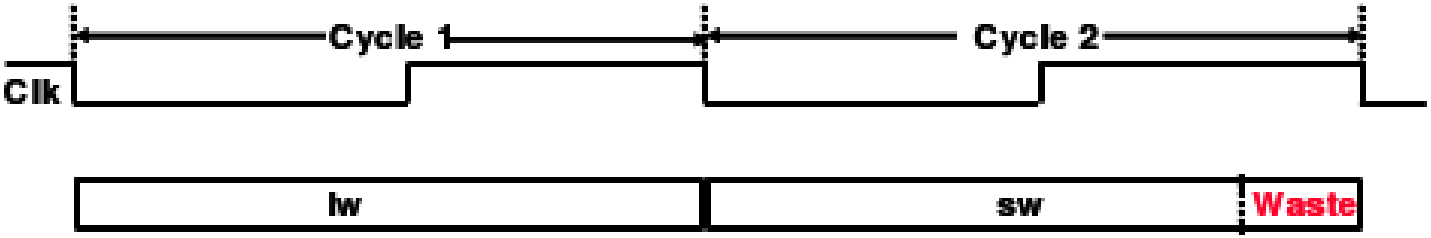
[Aula 09] Caminho de dados e controle 3 – MIPS Multiciclo

Prof. João F. Mari
joaof.mari@ufv.br

MIPS Multiciclo

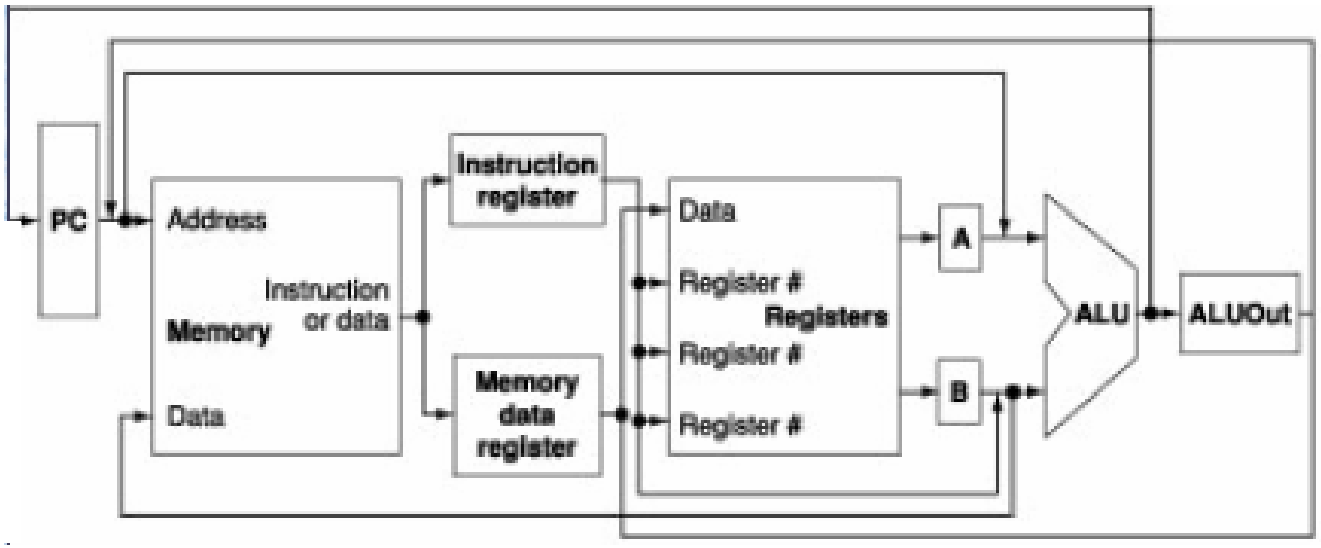
- Problema da máquina de ciclo único



- Implementação multiciclo:
  - Cada fase de execução da instrução em um ciclo
  - Unidades funcionais podem ser compartilhadas.
    - Ou seja, podem ser usadas mais de uma vez por instrução
  - Redução da quantidade de hardware necessário
- Principais vantagens do multiciclo:
  - Instruções são executadas em quantidades diferentes de períodos de clock
    - CPI variável. Caso comum pode ser melhorado.
  - É possível compartilhar unidades funcionais
    - Redução de custo.

