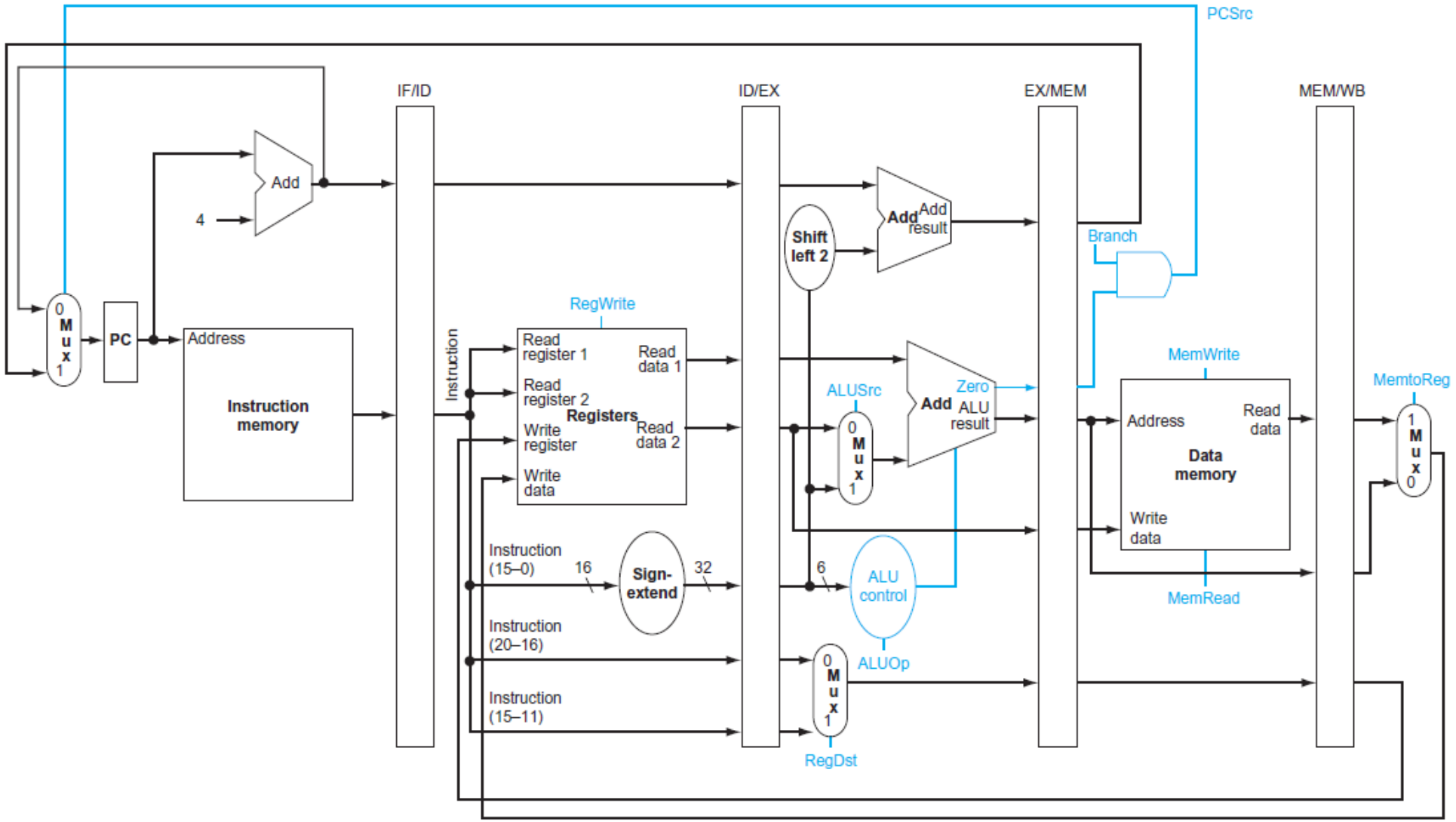
SSC0510
Arquitetura de Computadores

5ª Aula – Pipeline

Profa. Sarita Mazzini Bruschi

sarita@icmc.usp.br

Pipeline – Unidade de Controle



Pipeline – Unidade de Controle

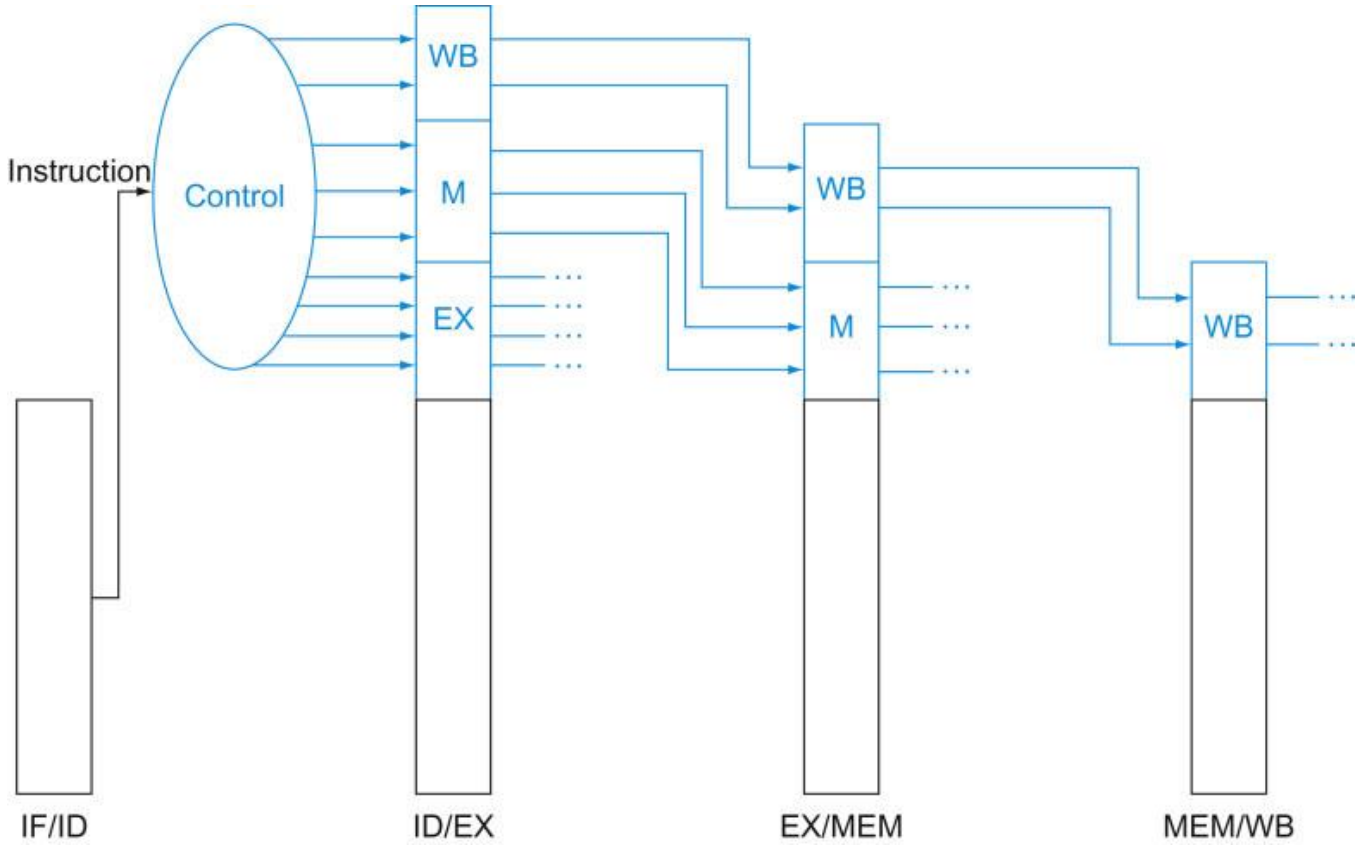
Signal name	Effect when deasserted (0)	Effect when asserted (1)
RegDst	The register destination number for the Write register comes from the rt field (bits 20:16).	The register destination number for the Write register comes from the rd field (bits 15:11).
RegWrite	None.	The register on the Write register input is written with the value on the Write data input.
ALUSrc	The second ALU operand comes from the second register file output (Read data 2).	The second ALU operand is the sign-extended, lower 16 bits of the instruction.
PCSrc	The PC is replaced by the output of the adder that computes the value of PC + 4.	The PC is replaced by the output of the adder that computes the branch target.
MemRead	None.	Data memory contents designated by the address input are put on the Read data output.
MemWrite	None.	Data memory contents designated by the address input are replaced by the value on the Write data input.
MemtoReg	The value fed to the register Write data input comes from the ALU.	The value fed to the register Write data input comes from the data memory.

