Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática
Arquitectura de Computadores 2 – Ano Letivo 2012/13 – Exame de Recurso

AC2, RECURSO
5/07/2013
N. Mec:_____

Nome:_____

Notas Importantes!

1. Pode responder a cada questão com uma opção que deverá assinalar com um X na tabela ao lado. Por cada resposta errada será descontado, à cotação global, 1/3 da cotação da respectiva pergunta.
2. Durante a realização do teste não é permitida a permanência na sala de calculadoras, telemóveis ou outros dispositivos electrónicos.
3. Cotações: Grupo I – cada 0.4 valores Grupos II e III – cada 0.6 valores.

	a	b	c	d
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
	a	b	c	d
21				
22				
23				
24				
25				
26				
27				
28				
29				
30				
31				
32				
33				
34				
35				
36				
37				
	a	b	c	d
38				
39				
40				

Grupo I

1. Um microcontrolador é um dispositivo programável que tipicamente:

a) disponibiliza através dos portos de I/O a generalidade dos sinais dos barramentos do microprocessador para ligação direta a sensores e atuadores de um sistema embutido

b) integra num único circuito integrado, microprocessador, memória e periféricos

c) devido a restrições de custos não utiliza mecanismos de multiplexagem para partilha de pinos entre diversas funcionalidades

d) todas as restantes respostas estão corretas

2. O modelo de programação de um periférico especifica:

a) o sub-conjunto de instruções *assembly* do CPU suportadas por esse periférico

b) os sinais elétricos usados na ligação do periférico a dispositivos externos, tais como sensores e atuadores

c) o conjunto de registos, respetivos campos e modos de acesso suportados pelo periférico

d) nenhuma das restantes respostas está correta

3. A descodificação de endereços consiste em:

a) determinar, em função do endereço presente no barramento, qual o periférico ou memória que deve ser selecionada

b) representar um endereço em binário de forma a utilizar o menor número possível de linhas do barramento

c) determinar em função do endereço gerado pelo periférico, qual o CPU ou memória que deve ser selecionada

d) preencher a totalidade do espaço de endereçamento do processador com memórias e periféricos

4. Numa transferência síncrona:

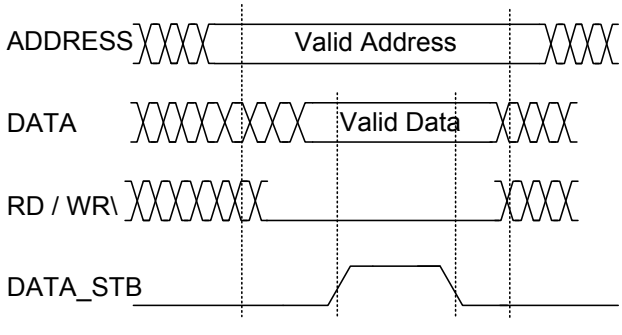
a) o CPU prolonga o ciclo de leitura/escrita por um ou mais ciclos de relógio, se for ativado um sinal de protocolo gerado pelo dispositivo externo

b) assume-se que o dispositivo externo responde à velocidade do CPU e, consequentemente, não existem sinais de protocolo envolvidos no *handshake* da transação

c) o CPU prolonga o ciclo de leitura/escrita até que o dispositivo externo sinalize através de sinais de protocolo que a operação pretendida foi completada

d) nenhuma das restantes respostas está correta

5. A figura ao lado corresponde ao diagrama temporal de uma transferência:



a) assíncrona de escrita, com dados e endereços disponibilizados numa configuração "*merged*"

b) assíncrona de escrita, com dados e endereços disponibilizados numa configuração "micro-ciclo"

c) síncrona de escrita, com dados e endereços disponibilizados numa configuração "micro-ciclo"

d) síncrona de escrita, com dados e endereços disponibilizados numa configuração "*merged*"

6. Para transferir eficientemente informação entre o CPU e um controlador de disco deve-se usar a técnica de transferência de informação baseada:

a) em interrupções não vectorizadas
b) em interrupções vectorizadas
c) num controlador de acesso direto à memória
d) em ciclos de *polling*