Visão de alto nível do caminho de dados multiciclo

- Memória e ALU unificadas.
- Registradores RI e MDR
  - Os dois valores são necessários no mesmo ciclo de clock.
- Registradores A e B
  - Armazenam os valores lidos no banco de registradores
  - Mais de uma leitura é possível por execução de instrução.
- O registrador SaidaALU armazena a saída da ALU



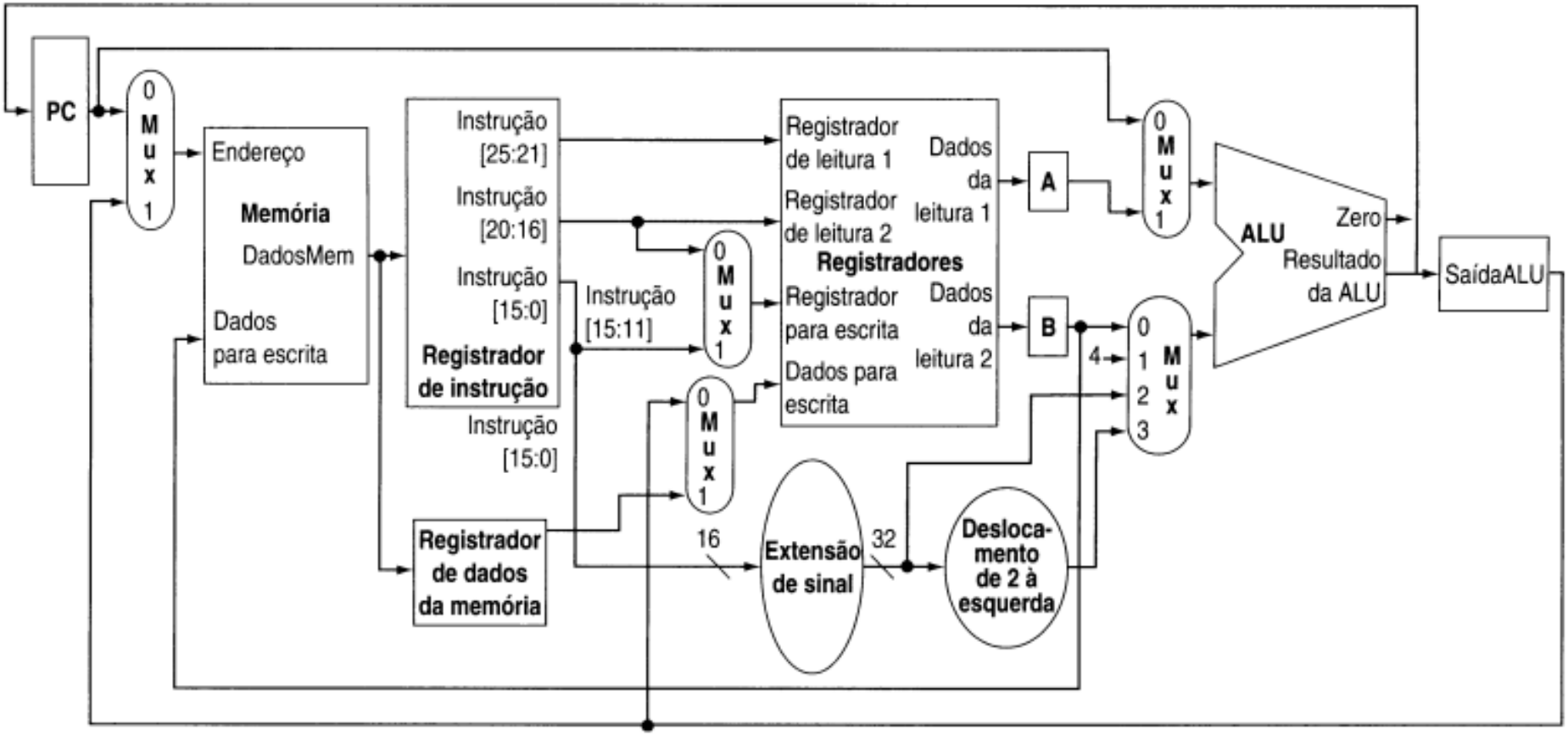
Visão de alto nível do caminho de dados multiciclo

- Várias unidades funcionais são compartilhadas para diferentes finalidades
  - Multiplexadores adicionais devem ser incluídos e os existentes devem ser expandidos.

- A substituição das três ALUs do caminho de dados de ciclo único por uma única ALU
  - É necessário acomodar todas as entradas das três ALUs diferentes na implementação monociclo.
  - Um multiplexador adicional é incluído para a primeira entrada da ALU
    - Escolher entre os registradores A e o PC

- O multiplexador na segunda entrada da ALU muda de duas para quatro entradas:
  - As duas entradas adicionais são:
    - a constante 4 (incremento o PC)
    - e o campo offset com sinal estendido e deslocado (cálculo do endereço do desvio)