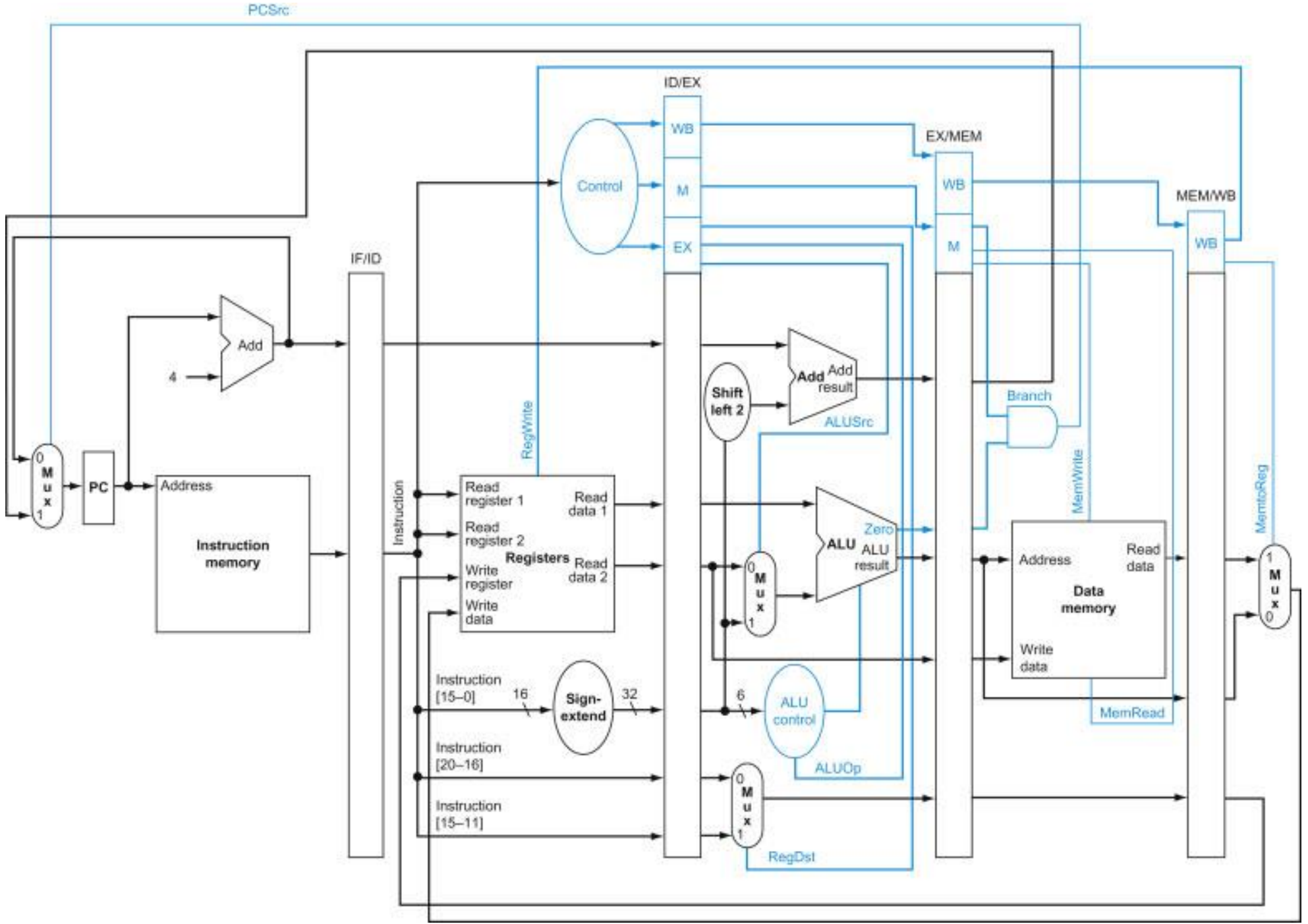
Pipeline – Unidade de Controle

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

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

Pipeline – Unidade de Controle




PCSrc
ID/EX
EX/MEM
MEM/WB
WB
M
EX
Control
IF/ID
Add
4
0
M
u
x
1
PC
Address
Instruction memory
Instruction
RegWrite
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
Instruction [15–0]
16
Sign-extend
32
Instruction [20–16]
Instruction [15–11]
Shift left 2
Add
Add result
ALUSrc
ALU
Zero
ALU result
6
ALU control
ALUOp
RegDst
Branch
MemWrite
MemRead
MemtoReg
Address
Read data
Data memory
Write data
1
0

