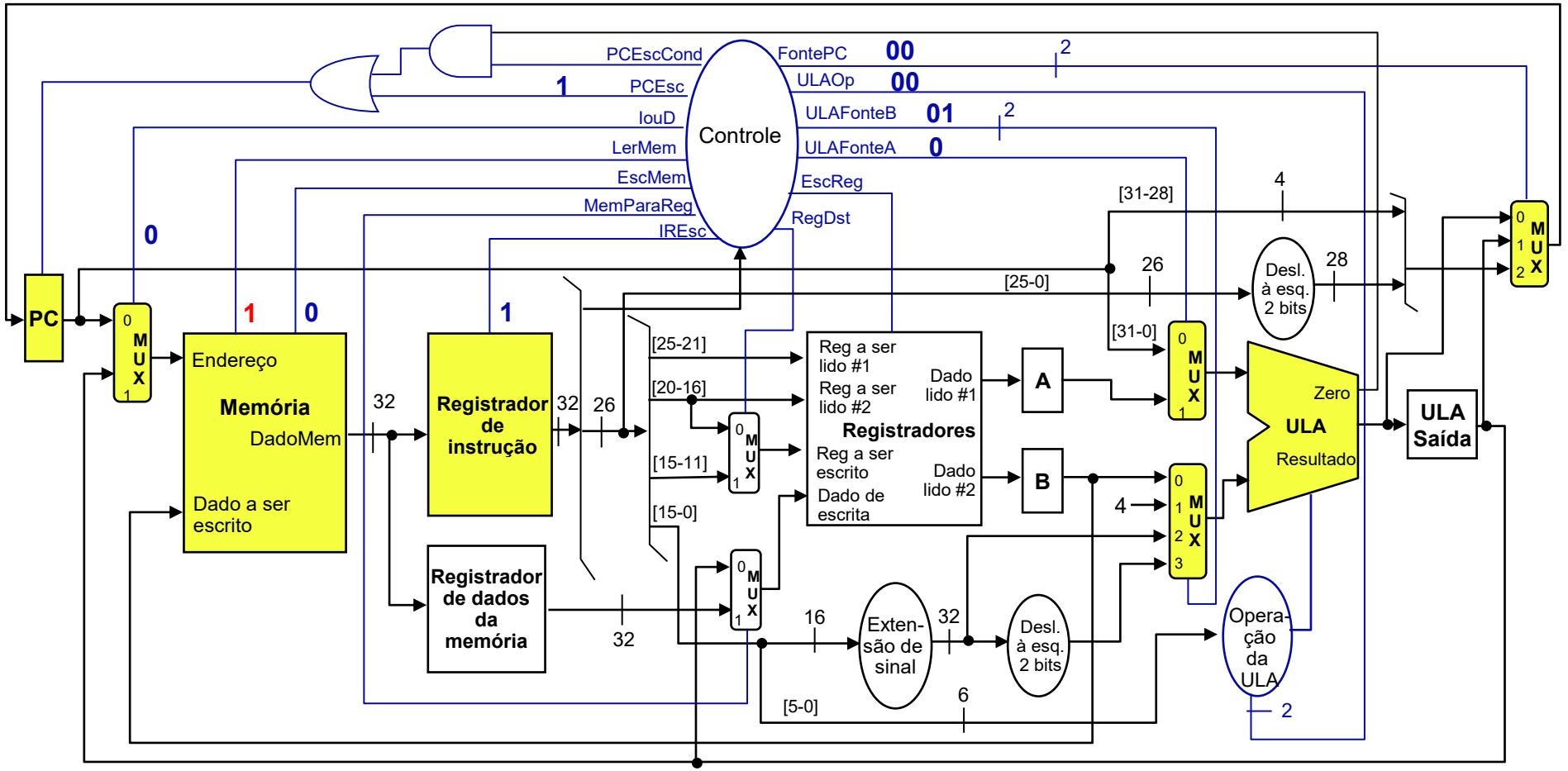
O Processador MIPS Multiciclo

Passos Necessários para Qualquer Instrução no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

O Processador MIPS Multiciclo

Busca da Instrução (e PC ← PC+4)