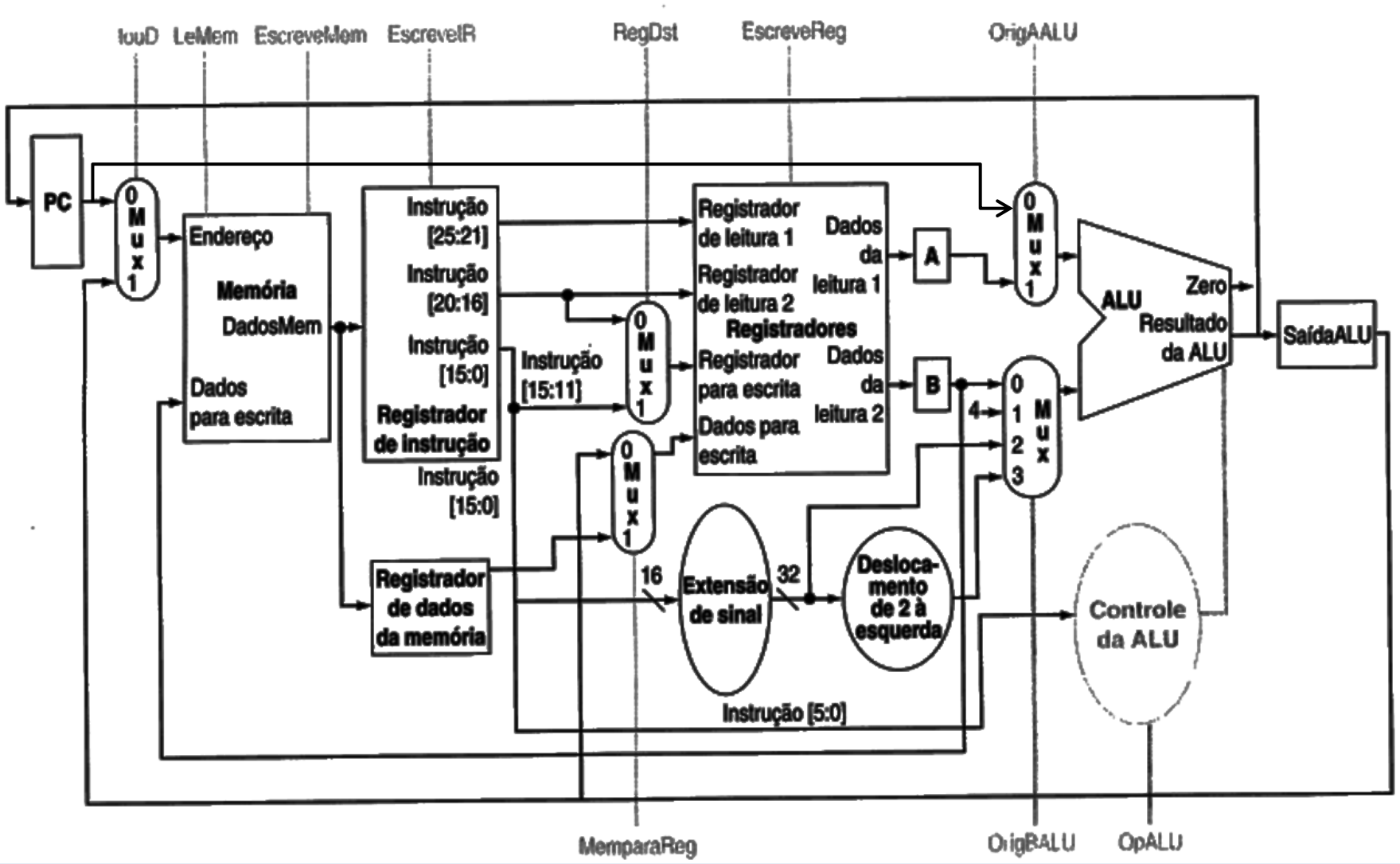
O caminho de dados multiciclo para o MIPS manipular instruções básicas



Unidade de controle principal

- Implementação multiciclo exige um conjunto diferente de sinais de controle:
  - PC, memória, registradores e IR: sinal de escrita
  - Memória: sinal de leitura
  - UAL: mesmo controle da implementação monociclo
  - Multiplexador de 4 entradas: 2 linhas de controle
  - Todos os demais multiplexadores: uma linha de controle
- Para instruções jump e branch equal existem três fontes possíveis para o PC:
  - **Não tomar o branch:** PC + 4
  - **Branch:** SaidaALU (desvio condicional)
  - **Jump:** 26 últimos bits do IR acrescido de 2 zeros à direita, e concatenados com os 4 MSB's do PC
- Controle de escrita do PC:
  - Incremento normal e deslocamento incondicional, o PC é escrito incondicionalmente
  - Se for um desvio condicional, passa o valor de UALSaída, somente se os registradores forem iguais
  - Dois sinais de escrita: PCEsc e PCEscCond

