Arquitectura de Sistemas e Computadores II
2ª Frequência

Departamento de Informática
Universidade de Évora

23 de Novembro de 2017

Indique todos os cálculos efectuados

Perguntas rápidas

1. [1 valor] A execução *pipelined* é uma técnica que aumenta o *throughput* do processador ou que diminui o tempo que uma instrução demora a executar?

2. [1 valor] Quantos ciclos de relógio demora a execução da instrução `sw` no *pipeline* MIPS de cinco andares, na ausência de atrasos devido a acessos à memória?

3. [1 valor] Se a instrução que está no andar MEM do *pipeline* MIPS gera uma excepção, o que pode acontecer a essa instrução quando o tratamento da excepção termina? Ser reexecutada desde o início, ser reexecutada a partir do andar MEM, ou não voltar a ser executada, devido ao programa ser abortado?

4. [1 valor] Quantas tabelas de páginas existem, num sistema com memória virtual, se, além do sistema operativo, estiverem em execução, em simultâneo, 5 programas?

Pipeline MIPS de 5 andares

Para este grupo, use como referência o *pipeline* da Figura 1. Tenha, no entanto, em atenção as caracterizações do funcionamento do *pipeline* feitas nas várias alíneas.

5. Considere que o significado e o efeito do código MIPS seguinte são exactamente aqueles que teria se fosse executado na implementação monociclo do processador (onde não existem *delay slots*). No fim da execução do código, os valores presentes nos registos usados não são importantes.

```
1.   ciclo:  slt   $t0, $a0, $a1
2.           beq   $t0, $0, fim
3.           lw    $t1, 0($a0)
4.           lw    $t2, 0($a1)
5.           sw    $t2, 0($a0)
6.           sw    $t1, 0($a1)
7.           addiu $a0, $a0, 4
8.           addiu $a1, $a1, -4
9.           beq   $0, $0, ciclo
10.  fim:    jr    $ra
```

(a) [2,5 valores] Identifique todas as dependências (de dados) existentes no código apresentado.

(b) [3 valores] Simule a execução do código apresentado num processador com *forwarding*, com decisão dos saltos condicionais no andar ID, com previsão perfeita do resultado das instruções de salto condicional e sem *delay slots*, com o salto correspondente à instrução 2 a ser efectuado na segunda vez que a instrução for executada. 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.

(CONTINUA. . .)

(c) [2,5 valores] Altere o código apresentado, reordenando as instruções e, se considerar útil, modificando o *offset* das instruções de acesso à memória, de modo a eliminar o maior número possível de atrasos e de ciclos desperdiçados durante a sua execução no *pipeline* com *forwarding*, com decisão dos saltos condicionais no andar ID e com um *branch delay slot*.

ILP

6. [2,5 valores] Organize o código original da pergunta 5, introduzindo as alterações que considerar convenientes, para ser executado no *pipeline* MIPS *double issue* (com *forwarding*, decisão dos saltos condicionais no andar ID, previsão perfeita e sem *delay slot*), 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.

Caches

7. Considere uma cache 2-*way set associative*, com dois conjuntos, blocos de 2 palavras, palavras de 64 bits, e aplicando a estratégia LRU para a escolha do bloco a substituir. O conteúdo dessa cache é parcialmente apresentado na figura seguinte (onde M[p] representa o valor da palavra de memória p):

Índice	*Valid*	*Tag*	Palavras		*Valid*	*Tag*	Palavras	
0	1			M[21]	1	2	M[8]	
1	1	3			0			

(a) [2,5 valores] Complete o que puder da figura.

(b) [3 valores] Simule funcionamento da cache, a partir do estado apresentado acima, para a sequência de acessos aos endereços seguintes:

```
74 170 160 80 100
```

Assuma que o último bloco da cache acedido foi o que se encontra na posição da direita, no conjunto com índice 0. 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, e calcule a *miss rate* verificada.