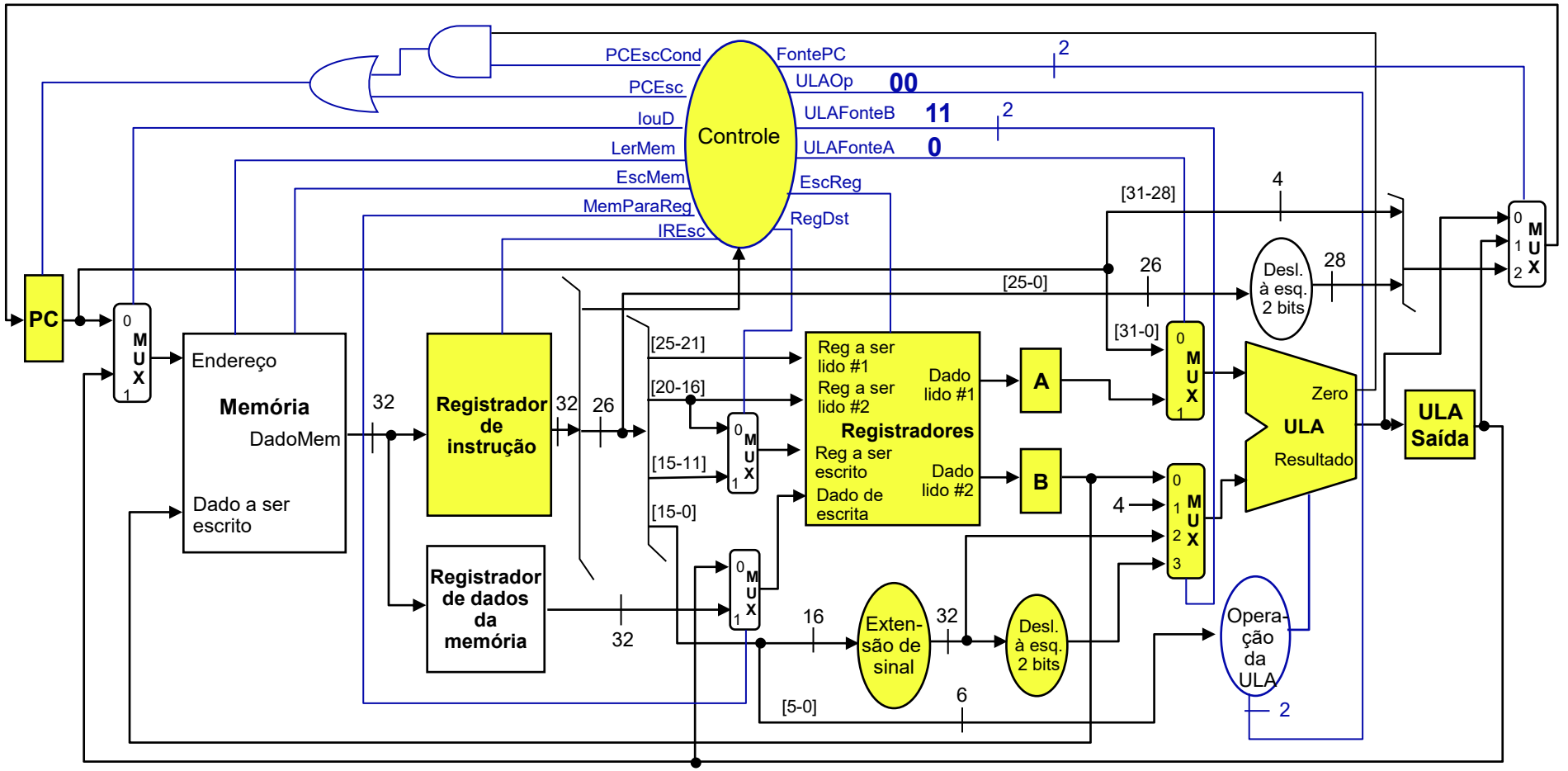
O Processador MIPS Multiciclo

Passos Necessários para Qualquer Instrução no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

O Processador MIPS Multiciclo

Decodificação da instrução & Leit. Rs e Rt (cálculo do end. p/ beq)



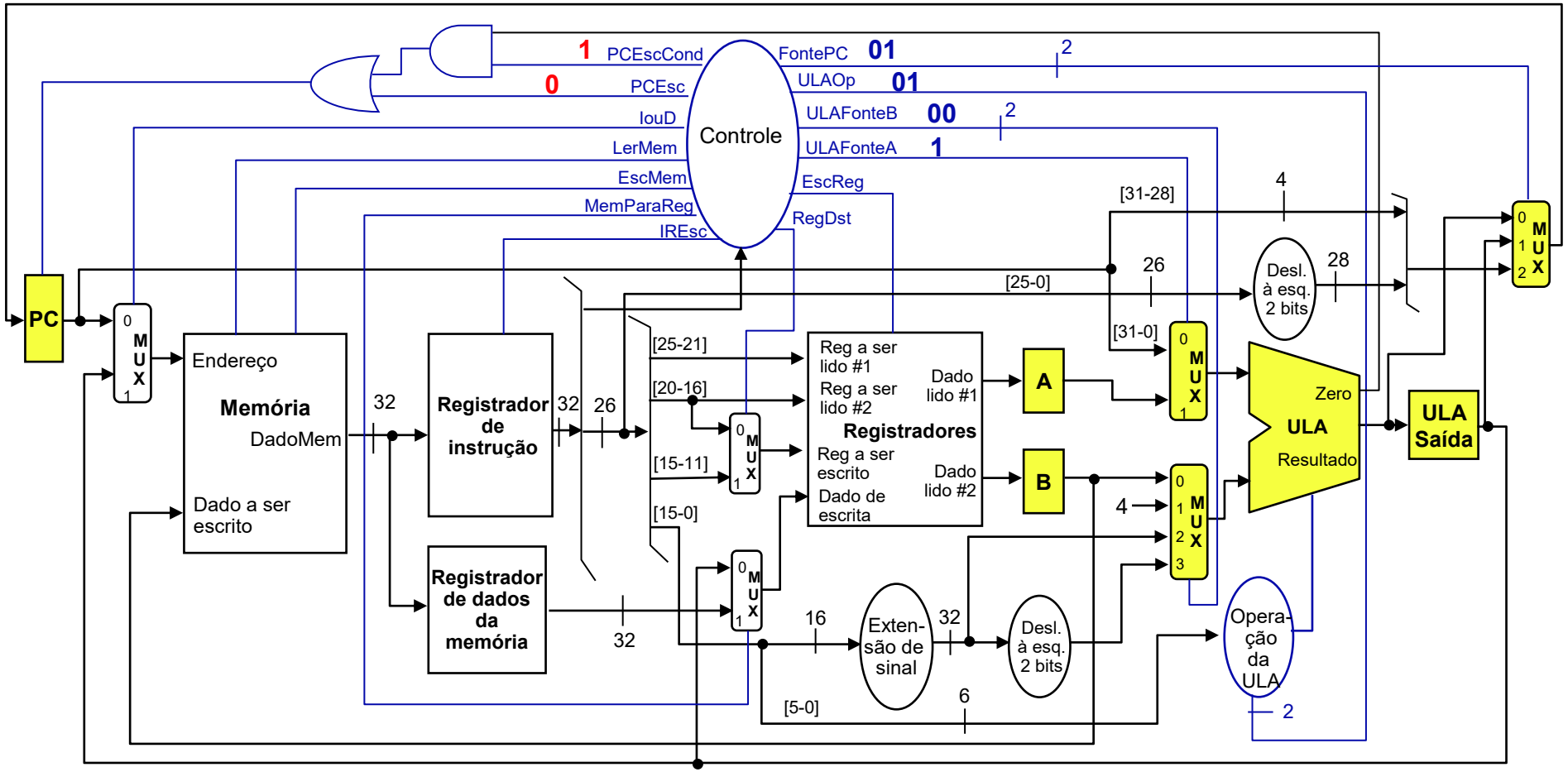
O Processador MIPS Multiciclo

Passos Necessários para Instrução beq no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