O caminho de dados multiciclo com as linhas de controle indicadas



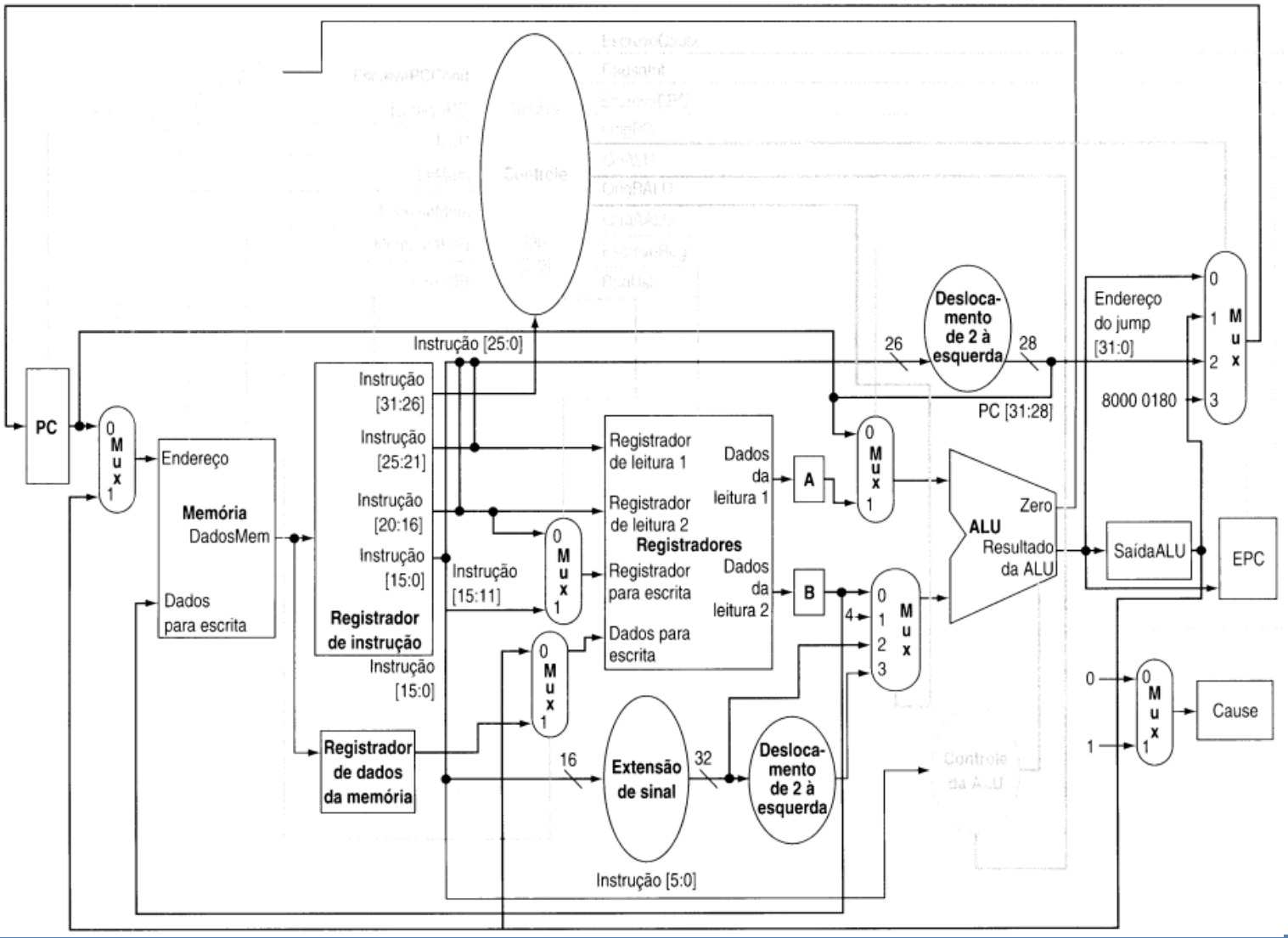
Ações dos sinais de controle de 1 bit

Nome do sinal	Efeito quando inativo	Efeito quando ativo
RegDst	O número do registrador de destino do banco de registradores para a entrada “Registrador para escrita” vem do campo rt.	O número do registrador de destino do banco de registradores para a entrada “Registrador para escrita” vem do campo rd.
EscreveReg	Nenhum.	O registrador de uso geral selecionado pelo número na entrada “Registrador para escrita” é escrito como valor da entrada “Dados para escrita”.
OrigAALU	O primeiro operando da ALU é o PC.	O primeiro operando da ALU vem do registrador A.
LeMem	Nenhum.	O conteúdo da memória no local especificado pela entrada Endereço é colocado na saída Dados da memória.
EscreveMem	Nenhum.	O conteúdo da memória no local especificado pela entrada Endereço é substituido pelo valor na entrada “Dados para escrita”.
MemparaReg	O valor enviado para a entrada “Dados para escrita” do banco de registradores vem de SaídaALU.	O valor enviado para a entrada “Dados para escrita” do banco de registradores vem do MDR.
IouD	O PC é usado para fornecer o endereço para a unidade de memória.	SaídaALU é usado para fornecer o endereço para a unidade de memória.
IRWrite	Nenhum.	A saída da memória é escrita no IR.
EscrevePC	Nenhum.	O PC é escrito; a origem é controlada por OrigPC.
EscrevePCCond	Nenhum.	O PC é escrito se a saída Zero da ALU também estiver ativa.

