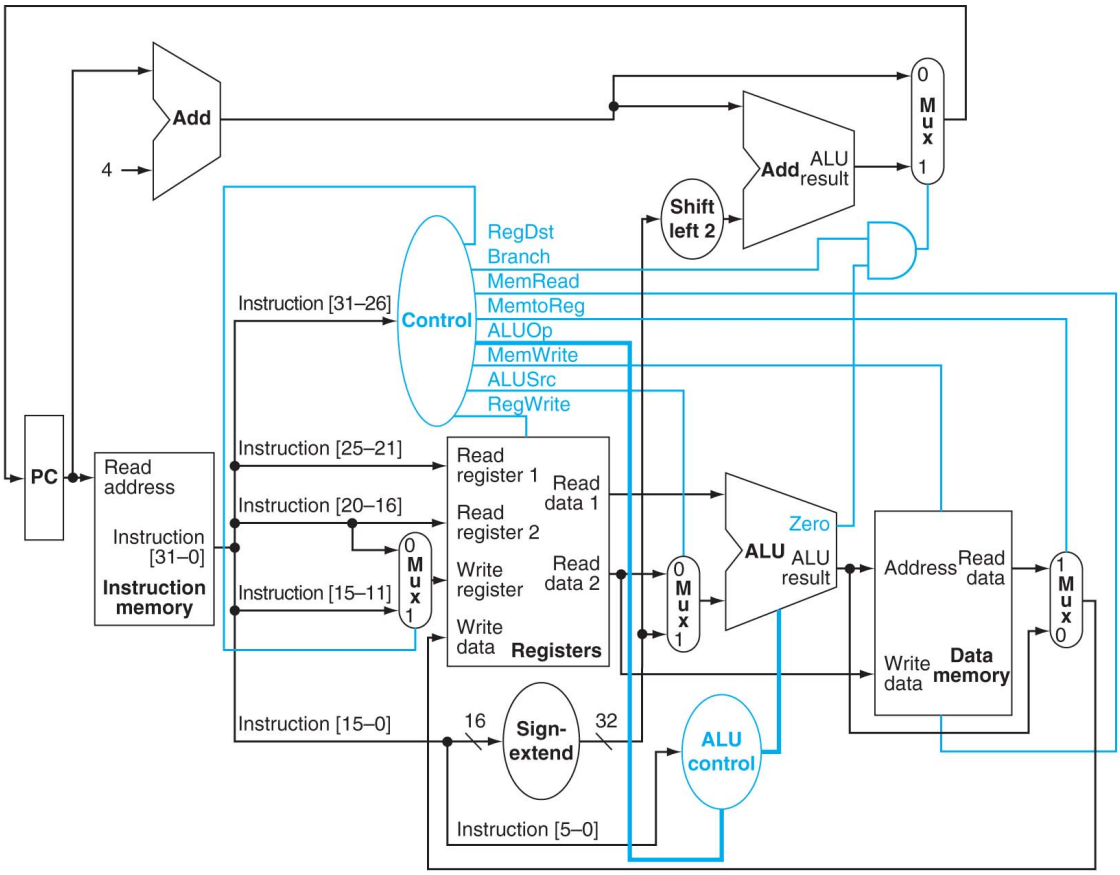
Arquitetura de Computadores I
4ª série de problemas
7.12.2015