Nome: _____ Número: _____

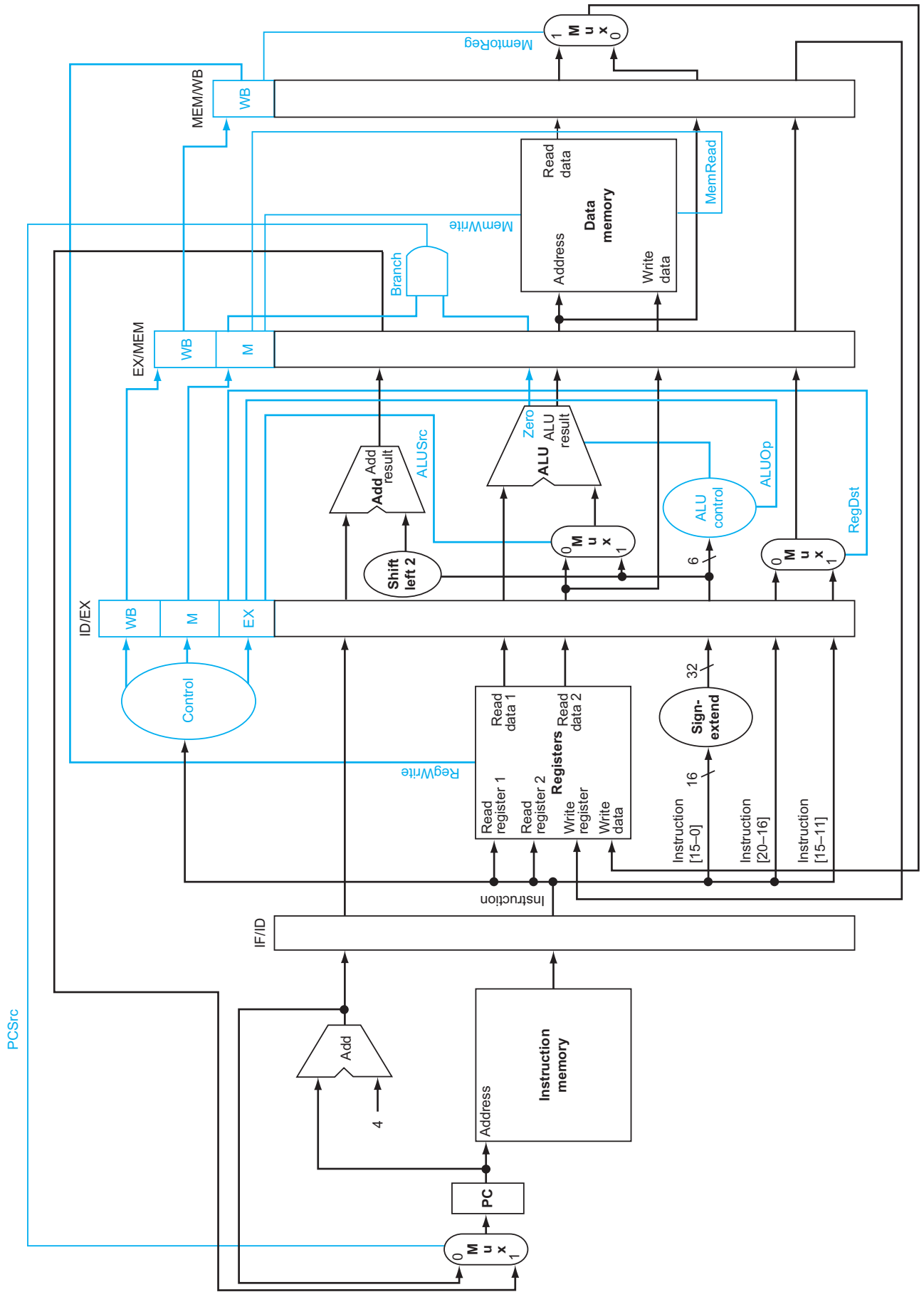

Figura 1: Diagrama de blocos do *pipeline* MIPS