O Processador MIPS Multiciclo

Execução da Instrução beq: testa se A == B



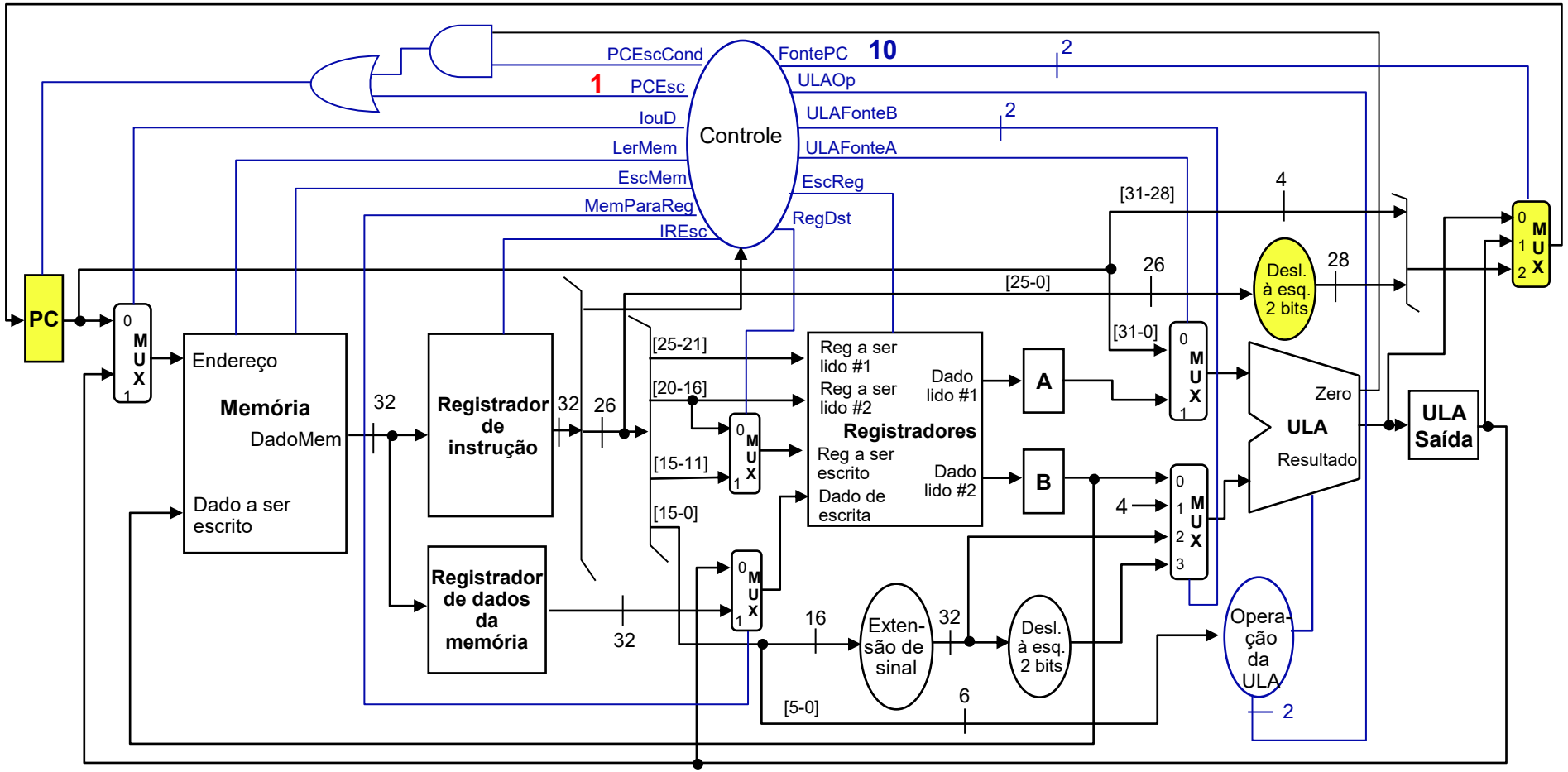
O Processador MIPS Multiciclo

Passos Necessários para Instrução jump no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

O Processador MIPS Multiciclo

Execução da Instrução jump: somente seleção de caminho



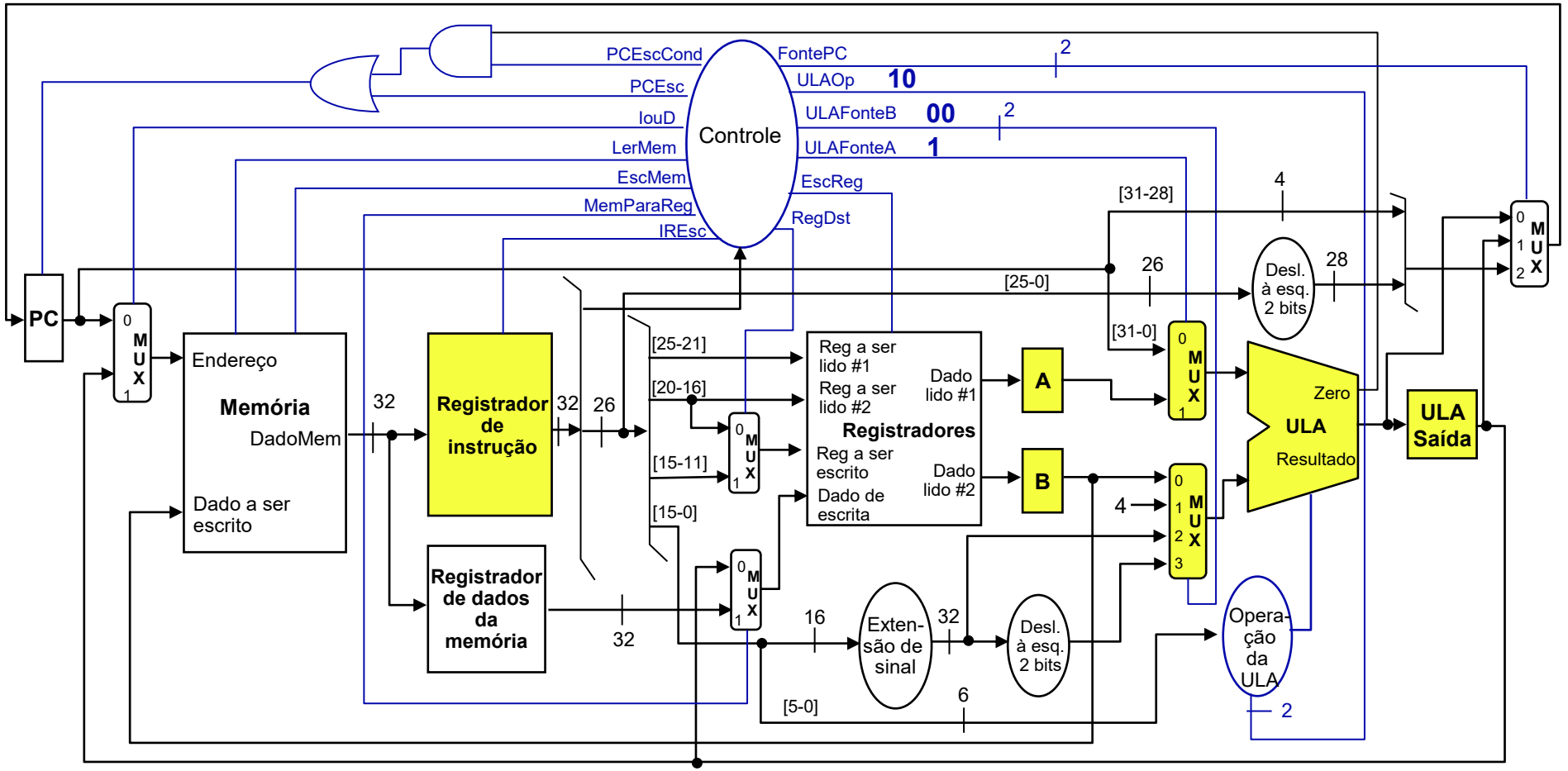
O Processador MIPS Multiciclo

Passos Necessários para Instruções Tipo R no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

