Arquitectura de Computadores II
2ª Frequência

Departamento de Informática
Universidade de Évora

24 de Novembro de 2021

Indique todos os cálculos efectuados e todas as fórmulas usadas

Perguntas rápidas

1. [1 valor] A execução *pipelined* é uma técnica que aumenta o número de instruções que o processador consegue executar por unidade de tempo ou que diminui o tempo que uma instrução demora a executar?

2. [1 valor] Se a instrução que está no andar MEM do *pipeline* RISC-V gera uma excepção, o que pode acontecer a essa instrução quando a execução do programa é retomada? Ser reexecutada desde o início, ser reexecutada a partir do andar MEM, ou não ser executada, por ser considerada já executada?

3. [1 valor] Numa cache 16-*way set associative* com 1024 conjuntos, em quantas posições pode ser colocado um dado bloco?

4. [1 valor] Mencione uma técnica usada para diminuir o número de *compulsory misses* numa cache.

Pipeline RISC-V de 5 andares

Use como referência o *pipeline* da Figura 1. Tenha, no entanto, em atenção, as caracterizações do funcionamento do *pipeline* feitas nas várias perguntas.

5. [4 valores] Simule a execução do código apresentado abaixo, no *pipeline* RISC-V com *forwarding*, com decisão dos saltos condicionais no andar EX e com previsão perfeita do resultado das instruções de salto condicional.

```
1.   ciclo:  lw   x5, 0(x10)
2.           beq  x5, x0, fim
3.           lw   x6, 0(x11)
4.           add  x7, x5, x6
5.           sw   x7, 0(x10)
6.           sub  x7, x5, x6
7.           sw   x7, 0(x11)
8.           addi x11, x11, 4
9.           addi x10, x10, 4
10.          beq  x0, x0, ciclo
11.  fim:
```

Apresente a evolução do estado do *pipeline* durante a execução, indicando todos os atrasos introduzidos e todos os pontos onde foi necessário o *forwarding* de algum valor, identificando claramente entre que andares o *forwarding* foi feito.

A simulação deve ter início na primeira instrução e terminar com a execução da instrução da linha 2 pela segunda vez.

(CONTINUA...)

6. [1,5 valores] Considere que o código abaixo é executado no *pipeline* de 5 andares, com decisão dos saltos condicionais no andar ID, e que o processador prevê, incorrectamente, que o salto condicional da linha 3 não será efectuado e que a instrução a executar a seguir é a da linha 4.

```
1.            add x1, x2, x3
2.            sub x4, x5, x6
3.            beq x0, x0, salto
4.            or  x7, x8, x9
5.            and x10, x11, x12
6.    salto:  lw  x13, 0(x14)
```

Apresente o conteúdo do *pipeline* no ciclo de relógio em que a instrução `beq` está no andar EX, que é o que se segue àquele em que o processador descobre que o salto é efectuado. (Basta indicar que instrução está em cada um dos andares do *pipeline*.)

ILP

7. [3,5 valores] Organize o código original da pergunta 5, introduzindo as alterações que considerar convenientes, para ser executado no *pipeline* RISC-V com *double issue* (com *forwarding*, decisão dos saltos condicionais no andar ID e previsão perfeita), em que cada *issue packet* pode conter uma instrução aritmética ou de salto, e uma instrução de acesso à memória, de modo a não haver a necessidade da introdução de atrasos durante a sua execução. (Se precisar de algum registo adicional, pode usar os registos `x28` a `x31`.)

Qual o CPI correspondente ao código que escreveu? E qual o IPC?

8. [1,5 valores] Considere o código RISC-V abaixo. Para cada uma das instruções, a partir da linha 2, diga, justificando, se essa instrução poderia ser executada antes da instrução da linha 1, num processador com execução fora de ordem.

```
1.    and x5, x6, x7
2.    or  x7, x8, x9
3.    add x10, x5, x11
4.    sw  x12, 0(x13)
5.    lw  x5, 0(x12)
```

Cache

9. Considere uma cache 2-*way set associative*, com 4 conjuntos, blocos de 4 palavras e substituição por LRU, num sistema em que um endereço e uma palavra têm 32 bits e em que as posições de memória com os endereços seguintes são acedidas pela ordem indicada:

```
44 108 76 111 140 172 47
```

(a) [3 valores] Assuma que a cache inicialmente está vazia e, para cada acesso, indique a palavra acedida, o número do bloco a que pertence a palavra, o índice da posição da cache que irá ocupar, o *tag*, se há um *hit* ou um *miss* e, quando aplicável, o número do bloco que será substituído.

