SSC0611
Arquitetura de Computadores

9ª Aula – Pipeline

Profa. Sarita Mazzini Bruschi

sarita@icmc.usp.br

Pipeline

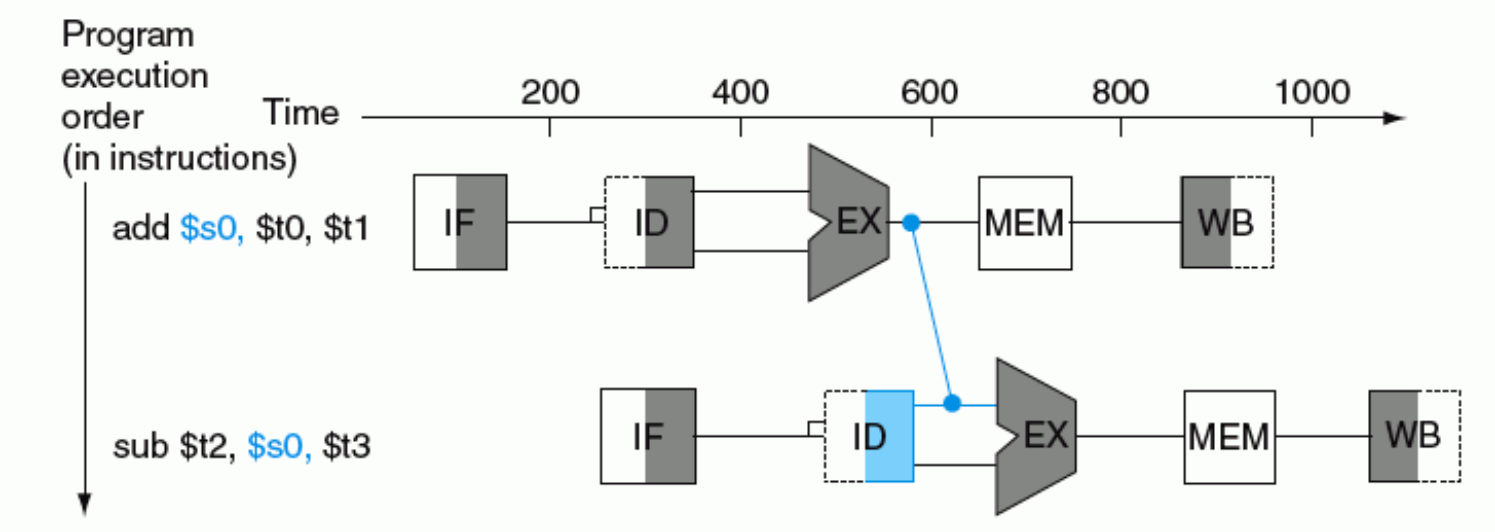
- Dependências ou Conflitos (*Hazards*)
  - Conflitos Estruturais
    - Pode haver acessos simultâneos à memória feitos por 2 ou mais estágios.
  - Dependências de Dados
    - As instruções dependem de resultados de instruções anteriores, ainda não completadas.
  - Dependências de Controle
    - A próxima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