O Processador MIPS Multiciclo

Execução de Instruções Tipo R: execução na ULA



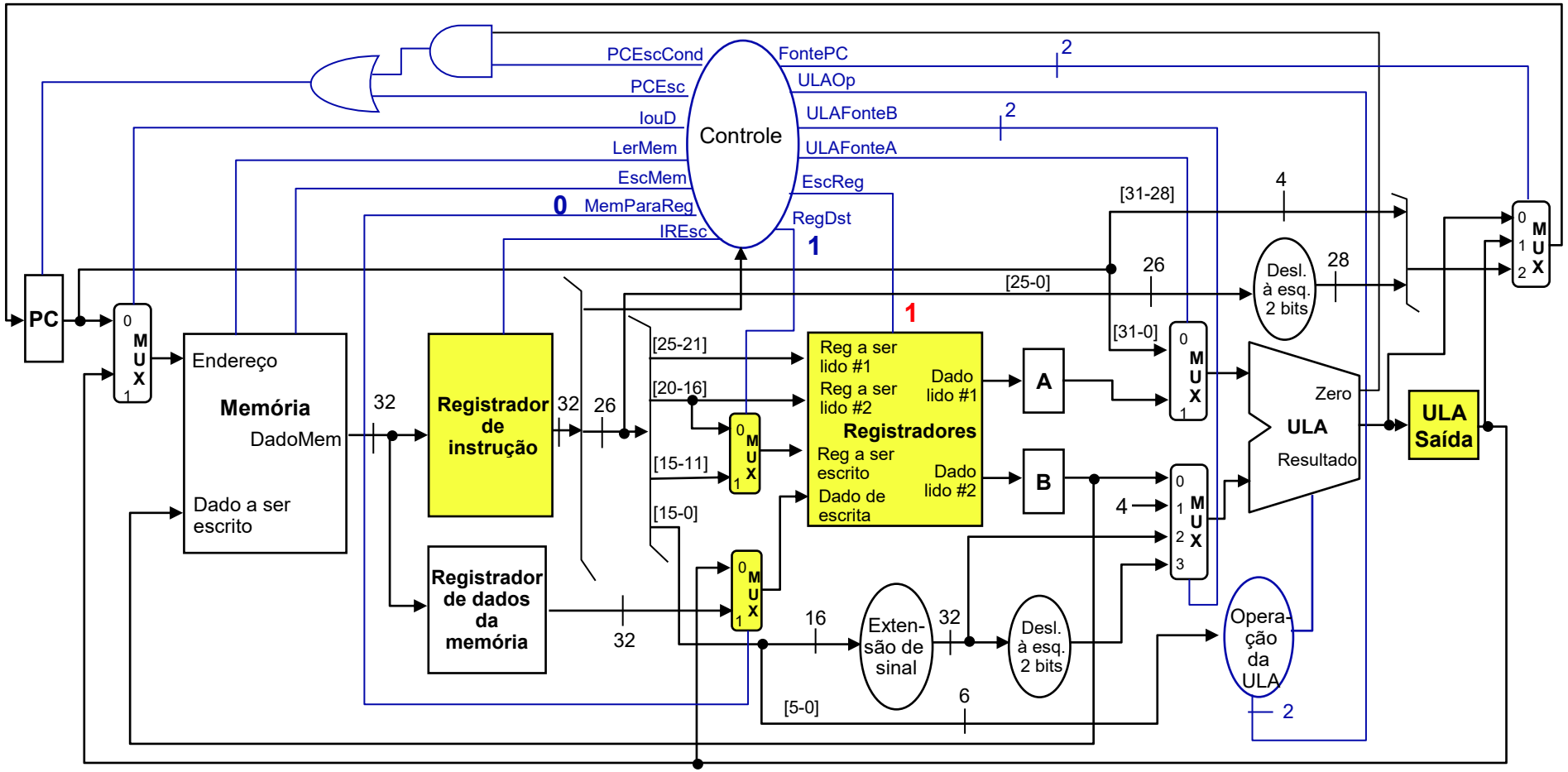
O Processador MIPS Multiciclo

Passos Necessários para Instruções Tipo R no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

O Processador MIPS Multiciclo

Execução de Instruções Tipo R: escreve em registrador



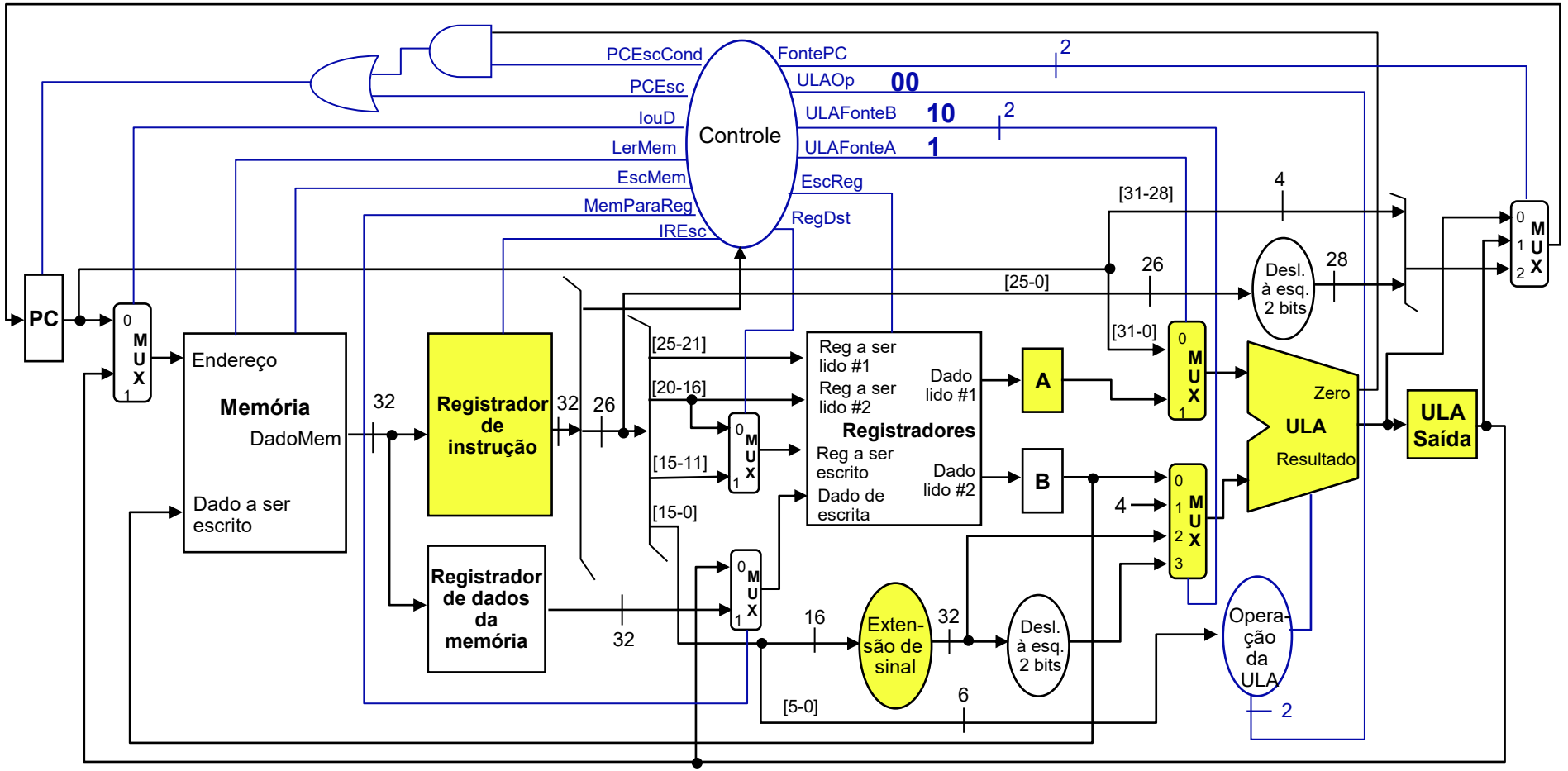
O Processador MIPS Multiciclo

Passos Necessários para Instruções lw e sw no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