7. Numa transferência por DMA, a funcionar no modo *cycle-stealing*, o respetivo controlador:

a) pode usar o barramento em qualquer instante, de acordo com as suas necessidades
b) apenas pode usar o barramento quando o CPU não estiver a aceder à memória ou a unidades de I/O
c) pode usar o barramento sempre que o árbitro lhe dê permissão para o fazer, desde que respeite um tempo mínimo entre utilizações do barramento
d) pode usar o barramento em qualquer instante, de acordo com as suas necessidades, desde que respeite um tempo mínimo entre utilizações do barramento

8. Um *watchdog timer* serve para:

a) interromper o processador quando o valor final da contagem for atingido
b) fazer o *reset* ao processador quando o valor final da contagem for atingido
c) monitorizar o programa de está em execução, gerando uma interrupção quando este tente usar os *timers* de modo incorreto
d) gerar um sinal de PWM

9. Num protocolo série a técnica de relógio implícito usa-se quando se pretende um barramento:

a) com o menor número de linhas possível e uma velocidade de transmissão de informação baixa
b) com o menor número de linhas possível e uma velocidade de transmissão de informação alta
c) capaz de enviar muita informação numa trama sem necessidade de sincronização intermédia ao longo da mesma
d) em que, o transmissor e o recetor adaptam mutuamente o seu ritmo de transmissão/receção

10. Um árbitro de um barramento *multimaster* baseado em prioridades FIFO garante:

a) que é sempre servido o *master* de maior prioridade com pedido pendente de atribuição de barramento
b) a ausência de fenómenos de *starvation*
c) que a atribuição do barramento é fixada pela ordem temporal inversa com que os *masters* fazem os seus pedidos
d) que a atribuição do barramento é fixada pela ordem temporal com que os *slaves* fazem os seus pedidos

11. Num porto de saída constituído por *flip-flops* tipo D *positive edge triggered*, o sinal de *clock enable* ativo alto (**`ClkEnable`**) é obtido a partir dos sinais **`Sel\`** e **`WR\`** (ambos ativos baixos) de acordo com a expressão lógica:

a) **`ClkEnable = (Sel\ . WR\)\`**
b) **`ClkEnable = Sel\ . WR\`**
c) **`ClkEnable = (Sel\ + WR\)\`**
d) **`ClkEnable = Sel\ + WR\`**

12. No protocolo SPI, o *master* seleciona o *slave* com quem vai comunicar através de:

a) um sinal de seleção que ativa antes de iniciar a transferência
b) informação transmitida na linha de dados
c) um sinal específico de selecção através do qual é transferido o endereço desse *slave*
d) um barramento de endereços de 7 bits a partir do qual cada dispositivo descodifica o seu próprio endereço

13. Num *device driver* para uma UART (porta série RS232) utilizam-se tipicamente:

a) interrupções para sinalizar diretamente a aplicação de alto nível de que foi recebido um novo caracter ou que a UART está disponível para o envio de um novo caracter

b) estruturas de dados do tipo FIFO como meio de comunicação com a aplicação de alto nível, sendo o envio e a recepção de caracteres da UART processados por interrupção

c) interrupções geradas pela aplicação de alto nível para proceder ao envio de novos caracteres de/para os *buffers* circulares

d) *buffers* circulares como meio de comunicação com a aplicação de alto nível, sendo o envio e a recepção de caracteres da UART processados por *polling*

14. O protocolo de comunicação série CAN:

a) suporta transferências de informação assíncronas, usando a técnica de *clock stretching*

b) é robusto a interferências eletromagnéticas

c) utiliza mecanismos de CRC para deteção de erros de transmissão

d) todas as restantes respostas estão corretas

15. Suponha que dispõe de 64 circuitos de memória de 4Mx2. Usando todos estes circuitos é possível construir um módulo de memória de:

a) 4M x 64

b) 8M x 32

c) 16M x 8

d) 32M x 16

16. O número de transístores tipicamente usados para armazenar um *bit* de informação, respetivamente para células de memórias estática e células de memória dinâmica, são:

a) 1 e 2

b) 1 e 6

c) 6 e 1

d) 2 e 1

17. O *dirty bit* é usado numa *cache* com política de escrita:

a) *write-through* para indicar que a informação armazenada no respetivo bloco foi alterada

b) *write-through* para indicar que o respetivo bloco não está a ser usado

c) *write-back* para indicar que a informação armazenada no respetivo bloco foi alterada

d) *write-back* para indicar que o respetivo bloco não está a ser usado

18. Numa *cache* com associatividade de 8 de 8 kByte e 128 blocos, o número de comparadores necessários para verificar se existe um *cache hit* é:

a) 8

b) 64

c) 8192

d) nenhuma das restantes respostas está correta

19. A técnica de memória virtual permite:

a) a utilização de armazenamento secundário para aumentar a dimensão aparente da memória física do sistema

b) que o espaço de endereçamento de um processo exceda o limite de memória física disponível

c) implementar mecanismos de proteção devido à independência dos espaços de endereçamento de cada processo

d) todas as restantes respostas estão corretas

20. A tradução de endereços virtuais em endereços físicos consiste na tradução do:

a) *virtual page number* no *physical page number* e sua justaposição com o *page offset* do endereço virtual

b) *physical page number* no *virtual page number* e sua justaposição com o *page offset* do endereço físico

c) *physical page offset* no *virtual page offset* e sua justaposição com o *page number* do endereço físico

d) *virtual page offset* no *physical page offset* e sua justaposição com o *page number* do endereço virtual

Grupo II

21. O sinal de seleção (CE – ativo alto) de uma memória de 4 kByte mapeada nos endereços **`0x4XXX`** e **`0xCXXX`** de um processador com um espaço de endereçamento de 16 bits pode ser obtido através da expressão:

a) **`CE = A14\ . A13  . A12`**

b) **`CE = A14  . A13\ . A12\`**

c) **`CE = A14\ + A13  + A12`**

d) **`CE = A14  + A13\ + A12\`**

22. Três *masters* I^2C iniciam uma tentativa de comunicação ao mesmo tempo. Os dois primeiros pretendem comunicar com um *slave* com o endereço 0x13 e o terceiro com um *slave* com o endereço 0x24. Podemos dizer que quem ganha o processo de arbitragem de acesso ao barramento I^2C é:

a) o terceiro *master*, porque quer comunicar com o *slave* com o endereço mais alto

b) o primeiro ou o segundo *master*, escolhido aleatoriamente, porque querem ambos comunicar com o *slave* com o endereço mais baixo

c) um dos três *masters*, escolhido aleatoriamente, porque todos dão início à comunicação ao mesmo tempo

d) impossível de determinar com os dados fornecidos

23. Num sistema de interrupções com uma única linha e identificação da fonte por *software*, a sequência de operações efetuada durante o atendimento a uma interrupção é, pela ordem indicada, a seguinte:

a) identificação da fonte, determinação do endereço da RSI, salvaguarda do endereço de retorno, salto para a RSI

b) salto para a RSI, identificação da fonte, salvaguarda do endereço de retorno

c) determinação do endereço da RSI, identificação da fonte, salvaguarda do endereço de retorno, salto para a RSI

d) salvaguarda do endereço de retorno, salto para a RSI, identificação da fonte

24. Na organização do sistema de interrupções designada por "interrupções vetorizadas", o processador obtém o vetor que identifica o periférico gerador da interrupção:

a) por *hardware* num ciclo de *interrupt acknowledge* durante o qual o periférico gerador da interrupção o coloca no barramento de dados

b) por *hardware* através da leitura do valor presente no barramento de endereços uma vez que quando o periférico ativa a linha de interrupção coloca simultaneamente nesse barramento o seu vetor

c) por *software* na rotina de serviço à interrupção "questionando" cada um dos periféricos do sistema

d) por *software*, antes de chamar a rotina de serviço à interrupção "questionando" cada um dos periféricos do sistema

25. Considere um controlador de DMA de 32 bits, dedicado, a funcionar em modo *cycle stealing* em que cada *bus cycle* demora apenas 1 ciclo de relógio. Considere ainda que o intervalo de tempo mínimo entre operações elementares é também de 1 ciclo de relógio. Para a transferência de 4096 *bytes* este controlador de DMA demora, no mínimo:

a) 2047 ciclos de relógio

b) 4095 ciclos de relógio

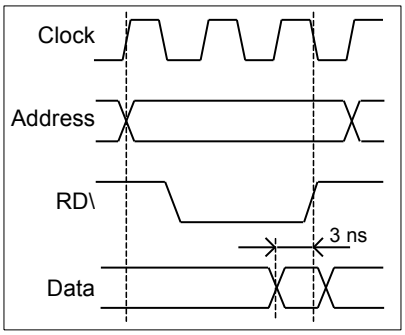
c) 8191 ciclos de relógio

d) 16383 ciclos de relógio

26. Considere um *timer* de 8 bits, com *reset* síncrono e *prescaler* (divisor de frequência) do seu sinal de relógio de entrada, que funciona, em modo alternado, com duas constantes de divisão KA e KB. Utilizando o *timer* como gerador de um sinal de PWM, e supondo que o tempo a "1" do sinal é determinado pela constante KA e que a frequência de entrada (antes do *prescaler*) é 65536 Hz, para se obter à saída um sinal com um período de 1s e um *duty-cycle* de 25%, o valor do *prescaler* e da constante KB deverão valer, respetivamente:

a) 512 e 32
b) 512 e 96
c) 256 e 63
d) 256 e 191

27. Considere um CPU a funcionar a uma frequência de 100 MHz ligado a uma memória com um tempo de acesso (referenciado ao seu sinal CE\) de 35 ns. O CPU suporta transferências de tipo semi-síncrono, estando o ciclo de leitura, sem *wait-states*, representado na figura ao lado (note o tempo de *setup* de 3ns). No barramento de dados que interliga o CPU e a memória existe um *buffer* com um tempo de propagação de 3ns e o descodificador que gera o sinal de seleção para a memória apresenta um atraso de propagação de 5 ns. Para que este sistema funcione corretamente o número de *wait-states* que é necessário introduzir no ciclo de leitura é:



a) 0
b) 1
c) 2
d) nenhuma das restantes respostas está correta

28. Sabe-se que uma memória dinâmica de 1Mx32 *bits* precisa de ser refrescada 100 vezes por segundo. Sabe-se também que tempo de ciclo (t_{RC}) dessa memória é de 100ns. Nestas condições, a percentagem de tempo que a memória está disponível para operações de leitura e de escrita é, aproximadamente, de:

a) 1%
b) 10%
c) 99%
d) 99.9%

29. O trecho de código *assembly* MIPS (com *branch delay slot* e unidades de *forwarding* para evitar *hazards* de dados) que se apresenta ao lado envia 1024 *words* para um periférico. Admitindo que este código é executado num processador de 250 MIPS e que o ciclo de *polling* é efetuado em média 4 vezes, a taxa de transferência média que se obtém é, aproximadamente:

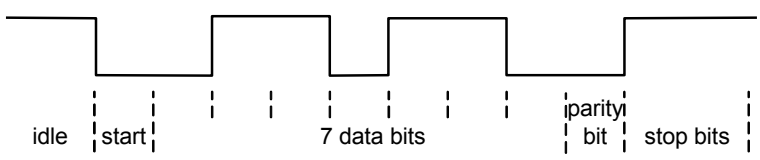
a) 10 Mbytes/s
b) 20 Mbytes/s
c) 40 Mbytes/s
d) 80 Mbytes/s

```
send: la    $a0,mem_buf
      la    $a1,io_addr
      ori   $a2,$0,1024
poll: lw    $t0,0($a1)
      ori   $t1,$0,0x1000
      and   $t0,$t0,$t1
      beq   $t0,$zero,poll
      nop
      lw    $t0,0($a0)
      addi  $a2,$a2,-1
      sw    $t0,4($a1)
      bne   $a2,$zero,poll
      addiu $a0,$a0,4
      jr    $ra
      ori   $v0,$zero,0
```

30. Ainda em relação ao código *assembly* MIPS ao lado, o *bit* do registo de *status* que indica, quando a 1, que o periférico está disponível para receber mais dados é o *bit* número:

a) 4
b) 12
c) 24
d) 28

31. Um dispositivo com interface RS232 e configurado para transmitir com 7 *bits* de dados, paridade par e 2 *stop bits*, produz a trama seguinte que é recebida por outro dispositivo RS232 incorretamente configurado para 8 *bits* de dados, paridade ímpar e 2 *stop bit,* e com o dobro do *baud rate*. Nestas circunstâncias o recetor, para a primeira trama que vai receber:



a) vai detetar uma trama inválida devido a um número incorreto de *stop bits*
b) vai detetar um erro de paridade
c) não vai detetar qualquer erro
d) não vai detetar o *start bit* devido à diferença de *baud rates*

32. O número mínimo de bits de controlo (excluindo os de dados) necessários para a implementação de uma memória *cache* com mapeamento direto, de 256 kByte de dados organizados em blocos de 16 bytes, com política de escrita do tipo *write-back*, num espaço de endereçamento de 32 bits, é:

a) 224 kbits
b) 240 kbits
c) 256 kbits
d) nenhuma das restantes respostas está correta

