Exame # Arquitectura de Computadores II **26.6.2014**

Correção

Cotações: 1 a 8 – 0.5 cada; 9 – 2; 10a – 0,5 b – 0,5 c – 3; 11a – 0,5 b – 1,5 c – 3; 12a – 0,5 b – 2,5 c - 2

1. O reportório de instruções de um processador inclui as instruções de IN e OUT para entrada e saída de dados dos periféricos. Quando utilizadas diz-se que o sistema de Entradas/saídas é do tipo:
   b. Isolado

2. RS-232 é um protocolo de comunicação:
   b. Assíncrono

3. O conteúdo do Registo de Status de um módulo de entrada/saída:
   a. só pode ser lido

4. *Baud Rate* define:
   b. o número de bits transmitidos por segundo

5. Qual a velocidade de transferência do modo *high-speed* no bus USB:
   b. 480 Mbits/s

6. A arbitragem no bus I2C é
   c. Descentralizada

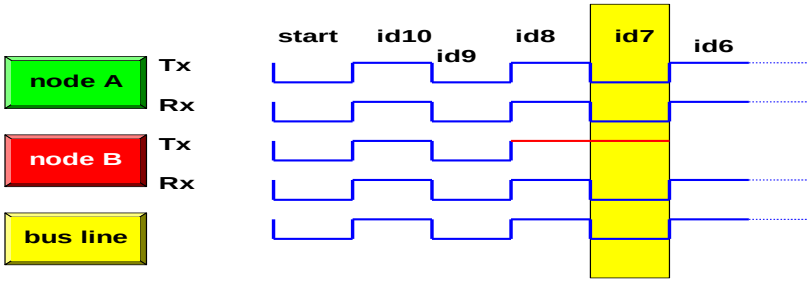
7. Para a memória cache utiliza-se
   a. SRAM

8. Numa unidade de disco o tempo de acesso a um bloco
   c. Depende da posição do bloco no disco e da posição em que se encontra a cabeça de leitura/escrita no início do acesso

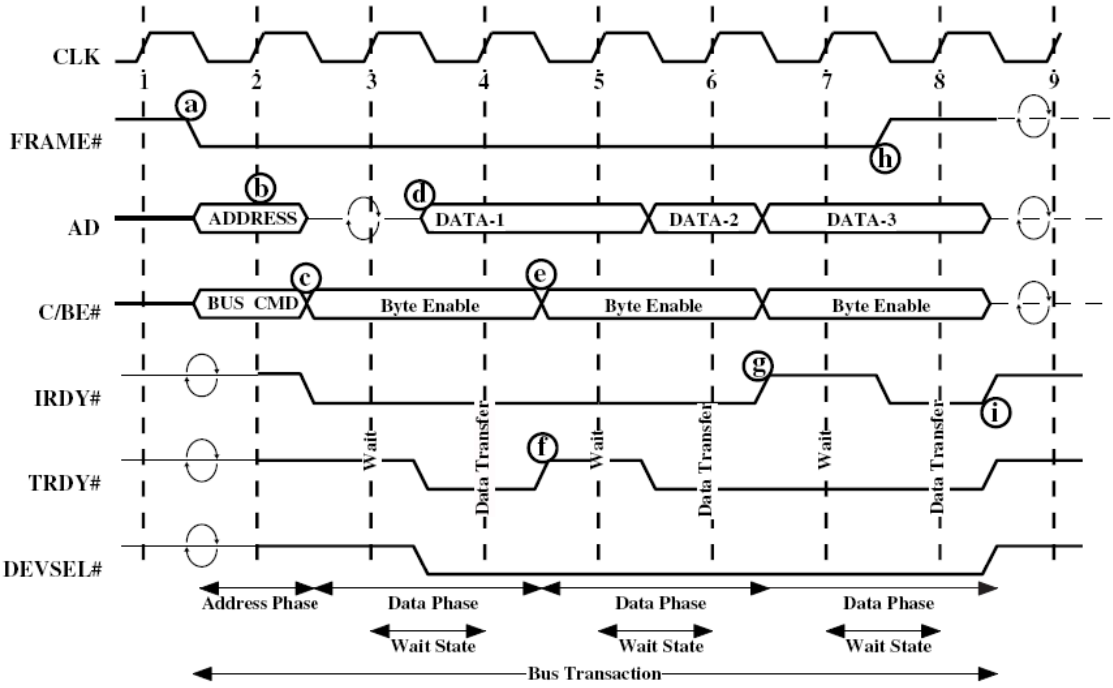
9. A arbitragem no bus CAN segue um protocol designado por *Carrier Sense Multiple Access with Collision Avoidance* (CSMD/CA). Descreva-o (Sugestão: use um diagrama temporal representando 2 nós tentando aceder simultaneamente ao bus e o resultado da arbitragem).