O Processador MIPS Multiciclo

Execução de Instruções: execução lw/sw (calc. endereço desvio)



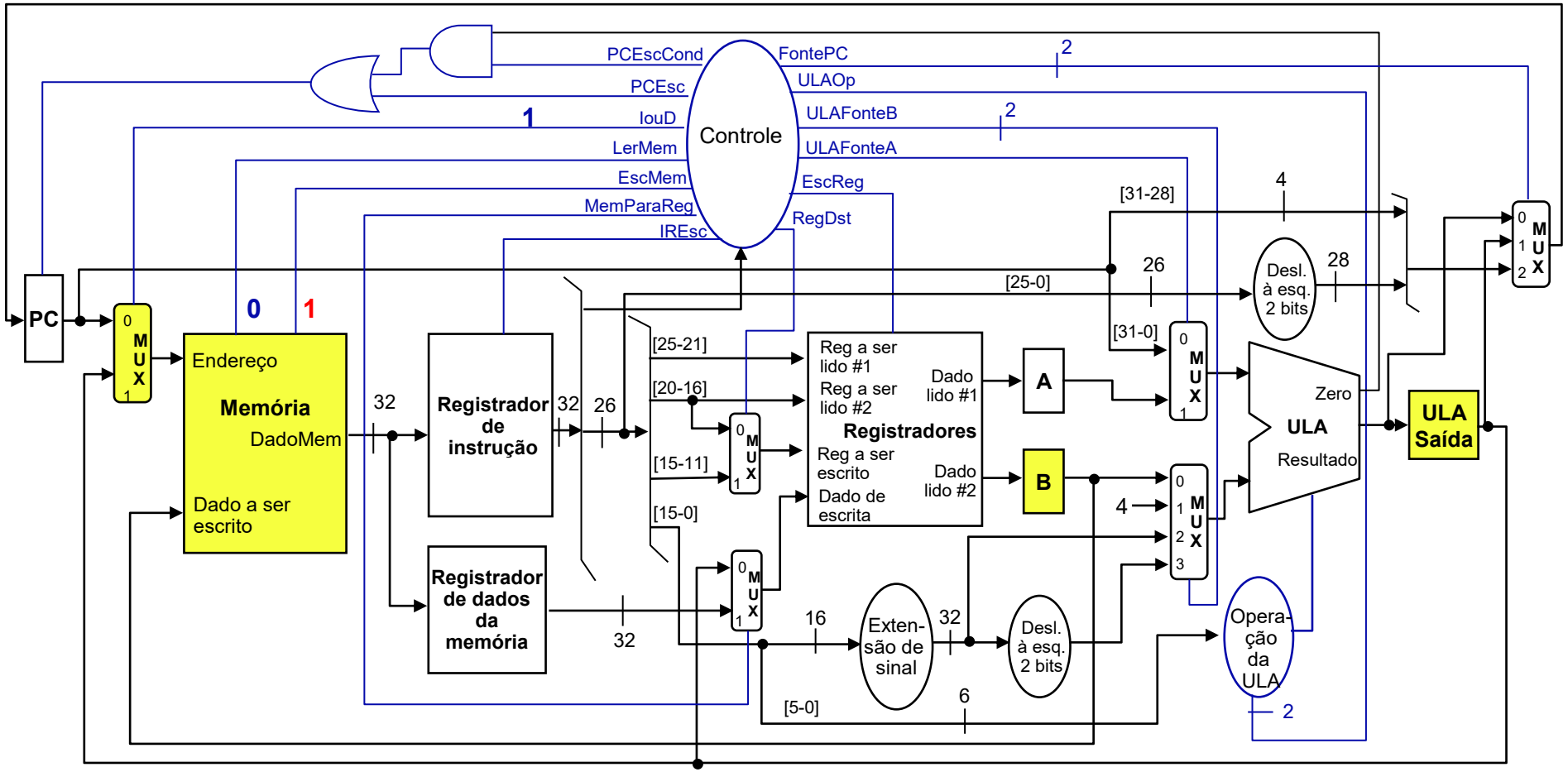
O Processador MIPS Multiciclo

Passos Necessários para Instrução sw no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