Pipeline

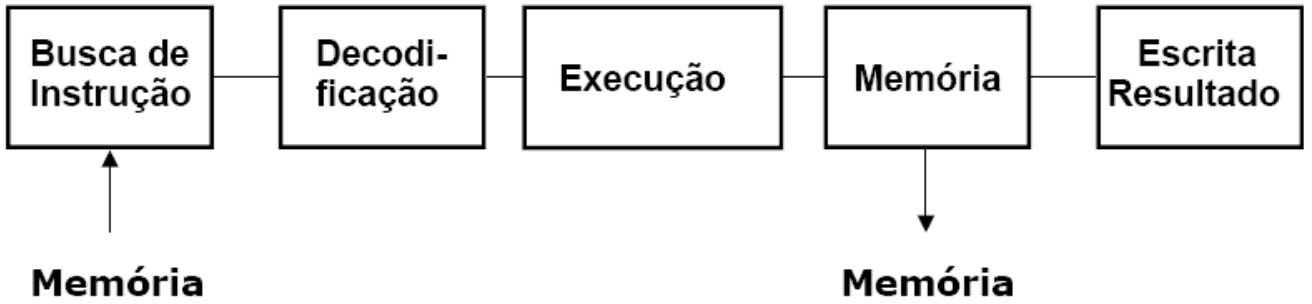
- Dependências ou Conflitos (*Hazards*)
  - Conflitos Estruturais
    - Pode haver acessos simultâneos à memória feitos por 2 ou mais estágios.
  - Dependências de Dados
    - As instruções dependem de resultados de instruções anteriores, ainda não completadas.
  - Dependências de Controle
    - A próxima instrução não está no endereço subsequente ao da instrução anterior.

Pipeline
Conflitos Estruturais

- Acessos Concorrentes à Memória
  - Uso de memórias multi-portas ou com múltiplos bancos com acessos independentes.
- Leitura de instrução e leitura/escrita de dados simultâneos à memória
  - Uso de arquitetura “Harvard” com caches de dados e instrução separados.
- Acesso simultâneo ao banco de registradores
  - Uso de banco de registradores com múltiplas portas.
- Uso simultâneo de uma mesma unidade funcional
  - Replicação da unidade funcional ou implementação “pipelined” dessa unidade.

Pipeline
Conflitos Estruturais

- Problema:
  - Acessos simultâneos à memória por 2 ou mais estágios



- Soluções:
  - Caches separadas de dados e instruções
  - Memória com múltiplos de acesso independentes

Pipeline
Dependência de Dados

- Problema: uma instrução faz uso de um operando que vai ser produzido por uma outra instrução que ainda está no pipeline.
- A execução da instrução seguinte depende de operando calculado pela instrução anterior.
- Tipos de dependências de dados:
  - Dependência verdadeiras
  - Dependências falsas
    - Antidependência
    - Dependência de saída

Pipeline
Dependência de Dados

- Tipos de dependências de dados:
  - Dependências verdadeiras (diretas) ou RAW *(Read After Write)*:
    - Uma instrução utiliza um operando que é produzido por uma instrução anterior.
  - Dependências falsas:
    - Antidependência ou WAR (*Write After Read*):
      - Uma instrução lê um operando que é escrito por uma instrução sucessora.
    - Dependência de saída ou WAW (*Write After Write*):
      - Uma instrução escreve em um operando que é também escrito por uma instrução sucessora.