   No bus CAN a arbitragem é descentralizada. Sendo o bus do tipo "wired-AND", a tensão na linha corresponderá a um 0 sempre que um nó transmite um 0, independentemente do que os outros nós tentam transmitir. Assim designa-se o 0 como nível dominante e o 1 como o nível recessive.
   Quando um nó tenta transmitir no bus começa por transmitir o respetivo identificador, começando pelo seu bit mais significativo (msb first). Ao identificador está associada a prioridade de acesso, sendo esta tanto mais alta quanto menor o valor do identificador.
   Cada nó é simultaneamente transmissor e recetor, isto é, quando está a colocar informação no bus está simultaneamente a monitorar (receber) o sinal do bus.
   O mecanismo de arbitragem faz uso destas caraterísticas. Quando 2 nós tentam simultaneamente aceder ao bus o primeiro que deteta que tentou transmitir 1 (recessive) e o sinal no bus é 0 (dominant), significando que há um outro nó com prioridade mais elevada (identificador mais baixo) a transmitir, desiste de transmitir evitando a colisão.
   A figura apresenta o exemplo dos nós A e B a competir pelo bus. Os 3 bits mais significativos dos respetivos identificadores, id10 (1), id9 (0) e id8 (1) são identicos, pelo que durante esses 3 ciclos A e B se mantêm a transmitir. No entanto id7 = 0 para A e id7 = 1 para B, o que leva B a detetar discrepância entre o valor que transmitiu e o valor no bus e a cancelar a transmissão. A fica assim com o bus.

Correção



10. O bus PCI é um standard utilizado em todos os computadores pessoais.
    a. O PCI é um bus multiplexado ou desmultiplexado? Justifique a sua resposta.
    Multiplexado pois endereços e dados usam as mesmas linhas do bus
    b. O PCI é um bus síncrono ou assíncrono? Justifique a sua resposta.
    Síncrono, pois todas as transações no bus são determinadas pelas transições do sinal de clock
    c. A figura representa o diagrama temporal de uma transação no bus PCI. Descreva os eventos assinalados (a…i)



a – Master que obteve o controle do bus (***Initiator***) aciona FRAME que se mantem até à ultima fase da transferência, coloca o endereço do dispositivo nas linhas AD e indica nas linhas C/BE# que está a iniciar uma operação de ***Read*** (em concreto um "*multiple read access*")
b – os dispositivos lêem o endereço nas linhas AD
c - o ***Initiator*** retira-se do bus AD para preparar a sua utilização pelo ***Target*** (*turnaround cycle*) e ativa IRDY para indicar estar pronto a receber o dado
d - o ***Target*** ativa DEVSEL para indicar que reconheceu o seu endereço, coloca o 1º dado (DATA-1) no bus AD e ativa TRDY para o assinalar
e – o ***Initiator*** lê DATA-1no início do 4º ciclo e muda as linhas *Byte Enable* em preparação para a leitura seguinte
f – *target* not ready para enviar dado – TRDY desativado (*wait state*)
5º ciclo: *target* coloca DATA-2 em AD que é lido pelo *Initiator*
g – 6º ciclo: *target* coloca DATA-3 em AD mas initiator not ready (IRDY desativado)

Correção

h – *initiator* desativa FRAME sinalizando que a leitura é a última e ativa IRDY, lendo o 3º dado no início do 8º ciclo
i - *initiator* desativa IRDY, colocando o bus no estado inativo.