O Processador MIPS Multiciclo

Término da Instrução sw: escreve na memória



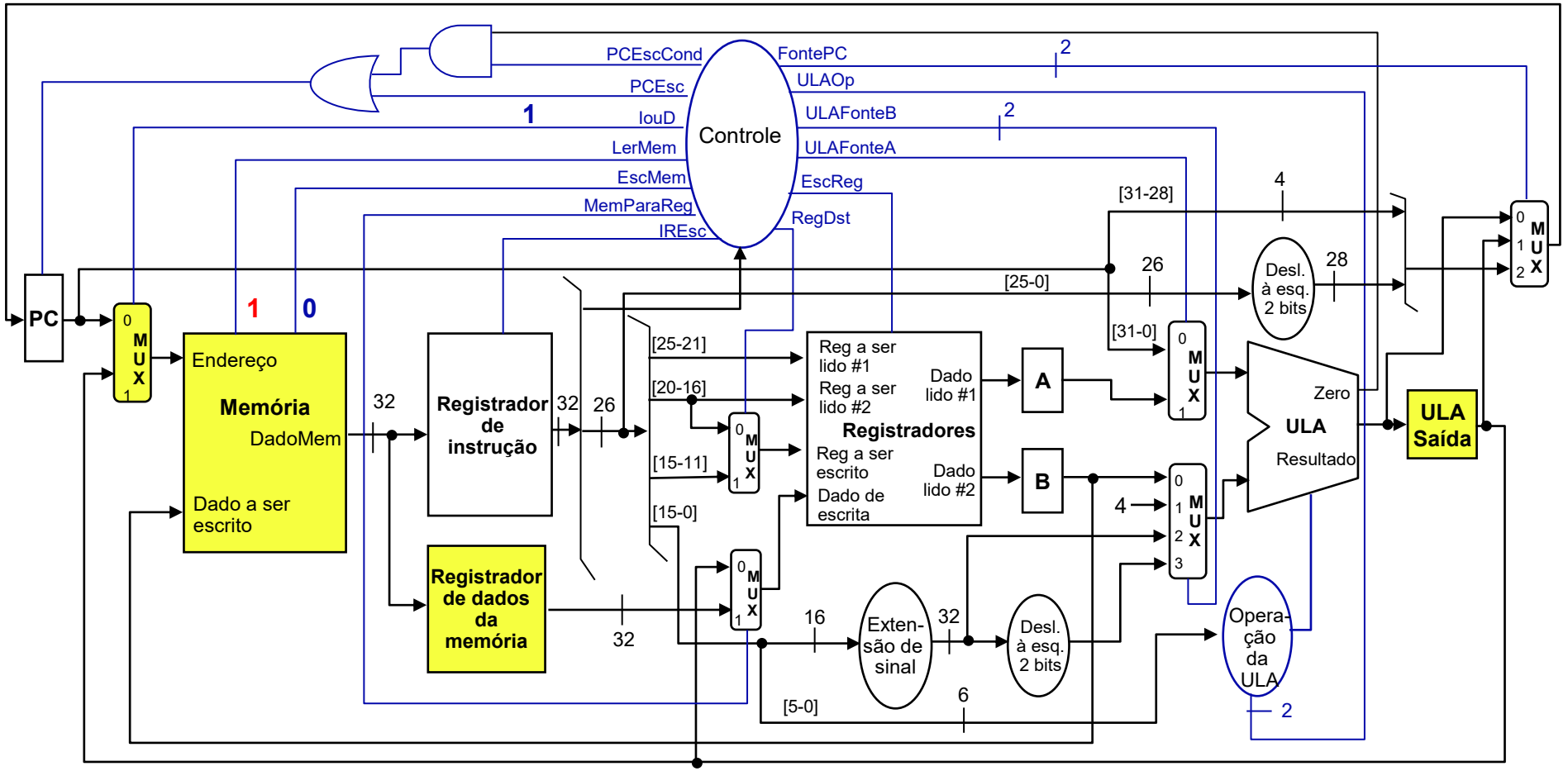
O Processador MIPS Multiciclo

Passos Necessários para Instrução lw no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

O Processador MIPS Multiciclo

Acesso à Memória na Instrução lw: lê dado da memória



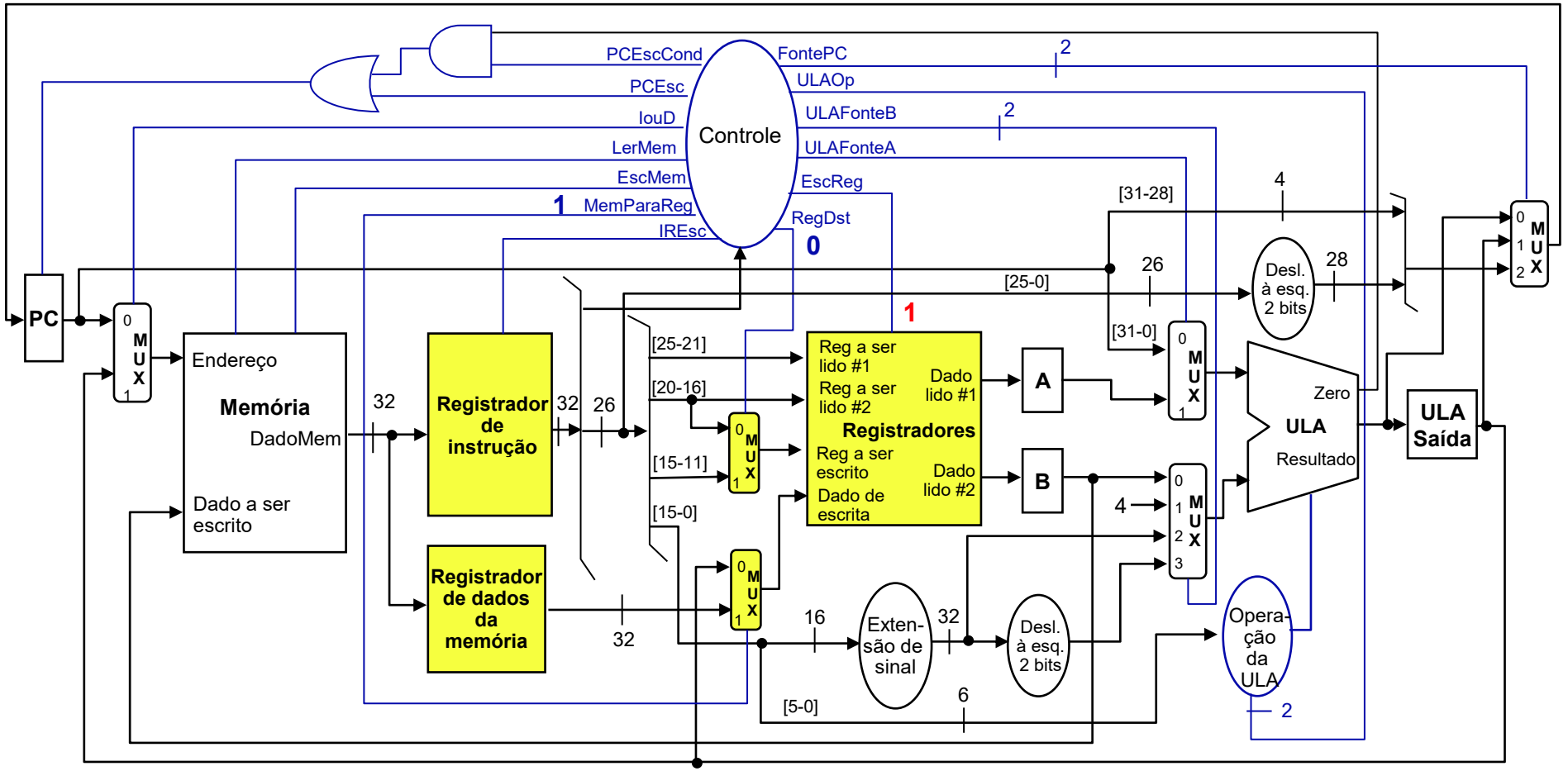
O Processador MIPS Multiciclo

Passos Necessários para Instrução lw no MIPS Multiciclo

Nome do passo	Instrução tipo R	Instrução lw	Instrução sw	Instrução beq	Instrução j
Busca da instrução	RI = Mem[PC] PC = PC + 4				
Decodificação da instrução & leitura dos registradores Rs e Rt & cálculo do endereço de desvio (cond.)	A = Reg [RI[25-21]] B = Reg [RI[20-16]] ULASaída = PC + (extensão de sinal(RI[15-0]) <<2)				
Execução, cálculo do endereço de acesso à memória, término de uma instrução branch/jump	ULASaída = A op B	ULASaída = A + extensão de sinal (RI[15-0])		Se (A == B) então PC = ULASaída	PC = PC[31-28] || (RI[25-0] <<2)
Término de uma instrução store word ou de tipo R	Reg [RI[15-11]] = ULASaída	RDM = Mem [ULASaída]	Mem [ULASaída] = B		
Término de uma instrução load word		Reg[RI[20-16]] = RDM			
Número de passos	**4**	**5**	**4**	**3**	**3**