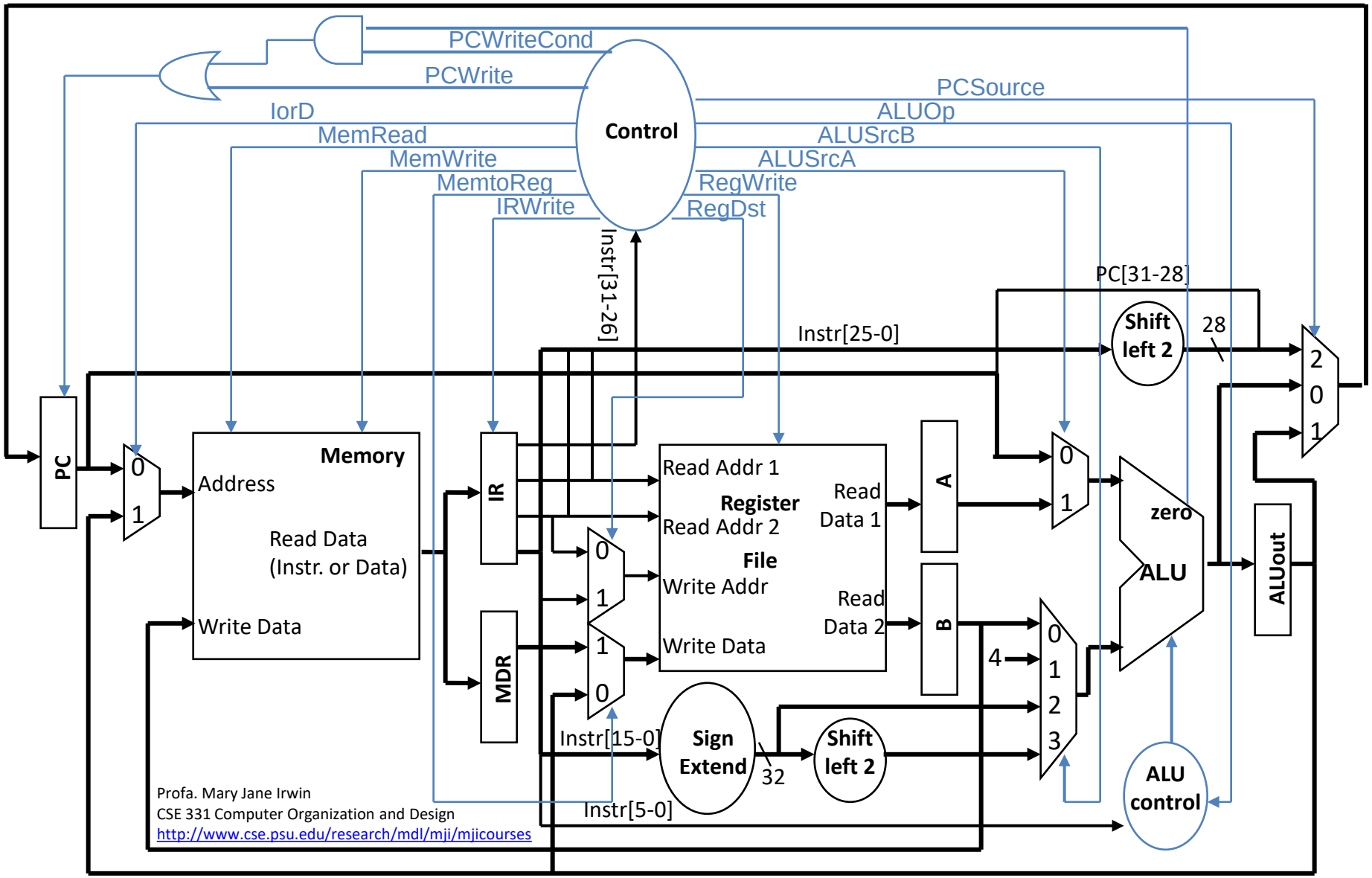
Ações dos sinais de controle de 2 bit

Nome do sinal	Valor (binário)	Efeito
OpALU	00	A ALU realiza uma operação de adição.
	01	A ALU realiza uma operação de subtração.
	10	O campo funct da instrução determina a operação da ALU.
OrigBALU	00	A segunda entrada para a ALU vem do registrador B.
	01	A segunda entrada da ALU é a constante 4.
	10	A segunda entrada da ALU são os 16 bits menos significativos com sinal estendido do IR.
	11	A segunda entrada da ALU são os 16 bits menos significativos com sinal estendido do IR deslocados em 2 bits para a esquerda.
OrigPC	00	A saída da ALU (PC + 4) é enviada ao PC para escrita.
	01	O conteúdo da SaídaALU (o endereço de destino do desvio) é enviado ao PC para escrita.
	10	O endereço de destino do jump (IR[25:0]) deslocado de 2 bits para a esquerda e concatenado com PC + 4[31:28] é enviado ao PC para escrita.

Caminho de dados completo para a implementação multiciclo com as linhas de controle necessárias



Caminho de dados completo para a implementação multiciclo com as linhas de controle necessárias



Dividindo a execução da instrução em ciclos de clock

- Dividir as instruções em mais de um ciclo de clock:
  - Cada etapa da instrução é realizada em um ciclo de clock.
- A organização da execução de cada instrução segue um número de etapas, que depende do tipo da instrução
  - Cada etapa deve realizar apenas:
    - uma operação de ALU;
    - um acesso à memória ou;
    - um acesso ao banco de registradores.
  - Com essa restrição, o ciclo de clock possui a duração de da etapa mais longa.
- Os registradores armazenam dados para os próximos ciclos da mesma instrução
  - Ex: Registradores A, B, MDR, SaidaALU.