Pipeline
Dependência de Dados

Dependência direta:

add R1, R2, R3

sub R4, R1, R6

Uma instrução utiliza um operando que é produzido por uma instrução anterior

Antidependência:

sub R4, R1, R6

add R1, R2, R3

Uma instrução lê um operando que é escrito por uma instrução sucessora

Dependência de Saída:

add R4, R2, R3

sub R4, R1, R6

Uma instrução escreve em um operando que é também escrito por uma instrução sucessora

Pipeline
Dependência de Dados

- Dependências Verdadeiras (Direta):
  - O pipeline precisa ser parado durante certo número de ciclos (*interlock*);
  - Inserção de instruções de “nop” ou escalonamento adequado das instruções pelo compilador;
  - O adiantamento (*bypassing* ou *forwarding*) dos dados pode resolver em alguns casos.
- Dependências Falsas:
  - Não é um problema em pipelines onde a ordem de execução das instruções é mantida;
  - Problema em processadores superescalares;
  - A renomeação dos registradores é uma solução usual para este problema.

Pipeline
Dependência de Dados

- Escalonamento de Instruções
  - Exemplo:

    a = b + e;

    c = b + f;

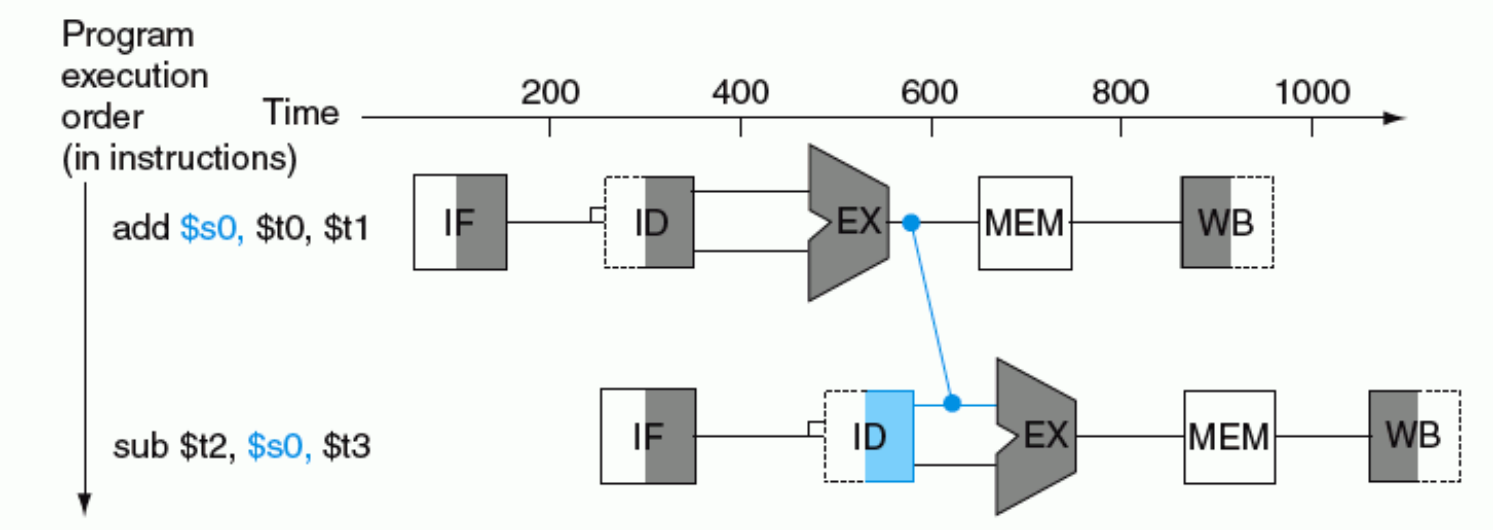
- Código gerado:

```
lw   $t1, 0 ($t0)
lw   $t2, 4 ($t0)
add  $t3, $t1, $t2
sw   $t3, 12($t0)
lw   $t4, 8 ($t0)
add  $t5, $t1, $t4
sw   $t5, 16($t0)
```

1. **Encontre as dependências verdadeiras**
2. **Insira “nops” de modo a evitar a parada do pipeline**
3. **Reordene de modo a minizar o número de “nops” inseridos**

Pipeline
Dependência de Dados

- Adiantamento de Dados
  - Caminho interno dentro do pipeline entre a saída e a entrada da ULA
    - Técnica conhecida como *forwarding* ou *bypassing*
    - Evita a *parada* do pipeline utilizando *buffers* internos em vez de esperar que o elemento de dado chegue nos registradores visíveis ao programador ou na memória



Pipeline
Dependência de Dados

- Adiantamento de Dados (cont.)
  - Em algumas situações, nem o *forwarding* pode resolver o problema de parada do pipeline