Pipeline
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

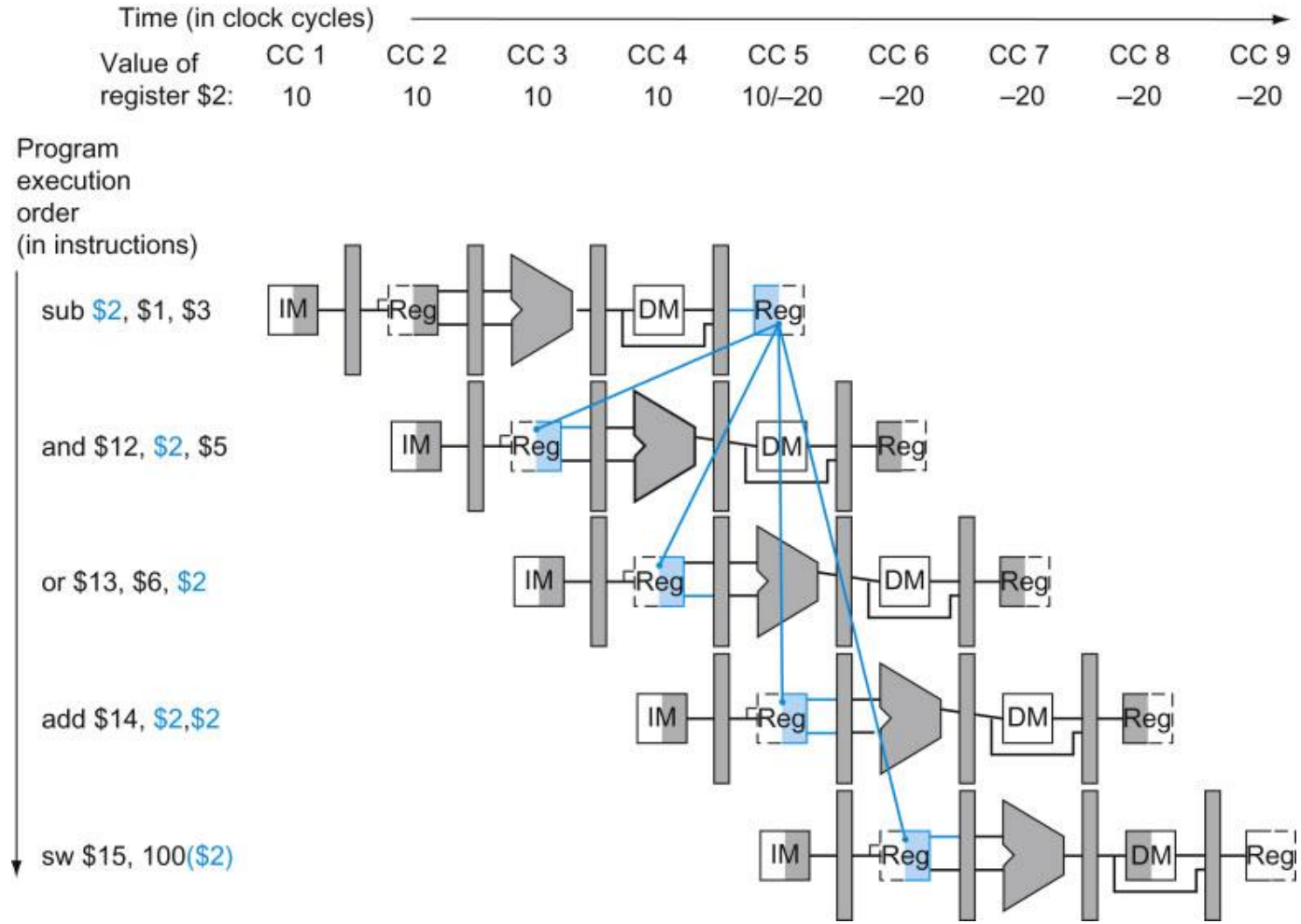
Pipeline
Dependência de Dados

- 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



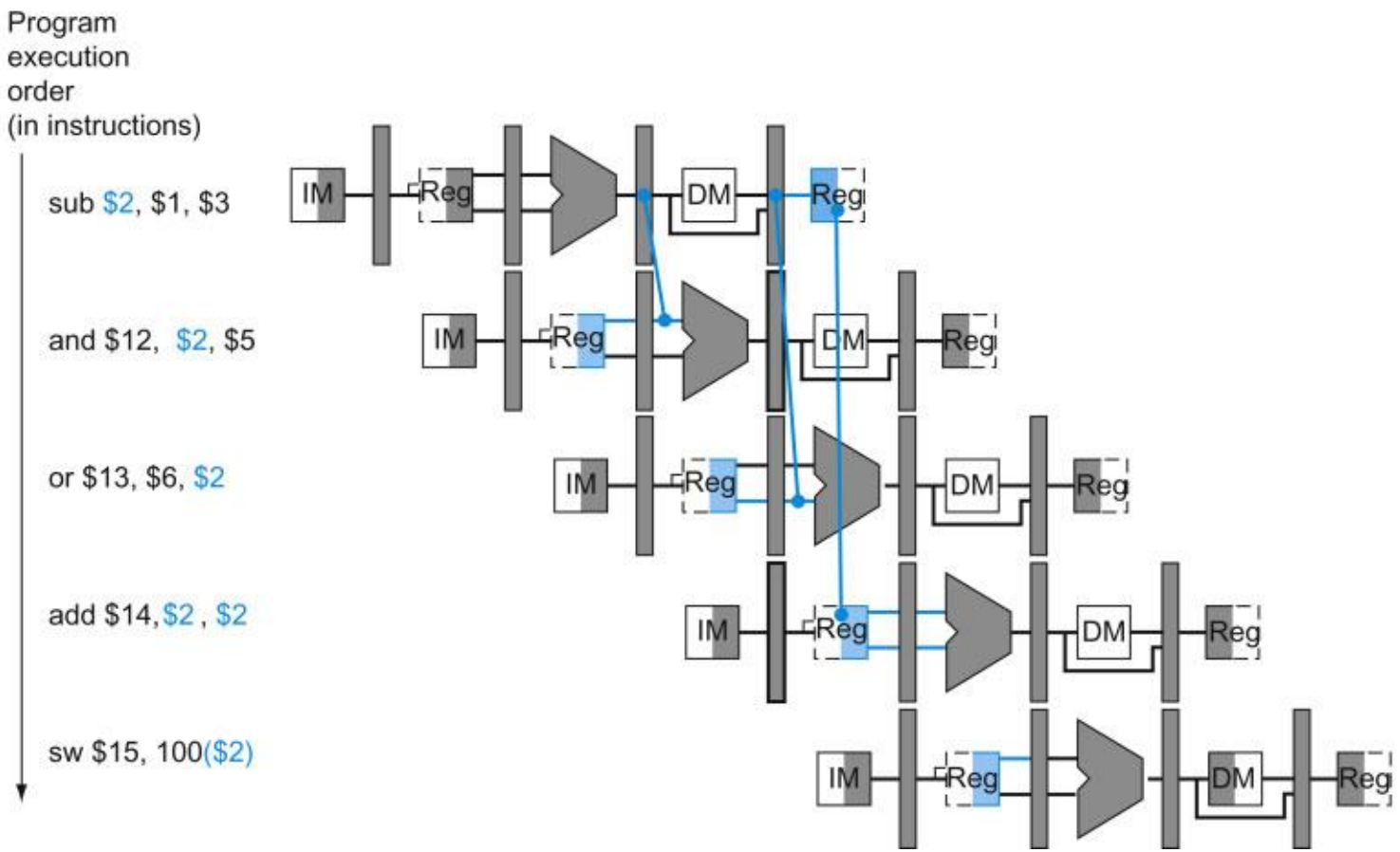
Pipeline
Dependências de Dados

```
sub    $2, $1,$3       # Register $2 written by sub
and    $12,$2,$5       # 1st operand($2) depends on sub
or     $13,$6,$2       # 2nd operand($2) depends on sub
add    $14,$2,$2       # 1st($2) & 2nd($2) depend on sub
sw     $15,100($2)     # Base ($2) depends on sub
```



Time (in clock cycles) →

	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6	CC 7	CC 8	CC 9
Value of register $2:	10	10	10	10	10/–20	–20	–20	–20	–20
Value of EX/MEM:	X	X	X	–20	X	X	X	X	X
Value of MEM/WB:	X	X	X	X	–20	X	X	X	X