Dividindo a execução da instrução em ciclos de clock

1. Busca da instrução
   - (PC + Memória)
2. Decodificação da instrução e busca dos registradores
   - (Controle e banco de registradores)
3. Execução, cálculo do endereço de memória ou conclusão do desvio
   - (UAL)
4. Acesso à memória ou conclusão de instrução tipo R
   - (Memória ou banco de registradores)
5. Conclusão de leitura de memória
   - (Banco de registradores)

Dividindo a execução da instrução em ciclos de clock

- 1. Busca da instrução
  - IR <= Memória[PC]
  - PC <= PC+4
  - Envia o endereço armazenado no PC para a memória
    - Escreve a instrução no RI

- 2. Decodificação e busca dos registradores
    - Não se sabe qual instrução está no IR.
    - Os registradores são lidos para evitar a perda de tempo durante a execução (por exemplo ler rs e rt)
      - Alguns valores podem ser descartadas após a decodificação da instrução
  - Carrega os registradores de entrada da ALU e do endereço de desvio condicional (salvo em SaidaALU)
    - A <= Reg[IR[25-21]];
    - B <= Reg[IR[20-16]];
    - SaidaALU <= PC + extensão de sinal (IR[15-0] << 2);

Dividindo a execução da instrução em ciclos de clock

- 3. Execução, cálculo do endereço de memória ou conclusão do desvio
  - a) Referência à memória (LW e SW)
    - SaidaALU <= A + extensão de sinal IR[15-0]
  - b) Instrução aritmética ou lógica (tipo R)
    - SaidaALU <= A op B
  - c) Desvio condicional (BEQ)
    - Se (A == B): PC <= SaidaALU
  - d) Desvio incondicional (J)
    - PC <= PC [31-28] || (IR[25-0]<<2)

Dividindo a execução da instrução em ciclos de clock

- 4. Etapa de acesso à memória ou conclusão de instrução tipo R
  - a) Referência à memória (LW ou SW)
    - MDR = Memória [SaidaALU]; (LW)

    ou
    - Memória [SaidaALU] <= B; (SW)
  - b) Instruções aritméticas ou lógicas (tipo R)
    - Reg[IR[15-11]] <= UALSaída;

- 5. Etapa de conclusão de acesso à memória (LW)
  - Reg[IR[20-16]] <= MDR

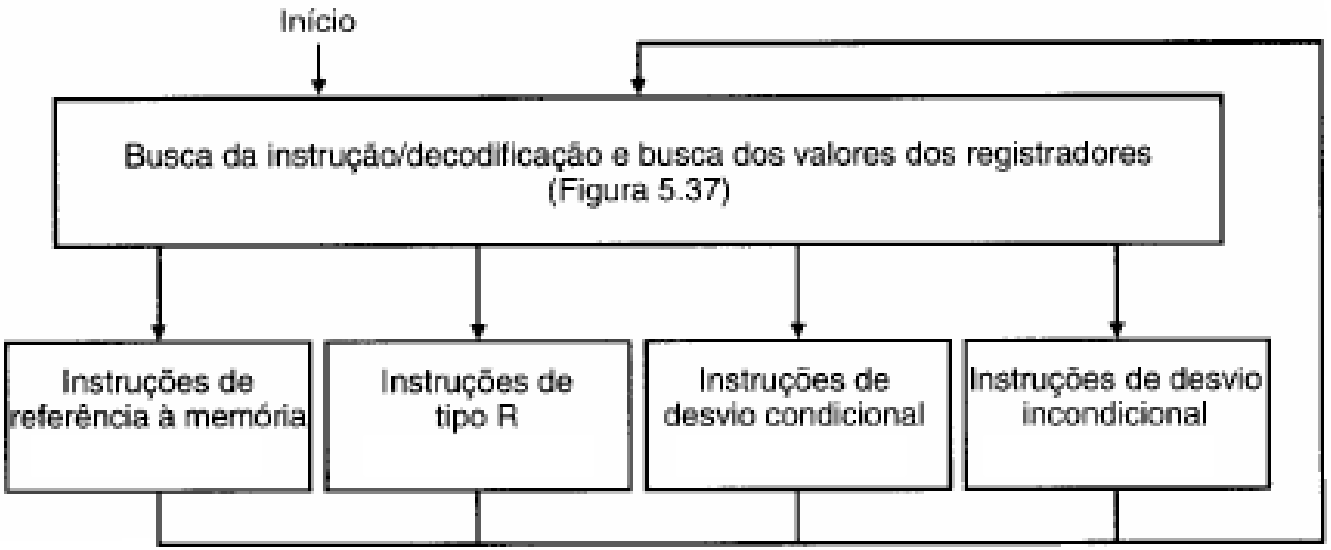
Dividindo a execução da instrução em ciclos de clock