I – Single-Cycle Datapath



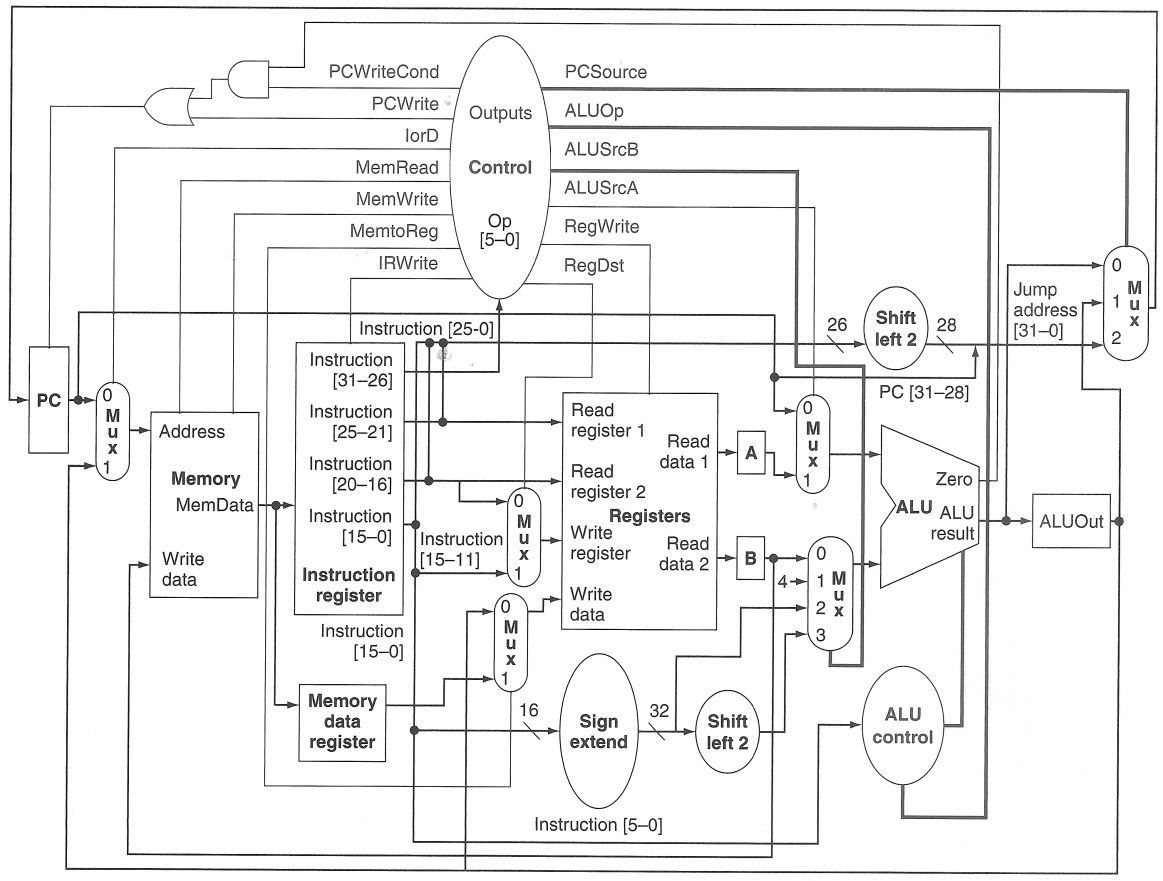
Instruction	RegDst	ALUSrc	Memto-Reg	Reg-Write	Mem-Read	Mem-Write	Branch	ALUOp1	ALUOp0
R-format	1	0	0	1	0	0	0	1	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	X	0	0	0	1	0	1

1. Descreva o efeito que teriam os seguintes bloqueios ("*stuck-at*") dos sinais de control a 0 ou a 1, indicando para cada caso quais as instruções que não executariam corretamente:
   a. RegWrite = 0
   b. RegWrite = 1
   c. ALUop0 = 0
   d. ALUop0 = 1
   e. ALUop1 = 0
   f. ALUop1 = 1
   g. Branch = 0
   h. Branch = 1
   i. MemRead = 0
   j. MemRead = 1

k. MemWrite = 0
l. MemWrite = 1

2. Pretende-se que o processador execute tambem a instrução ***jump register***. Indique o que seria necessário acrescentar ao datapath e as alterações a introduzir na tabela com os valores dos sinais de control (Nota: na instrução ***jr*** o campo rs do código de instrução indica o registo que contem o endereço da instrução seguinte a executar).