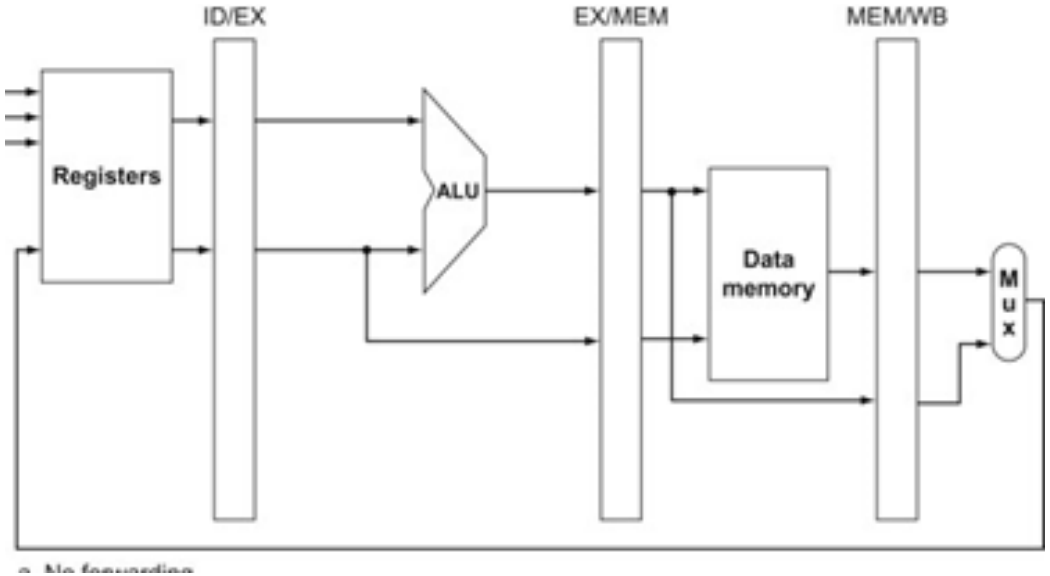
Pipeline
Dependência de Dados

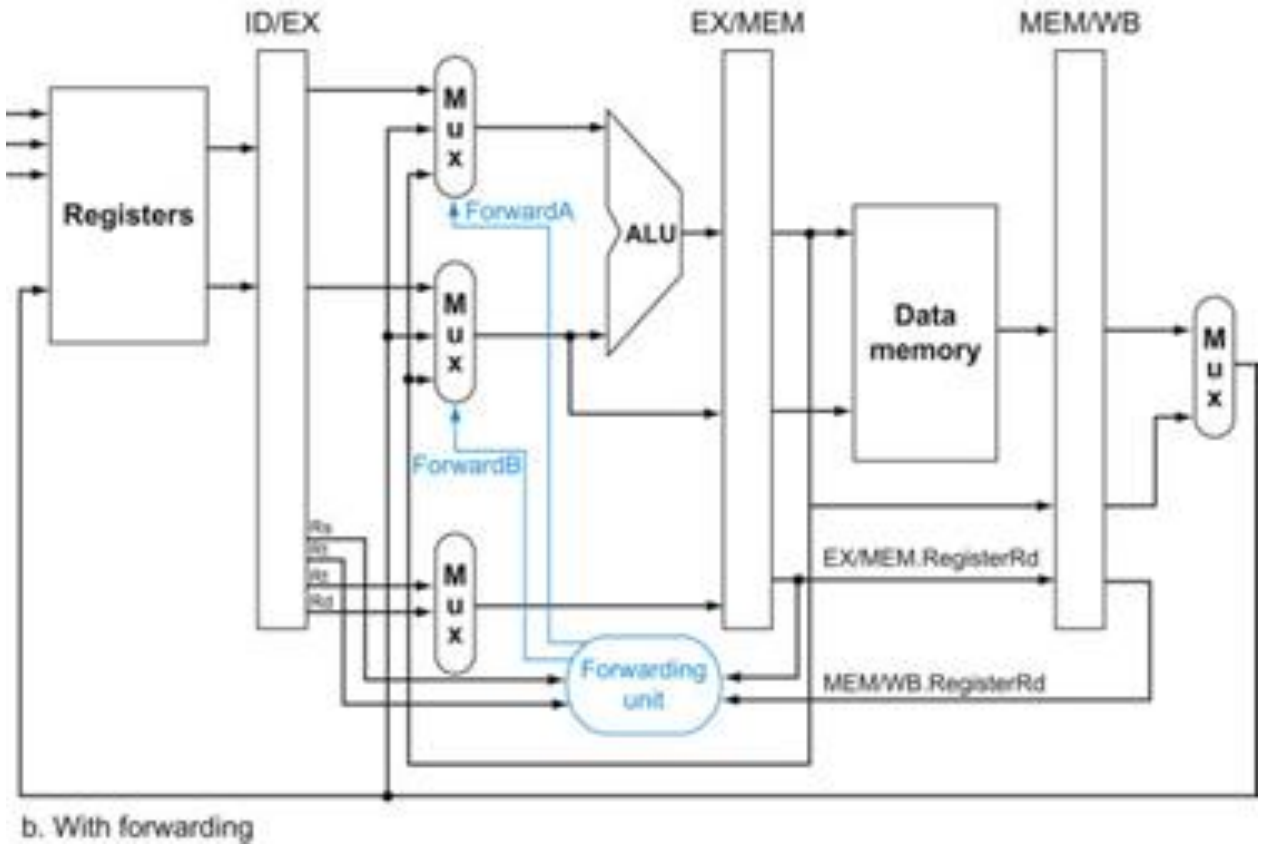
- Como implementar o adiantamento (*forwarding*)?
- Condições para que o adiantamento seja considerado:
  - Destino da instrução que está entre os ciclos de EX e MEM é igual a um dos registradores de origem da instrução que está entre os ciclos de ID e EX
    - 1a. EX/MEM.RegisterRd = ID/EX.RegisterRs
    - 1b. EX/MEM.RegisterRd = ID/EX.RegisterRt
  - Destino da instrução que está entre os ciclos de MEM e WB é igual a um dos registradores de origem da instrução que está entre os ciclos de ID e EX
    - 2a. MEM/WB.RegisterRd = ID/EX.RegisterRs
    - 2b. MEM/WB.RegisterRd = ID/EX.RegisterRt