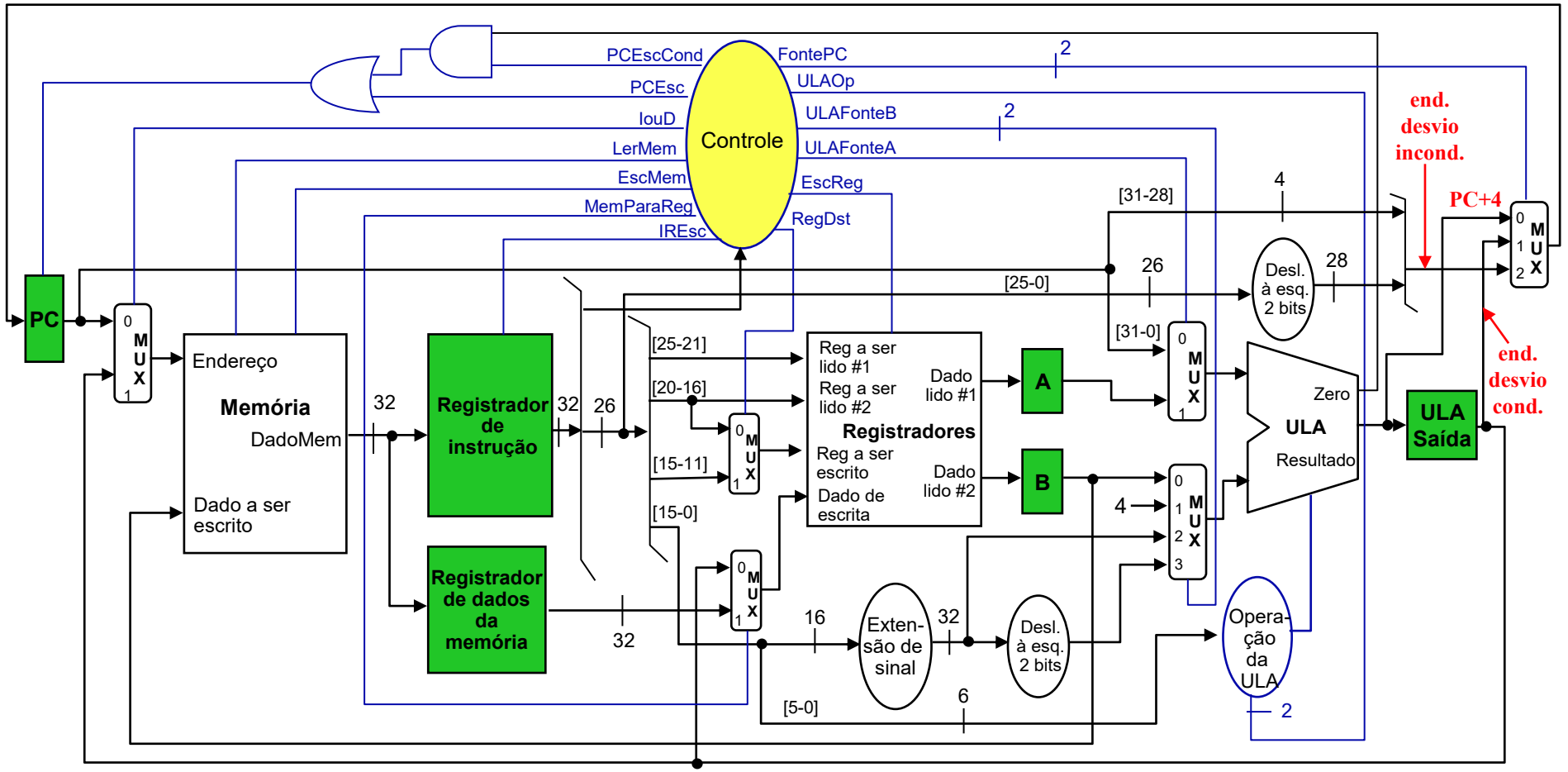
O Processador MIPS Multiciclo

Término da Instrução lw: escreve em registrador



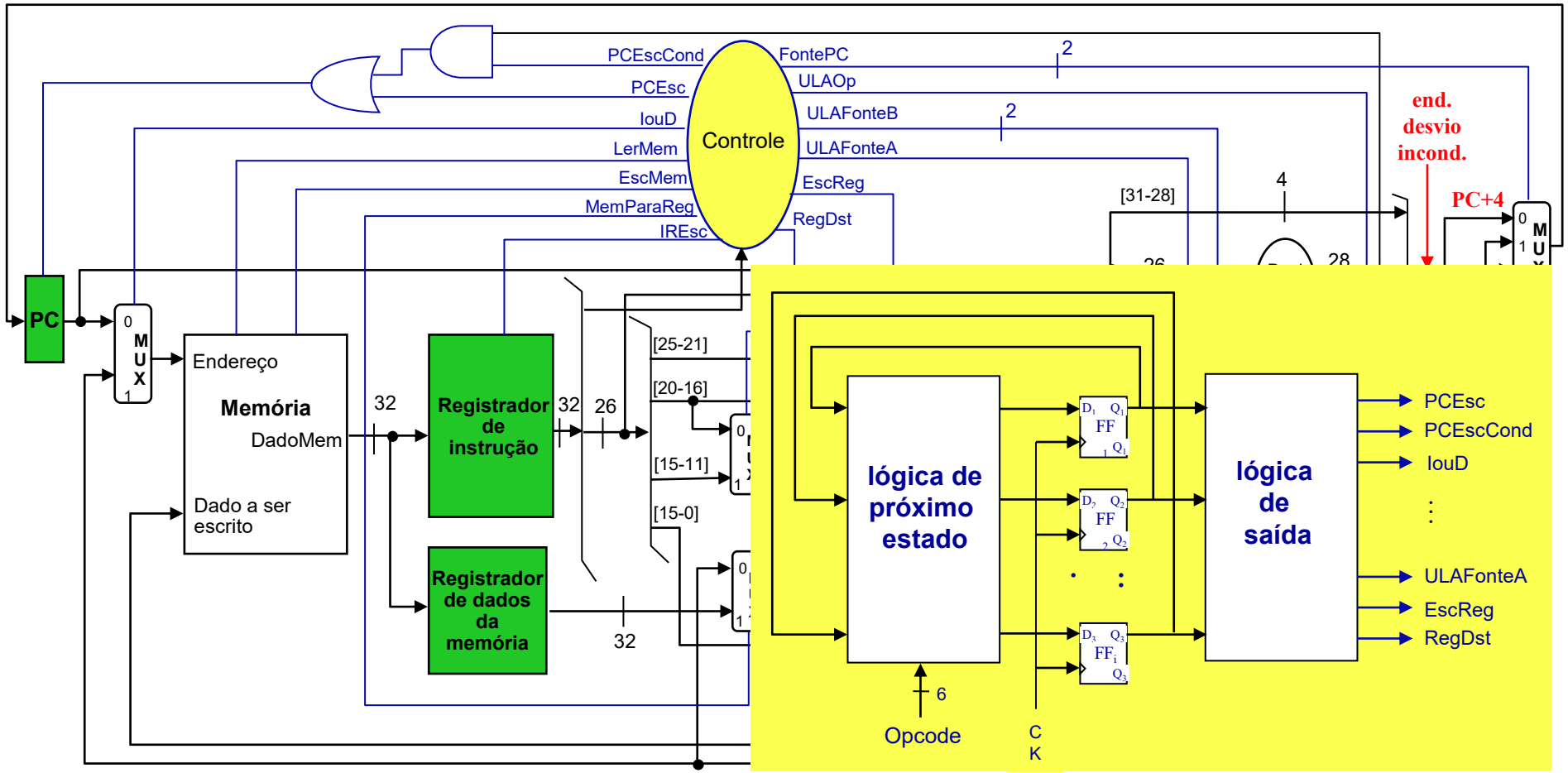
O Processador MIPS Multiciclo

Caminhos Críticos (para Estimar o Relógio)



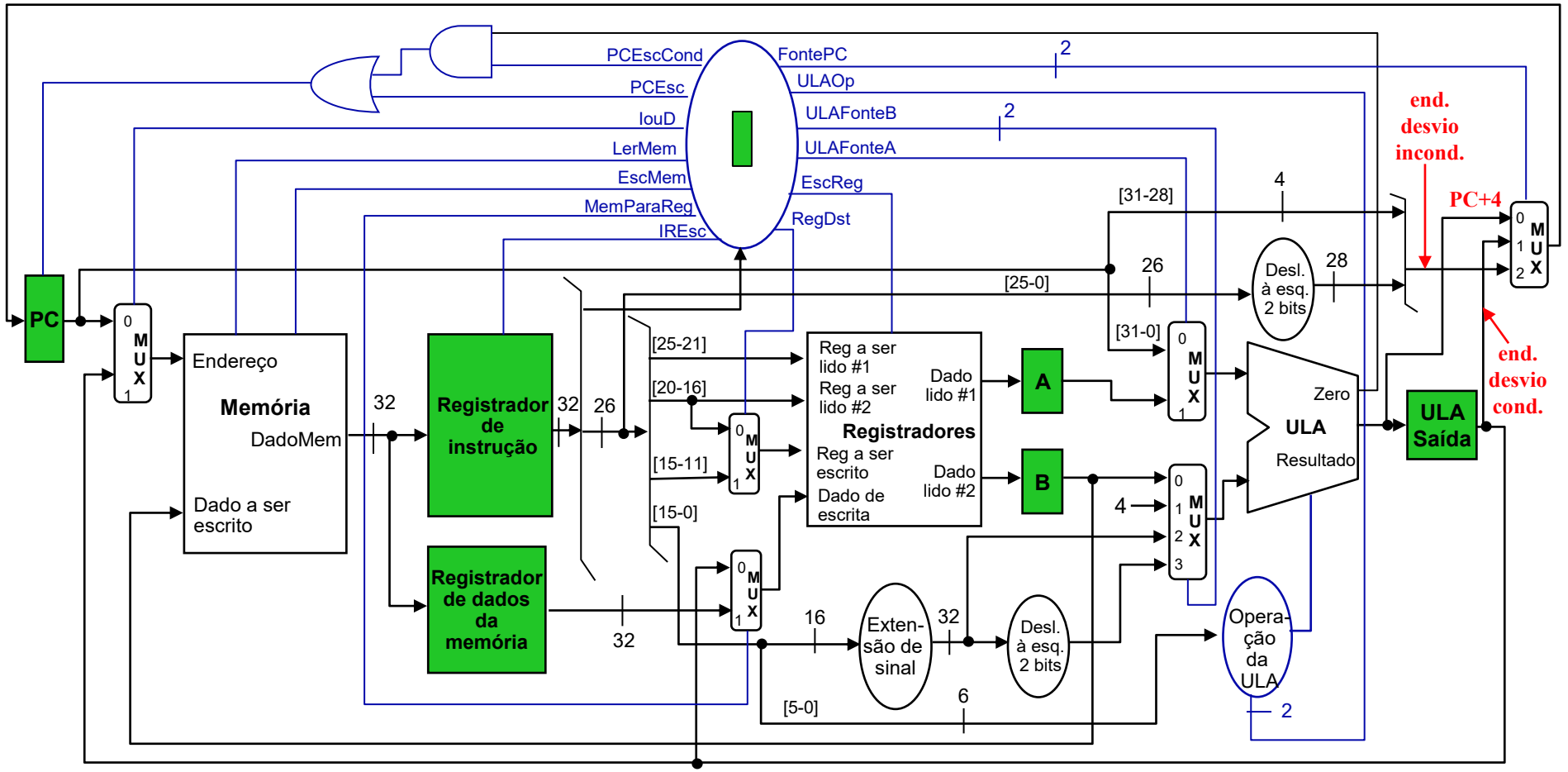
O Processador MIPS Multiciclo

Caminhos Críticos (para Estimar o Relógio)



O Processador MIPS Multiciclo

Caminhos Críticos (para Estimar o Relógio)

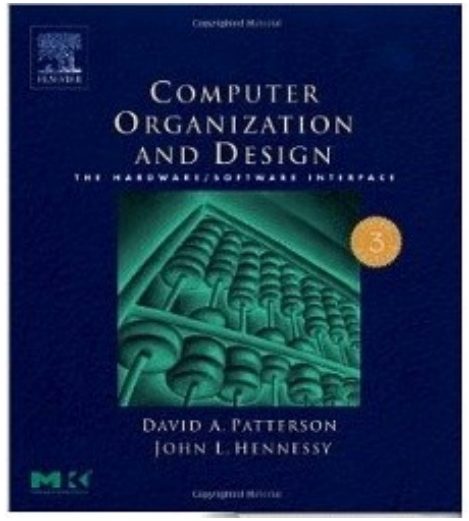


O Processador MIPS Multiciclo

Leitura da Semana

Livro:

PATTERSON, David A.; HENNESSY, John L. “Computer Organization and Design: the hardware/software Interface”, 3rd edition, Morgan Kaufmann Publishers, San Francisco, California, USA, 2007.



Se usar a 2ª Edição: Seção 5.4.
Se usar a 3ª Edição: Seção 5.5.