II - **Multi-Cycle Datapath**

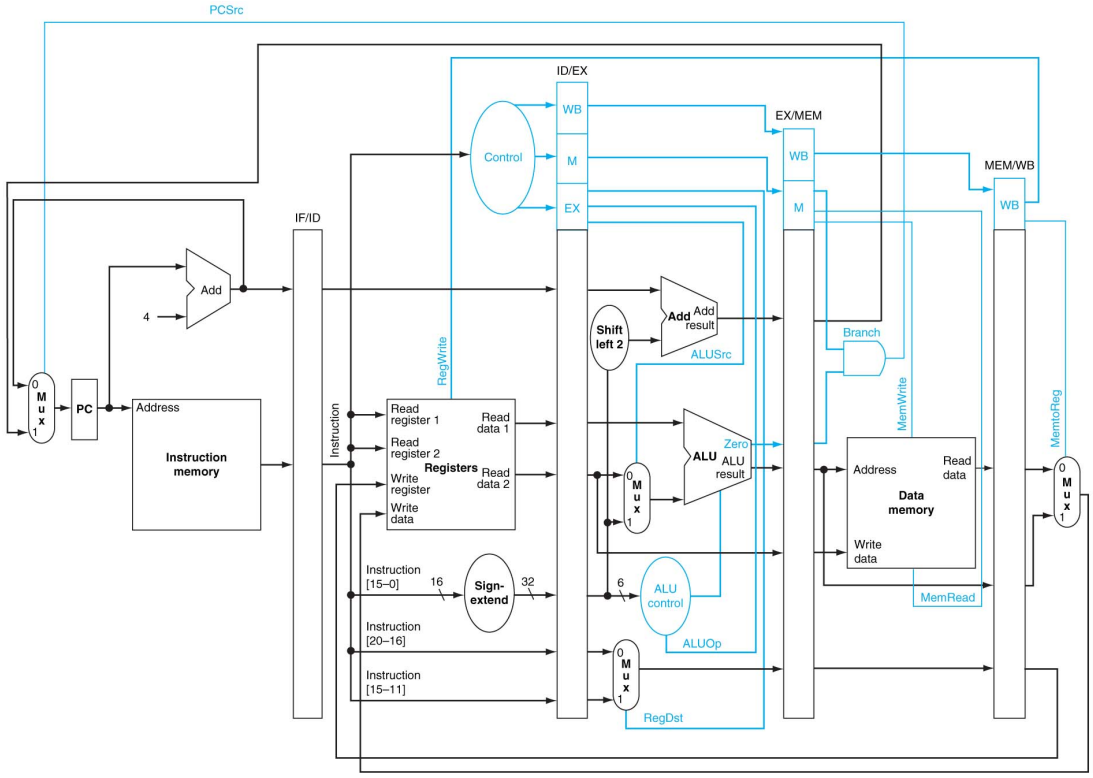


3. Descreva o efeito que teriam os seguintes bloqueios ("*stuck-at*") dos sinais de control a 0 ou a 1, indicando para cada caso quais as instruções que não executariam corretamente:
   a. RegWrite = 0
   b. MemRead = 0
   c. MemRead = 1
   d. MemWrite = 0
   e. MemWrite = 1
   f. IRWrite = 0
   g. IRWrite = 1
   h. PCWrite = 0
   i. PCWrite = 1
   j. PCWriteCond = 0
   k. PCWriteCond = 1
4. Pretende-se que o processador execute tambem a instrução ***add immediate*** **(addi)**. Indique as modificações a introduzir na máquina de estados que representa a unidade de controle e

nas respetivas saídas e eventuais alterações necessárias no datapath. Indique o valor dos sinais de control em cada ciclo da execução de addi.

5. Assuma que a unidade de controle é microprogramada. Escreva um microprograma que implemente **addi**.

III – Pipelined Datapath



6. Utilizando a representação multicycle do pipeline mostre os reencaminhamentos de dados (*forwarding paths*) necessários para executar a seguinte sequência de instruções:

```
add  $3,  $4,  $6
sub  $5,  $3,  $2
lw   $7,  100($5)
add  $8,  $7,  $2
```

7. Identifique todas as dependências de dados na sequência de instruções seguinte. Quais dessas dependências podem ser resolvidas com *forwarding* e quais e quais introduzem uma bolha (*bubble*) no pipeline.

```
add  $3,  $4,  $2
sub  $5,  $3,  $1
lw   $6,  200($3)
add  $7,  $3,  $6
```

8. A seguinte sequência de instruções é executada no pipeline:

```
lw   $5,  40($2)
add  $6,  $3,  $2
or   $7,  $2,  $1
and  $8,  $2,  $3
sub  $9,  $2,  $1
```

Cada registo tem o valor inicial 10_{10} + número do registo (p.ex. o registo $8 tem o valor 18_{10}). Cada posição da memória de dados tem o valor inicial 1000_{10} + o seu endereço (p.ex. Memory[8] tem o valor 1008_{10}). No ciclo 5 o PC tem o valor 100_{10}, o endereço da instrução **sub.**

Indique o valor em cada campo dos 4 registos do pipeline, IF/ID, ID/EX, EX/MEM, MEM/WB, no ciclo 5.