33. Considere um processador com um espaço de endereçamento de 64 bits e uma memória *cache* com uma associatividade de 8, de 2 MByte e blocos de 64 bytes. A dimensão, em bits, dos campos *tag*, *set* e *byte* é:

a) *Tag*: 43 *Set*: 15; *Byte*: 6
b) *Tag*: 46; *Set*: 12; *Byte*: 6
c) *Tag*: 55; *Set*: 3; *Byte*: 6
d) nenhuma das restantes respostas está correta

34. Considere um sistema de supervisão, baseado no protocolo SPI, que recolhe periodicamente informação proveniente de 10 sensores de temperatura, cada um deles com uma resolução de 8 bits (i.e. 8 bits de dados). O tempo mínimo que o *master*, a funcionar com uma frequência de relógio de 100 kHz, necessita para adquirir os valores de todos os sensores (cada um implementado num *slave* distinto) é, aproximadamente:

a) 80 μs
b) 200 μs
c) 800 μs
d) 2ms

35. Na técnica de memória virtual, o número de entradas do TLB é:

a) dependente da implementação sendo sempre muito inferior ao número de entradas da *page table*
b) igual ao número de entradas da *page table*
c) igual ao número máximo de páginas virtuais
d) igual ao número máximo de páginas virtuais de memória usadas pelo processo em execução

36. Um determinado sistema de memória consegue responder a um acesso a uma posição de memória que se encontra na *cache* em 2 ciclos de relógio e a um acesso a uma posição de memória que se encontra na memória central em 100 ciclos de relógio. Para um programa que tenha um *hit ratio* de 99%, o tempo médio de acesso a uma posição de memória é, aproximadamente, de:

a) 2 ciclos de relógio
b) 3 ciclos de relógio
c) 50 ciclos de relógio
d) 100 ciclos de relógio

37. O número total de pinos (excluindo os de alimentação) de uma memória estática de 32kx8, com um sinal único de leitura/escrita, é:

a) 18
b) 19
c) 20
d) 25

Grupo III

Um sistema possui um espaço de endereçamento virtual de 4 GBytes, um espaço de endereçamento físico também de 4Gbytes e páginas de memória de 4 kBytes. Considere também que:

- o processador pode fazer acessos de 1, 2 e 4 *bytes* à memória usando endereços arbitrários (isto é, não necessariamente alinhados em múltiplos de 1, 2 e 4, respetivamente)
- o PTR (*Page Table Register*) tem correntemente o valor **`0x10000000`**
- o endereço do início da *page table* de um processo em fila de espera para ser executado é **`0x11000000`**
- cada entrada da *page table* tem 32 bits e contém a seguinte informação

Valid, Read, Write, Execute, Dirty [31:27]	Bits não usados [26:20]	**PPN** [19:0]

- o conteúdo de algumas posições da memória (física) são os seguintes

Endereço	Valor
...	
0x10000400	0xD0000102
0x10000404	0x60000000
0x10000408	0xE8001000
0x1000040C	0xE0001001
0x10000410	0xA0001002
...	

Endereço	Valor
...	
0x11004000	0xD0000102
0x11004004	0x60000000
0x11004008	0xC0001105
0x1100400C	0xB0001104
0x11004010	0xA8001103
...	

38. Para o processo correntemente em execução, o endereço virtual **`0x00102400`** é traduzido no endereço físico:

a) **`0x01000400`**
b) **`0x00400400`**
c) **`0x04000400`**
d) nenhuma das restantes respostas está correta

39. Podemos afirmar que o processo correntemente em execução e o que se encontra na fila de espera para execução no CPU:

a) partilham pelo menos uma página de código
b) partilham pelo menos uma página de dados
c) têm, cada um, pelo menos 5 páginas residentes na memória física
d) todas as respostas estão corretas

40. Para o processo correntemente em execução, em qual dos seguintes casos não é gerada uma exceção?

a) *instruction fetch* usando o endereço virtual **`0x00102400`**
b) leitura de um inteiro de 4 *bytes* do endereço virtual **`0x00101FFE`**
c) escrita de um inteiro de 4 *bytes* no endereço virtual **`0x00102FFE`**
d) leitura de um inteiro de 4 *bytes* do endereço virtual **`0x00103FFE`**

(Área de Rascunho)