Etapa	Ação para instruções tipo R	Ação para instruções de acesso à memória	Ação para desvios	Ação para jumps
Busca da instrução	IR <= Memória[PC] PC <= PC + 4			
Decodificação da instrução e busca dos registradores	A <= Reg[IR[25:21]] B <= Reg[IR[20:16]] SaídaULA <= PC + (estende-sinal (IR[15:0]) << 2)			
Execução, cálculo do endereço ou conclusão do desvio/jump	SaídaALU <= A op B	SaídaALU <= A + estende_sinal (IR[15:0])	If(A==B) PC <= SaídaALU	PC <= {PC[31:28], (IR[25:0], 2'b00)}
Acesso à memória ou conclusão de instrução do tipo R	Reg[IR[15:11]] <= SaídaALU	Load: MDR <= Memória[SaídaALU] ou Store: Memória[SaídaALU] <= B		
Conclusão da leitura da memória		Load: Reg[IR[20:16]] <= MDR		

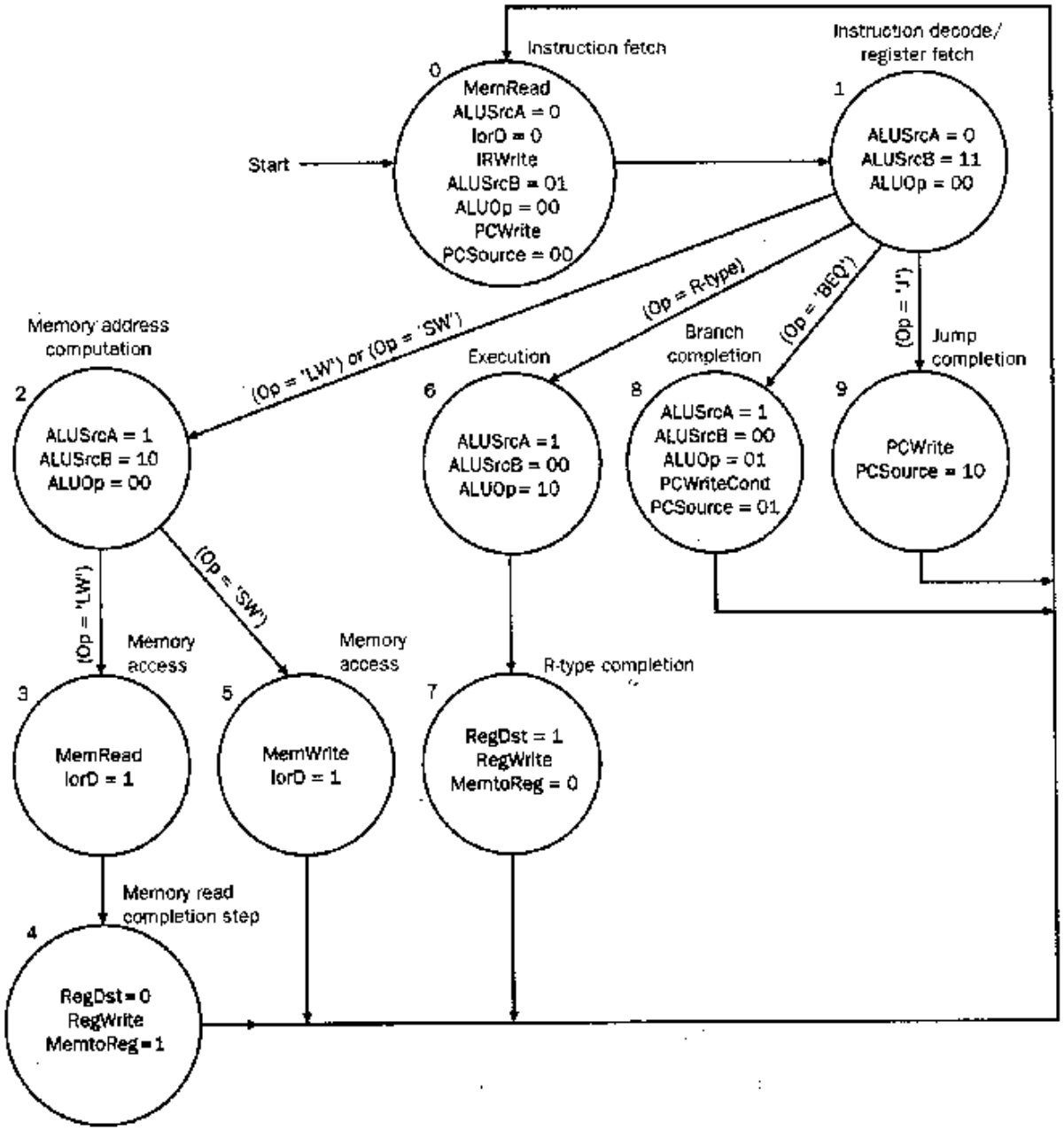
DEFININDO O CONTROLE

Definindo o controle

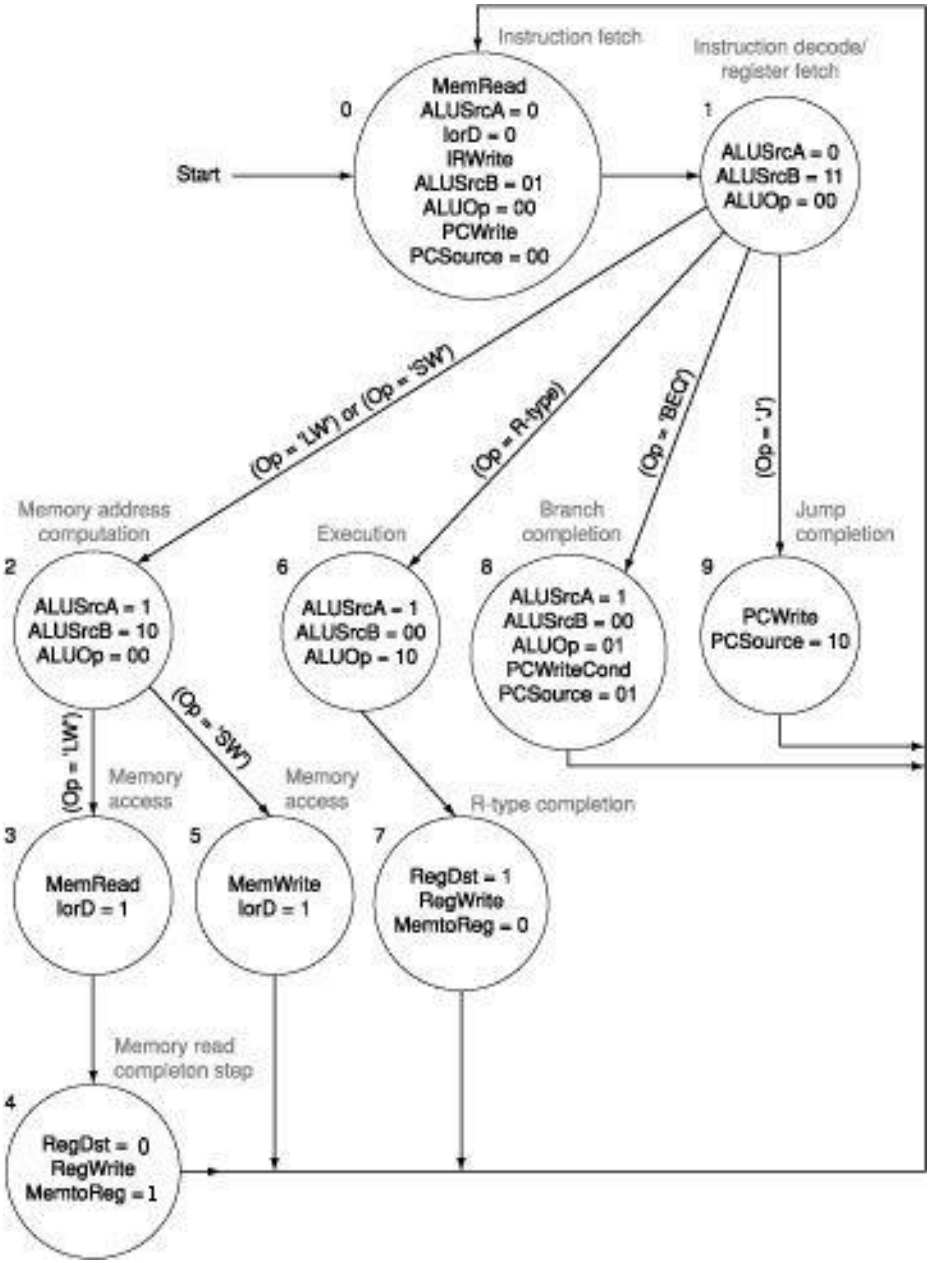
- Controle da implementação multiciclo pode ser realizado:
  - Máquina de estados finitos (MEF)
  - Microprogramação
- Máquina de estados finitos
  - Visão de alto nível



O controle da máquina de estados finitos completo para o caminho de dados



O controle da máquina de estados finitos completo para o caminho de dados



Bibliografia

1. PATTERSON, D.A; HENNESSY, J.L. **Organização e Projeto de Computadores: A Interface Hardware/Software**. 3a. Ed. Elsevier, 2005.
   – Capítulo ***5.***

2. Notas de aula do prof. Luciano J. Senger:
   – http://www.ljsenger.net/classroom.html

3. Notas de aula da Profa. Mary Jane Irwin
   – CSE 331 Computer Organization and Design
   – http://www.cse.psu.edu/research/mdl/mji/mjicourses

FIM – Aula 09

- FIM:
  - **Aula 09** – Caminho de dados e controle 2 – MIPS Multiciclo
- Próxima aula:
  - **Aula 10** – Pipelining: Introdução