Pipeline
Dependência de Dados

- No exemplo do código anterior, tem-se as seguintes dependências:
  - Entre as instruções: `sub $2,$1,$3` e `and $12,$2,$5`
    - Da numeração do slide anterior, qual é essa dependência?
  - Entre as instruções: `sub $2,$1,$3` e `or $16,$6,$2`
    - Da numeração do slide anterior, qual é essa dependência?
  - As duas dependências `sub-add` não são problema pois o banco de registradores irá fornecer os dados corretos no ciclo de decodificação
  - Não existe dependência entre as instruções `sub` e `sw`

Pipeline
Dependência de Dados





b. With forwarding

Pipeline
Dependência de Dados

Mux control	Source	Explanation
ForwardA = 00	ID/EX	The first ALU operand comes from the register file.
ForwardA = 10	EX/MEM	The first ALU operand is forwarded from the prior ALU result.
ForwardA = 01	MEM/WB	The first ALU operand is forwarded from data memory or an earlier ALU result.
ForwardB = 00	ID/EX	The second ALU operand comes from the register file.
ForwardB = 10	EX/MEM	The second ALU operand is forwarded from the prior ALU result.
ForwardB = 01	MEM/WB	The second ALU operand is forwarded from data memory or an earlier ALU result.

Pipeline
Dependência de Dados

- Definição das condições para a detecção de dependência de dados:
  - Dependência EX:

    if (EX/MEM.RegWrite
    and (EX/MEM.RegisterRd ≠ 0)
    and (EX/MEM.RegisterRd = ID/EX.RegisterRs)) ForwardA = 10

    if (EX/MEM.RegWrite
    and (EX/MEM.RegisterRd ≠ 0)
    and (EX/MEM.RegisterRd = ID/EX.RegisterRt)) ForwardB = 10

Pipeline
Dependência de Dados

- Definição das condições para a detecção de dependência de dados:
  - Dependência MEM:

    if (MEM/WB.RegWrite
    and (MEM/WB.RegisterRd ≠ 0)
    and (MEM/WB.RegisterRd = ID/EX.RegisterRs)) ForwardA = 01

    if (MEM/WB.RegWrite
    and (MEM/WB.RegisterRd ≠ 0)
    and (MEM/WB.RegisterRd = ID/EX.RegisterRt)) ForwardB = 01

Pipeline
Dependência de Dados

- Problema: quando se está acumulando os valores de um vetor em um registrador e lê e escreve no mesmo registrador:

  add $1,$1,$2

  add $1,$1,$3

  add $1,$1,$4

  . . .

  Ao final de qual estágio do pipeline queremos o valor do registrador $1?

Pipeline
Dependência de Dados

- Lógica

if (MEM/WB.RegWrite
and (MEM/WB.RegisterRd ≠ 0)
and (EX/MEM.RegisterRd ≠ ID/EX.RegisterRs)
and (MEM/WB.RegisterRd = ID/EX.RegisterRs)) ForwardA = 01

if (MEM/WB.RegWrite
and (MEM/WB.RegisterRd ≠ 0)
and (EX/MEM.RegisterRd ≠ ID/EX.RegisterRt)
and (MEM/WB.RegisterRd = ID/EX.RegisterRt)) ForwardB = 01

Pipeline
Dependência de Dados