Apresente o conteúdo final da cache, tão completo quanto possível.

(b) [0,5 valores] Calcule a *miss rate* verificada na alínea anterior.

(c) [2 valores] Calcule o número de bits do *tag*.

Nome: _____ Número: _____

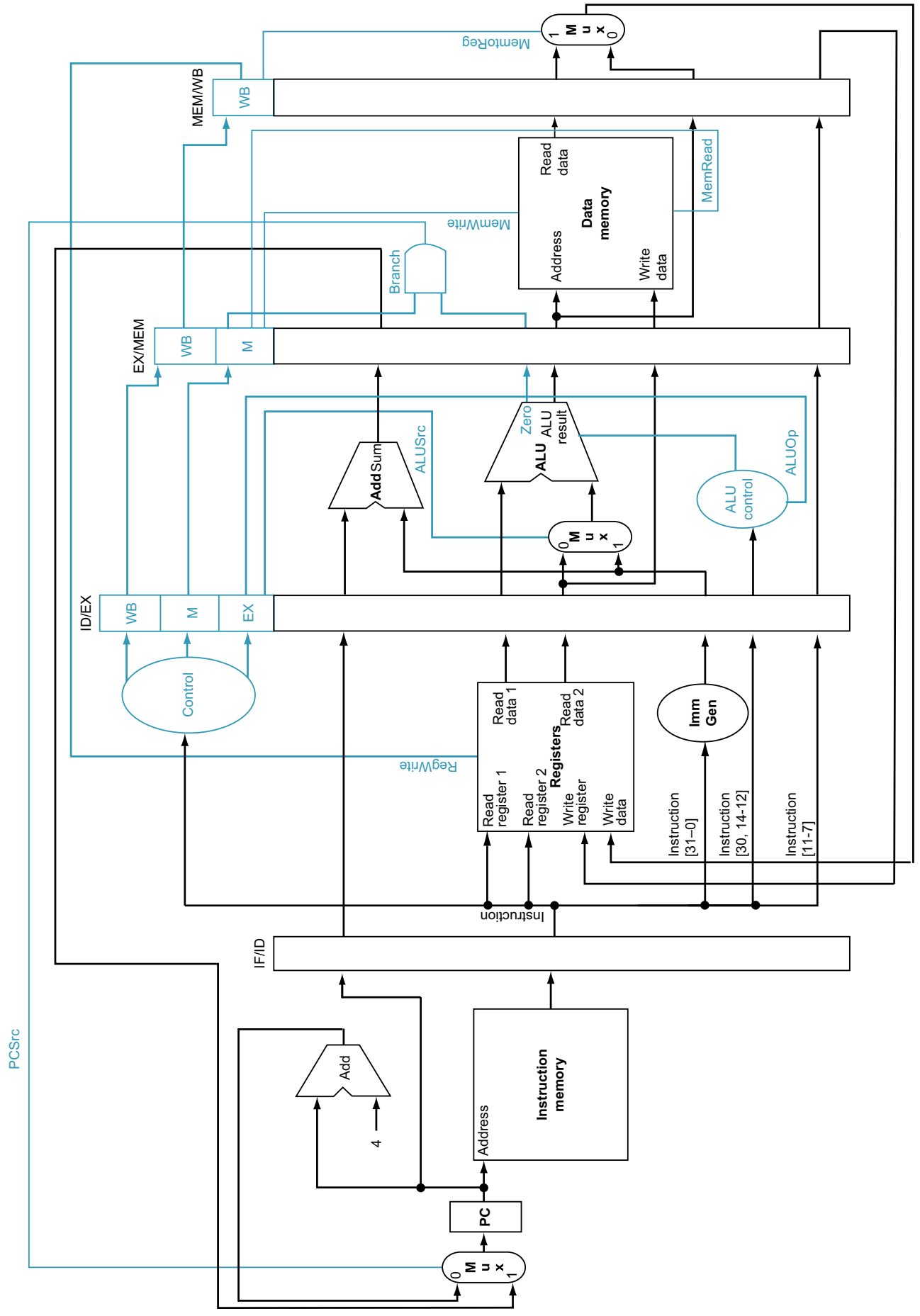

Figura 1: Diagrama de blocos do *pipeline* RISC-V