Arquiteturas de Sistemas Críticos de Computadores
Departamento de Engenharia Informática - Instituto Superior de Engenharia do Porto

Exame Época Recurso - Duração: 60 minutos

Notas:

- Cada **três** respostas erradas anulam uma resposta correta
- Nota mínima de 7.5 valores

Nº: _____ Nome:_____

1) O código *assembly* de um programa compilado para uma arquitectura RISC, comparado com o mesmo programa compilado para uma arquitectura CISC, caracteriza-se geralmente por:
(a) As instruções endereçarem um elevado número de operandos em memória
(b) O número de instruções que compõem o programa ser muito menor
(c) Serem necessárias mais instruções para realizar operações complexas

2) O desempenho do CPU pode ser medido usando a fórmula T_{exec} = #I * CPI * T_{cc}. Logo:
(a) O tempo de execução (T_{exec}) de qualquer programa diminui na mesma proporção em que aumenta a frequência do relógio do processador (T_{cc})
(b) Um algoritmo com menor número de instruções (#I) do que outro funcionalmente equivalente, resulta invariavelmente num programa mais rápido
(c) Para o mesmo processador obtem-se um programa mais rápido usando um algoritmo e um compilador que reduza o número de instruções (#I) e que minimize o tempo médio de todas as instruções (CPI)

3) Em MIPS, o controlo do fluxo de informação exercido com sinais de controlo:
(a) Não depende da instrução a ser executada
(b) Activa apenas uma parte do processador
(c) Determina o ritmo a que a informação avança no processador

4) Em MIPS, num CPU de ciclo único sem *pipeline*, podemos afirmar que:
(a) Esta solução é promissora do ponto de vista do desempenho, pois todas as instruções demoram apenas um ciclo de relógio a completar (CPI=1)
(b) Um dos grandes inconvenientes desta abordagem é que o ciclo de relógio deve ser tão longo como a instrução mais demorada
(c) Esta solução obriga ao desenho de um controlo de fluxo e respectiva unidade de controlo extremamente complexos

5) Em MIPS, os campos de codificação em binário de uma instrução do tipo I designam-se por:
(a) *opcode* | *rs* | *rt* | *offset/immediate*
(b) *opcode* | *target address*
(c) *opcode* | *rs* | *rt* | *rd* | *shamt* | *funct*

6) Em MIPS, a stack é gerida de acordo com os seguintes princípios:
(a) cresce no sentido dos endereços mais altos, apontando o registo `$sp` para a última posição ocupada
(b) cresce no sentido dos endereços mais baixos, apontando o registo `$sp` para a primeira posição livre
(c) cresce no sentido dos endereços mais baixos, apontando o registo `$sp` para a última posição ocupada

7) Em MIPS, num CPU com *pipeline*:
(a) O ciclo de relógio é determinado pela fase mais lenta de uma instrução
(b) A técnica de *forwarding* resolve a latência do acesso à memória
(c) A performance global é melhorada diminuindo o tempo de execução das instruções individuais

8) Em MIPS, num CPU com *pipeline*, a técnica de *forwarding* permite:
(a) Utilizar como operando de uma instrução um resultado produzido por outra instrução que se encontra numa etapa mais recuada do *pipeline*
(b) Utilizar como operando de uma instrução um resultado produzido por outra instrução que se encontra numa etapa mais avançada do pipeline
(c) Escrever o resultado de uma instrução num registo da arquitetura antes de ela chegar à fase WB

9) Em MIPS, num CPU com *pipeline* simples, na terceira fase de execução de uma instrução de salto condicional ("beq/bne"), as operações realizadas são:
(a) Calcular o valor do *Branch Target Address* e comparar os registos (operandos da instrução)
(b) Calcular o valor de PC+4 e comparar os registos (operandos da instrução)
(c) Obter o valor dos registos (operandos da instrução), comparar os seus valores e calcular o valor do *Branch Target Address*

10) Em MIPS, num CPU com *pipeline* as situações denominadas por *control hazard* ocorrem quando:
(a) Um dado recurso de *hardware* é necessário para realizar no mesmo ciclo de relógio duas ou mais operações relativas a instruções em diferentes etapas do *pipeline*
(b) É necessário iniciar a execução de uma nova instrução e existe numa etapa mais avançada do *pipeline* uma instrução que ainda não terminou e que pode alterar o fluxo de execução
(c) Existe uma dependência entre o resultado calculado por uma instrução e o operando usado por outra que segue mais atrás no *pipeline*

11) Numa cache *n-way set associative* um bloco pode ser colocado:
(a) Em qualquer um dos conjuntos da cache e dentro desse conjunto numa posição mapeada diretamente
(b) Numa qualquer entrada da cache pois esta não está dividida por conjuntos
(c) Numa qualquer entrada de um conjunto mapeado diretamente

12) Qual dos seguintes tipos de cache usa um comparador por entrada para pesquisar em paralelo todas entradas?
(a) *Fully associative*
(b) *Direct mapped*
(c) *N-way set associative*

13) Numa cache com um elevado grau de associatividade, a política de substituição de um bloco normalmente usada é:
(a) Least Recently Used (LRU)
(b) Escolha aleatória do bloco a substituir
(c) A seleção do bloco a substituir é determinada diretamente pelo seu endereço

14) No mecanismo de memória virtual, a escrita de dados na memória RAM:
(a) Segue uma política *write-through* para fazer face à enorme latência do acesso à memória
(b) Segue uma política *write-back* para fazer face à enorme latência do acesso à memória
(c) É *direct mapped* para minimizar o número de colisões

15) A tabela de páginas usada no mecanismo de memória virtual
(a) É necessária porque a colocação das páginas em memória é *fully associative*
(b) É necessária porque a colocação das páginas em memória é *direct mapped*
(c) Tem um tamanho independente do tamanho da página

16) O desenho da memória em *interleaving*
(a) Permite aceder a *n* palavras em simultâneo num único acesso à memória
(b) Permite aceder a *n* palavras na memória com uma abordagem similar ao *pipelining* na execução das instruções
(c) Nenhuma das anteriores

17) A sequência "TLB miss -> Page Table miss -> Cache hit" é
(a) Possível, TLB falha, a página não está em memória, mas o bloco está na cache
(b) Impossível, porque o bloco não pode estar na cache se a página correspondente não estiver em memória
(c) Nenhuma das anteriores

18) A velocidade da memória afeta a decisão sobre qual o tamanho dos blocos na cache. Qual dos seguintes princípios de design pode ser considerado válido?
(a) Quanto menor for a latência da memória, maior o tamanho do bloco
(b) Quanto maior for a largura de banda da memória, menor é o tamanho do bloco
(c) Quanto maior for a largura de banda da memória, maior é o tamanho do bloco

19) Aumentar a associatividade de uma cache
(a) Diminui o *miss rate* de forma linear à medida que aumentamos o grau de associatividade
(b) Diminui o *miss rate* mas o ganho é cada vez menor à medida que aumentamos o grau de associatividade
(c) Diminui custo da pesquisa à medida que aumentamos o grau de associatividade

20) Numa hierarquia de memória com múltiplos níveis de cache podemos afirmar que
(a) As caches L1 têm como principal preocupação o *hit time* e as caches L2 o *miss penalty*
(b) As caches L1 têm como principal preocupação o *miss penalty* e as caches L2 o *hit time*
(c) Nenhuma das anteriores