2. Pretende-se projetar um sistema de memória que permita detetar erros nos bytes armazenados.
   a. Se apenas se pretender detetar erros de um bit, que solução adotaria?
      Bit de paridade

   b. Se se pretender detetar erros num ou dois bits e corrigir os erros de 1 bit, que código de deteção e correção de erros adotaria? Quantos bits adicionais para deteção e correção de erros seriam necessários por cada byte?
      Código de Hamming. 4 bits (C8, C4, C2, C1) + 1 bit de paridade global (P):
      Posição: 12 11 10 9 8 7 6 5 4 3 2 1 0
      M8M7M6M5C8M4M3M2C4M1C2C1P
      Em que:

C1 = M1 XOR M2 XOR M4 XOR M5 XOR M7
C2 = M1 XOR M3 XOR M4 XOR M6 XOR M7
C4 = M2 XOR M3 XOR M4 XOR M8
C8 = M5 XOR M6 XOR M7 XOR M8
P = M1 XOR M2 XOR M3 XOR M4 XOR M5 XOR M6 XOR M7 XOR M8 XOR C1 XOR C2 XOR C4 XOR C8

   c. Para o byte 01001101 gere os bits de deteção e correção de erros. Mostre que o código identifica corretamente um erro no bit 5.
   b. C1 = 1 XOR 0 XOR 1 XOR 0 XOR 1 = 1
   c. C2 = 1 XOR 1 XOR 1 XOR 0 XOR 1 = 0
   d. C4 = 0 XOR 1 XOR 1 XOR 0 = 0
   e. C8 = 0 XOR 0 XOR 1 XOR 0 = 1
   f. P = 1 XOR 0 XOR 1 XOR 1 XOR 0 XOR 0 XOR 1 XOR 0 XOR 1 XOR 0 XOR 0 XOR 1 = 0

      Os 13 bits corretos seriam então:

      M8M7M6M5C8M4M3M2C4M1C2C1P
      0 1 0 0 1 1 1 0 0 1 0 1 0
um erro em M5 levaria a que a palavra armazenada fosse:
      M8M7M6M5C8M4M3M2C4M1C2C1P
      0 1 0 **1** 1 1 1 0 0 1 0 1 0
2 modos de deteção possíveis:

Modo A
C1 calculado = M1 XOR M2 XOR M4 XOR M5 XOR M7 = 0 C1 armazenado = 1
C2 calculado = M1 XOR M3 XOR M4 XOR M6 XOR M7 = 0 C2 armazenado = 0
C4 calculado = M2 XOR M3 XOR M4 XOR M8 = 0 C4 armazenado = 0
C8 calculado = M5 XOR M6 XOR M7 XOR M8 = 0 C8 armazenado = 1
P calculado = 1 P armazenado = 0 -> erro num bit

C1 calculado XOR C1 armazenado = 1
C2 calculado XOR C2 armazenado = 0 Posição do bit errado: 1001 -> M5
C4 calculado XOR C4 armazenado = 0
C8 calculado XOR C8 armazenado = 1

Correção

Modo B
M1 XOR M2 XOR M4 XOR M5 XOR M7 XOR C1 = 1
M1 XOR M3 XOR M4 XOR M6 XOR M7 XOR C2 = 0
M2 XOR M3 XOR M4 XOR M8 XOR C4 = 0 Posição do bit errado: 1001 -> M5
M5 XOR M6 XOR M7 XOR M8 XOR C8 = 1

12. Um sistema de 32-bits com a memória byte-addressble tem uma cache direct-mapped de 4kBytes com blocos (linhas) de 8 palavras de 32 bits.
    a. Quantas linhas tem a cache?
    4 kBytes = 2^{12} bytes = 2^{10} palavras de 32 bits
    2^3 palavras/linha
    no. de linhas da cache = 2^{10} / 2^3 = 2^7 = 128

    b. Quais os campos em que estão divididos os endereços da memória e qual o número de bits de cada um deles?

31 ... 12	11 ... 5	4 ... 0
Tag	Index	Byte Offset

    Byte offset: 4*8 bytes/linha = 2^5 bytes - campo de 5 bits; destes os bits 1..0 indicam a posição do byte na palavra e os bits 4..2 a posição da palavra na linha.

    Index: campo de 7 bits que indica a linha onde pode estar armazenada a palavra

    Tag: os 20 bits mais significativos do endereço que indicam o bloco de memória a que pertence o endereço, e que quando coincide com o conteúdo do campo Tag da cache indica que a linha da cache que é acedida armazena o conteúdo do bloco de memoria endereçado.

    c. O conteúdo da posição de memória 375B2A5C foi transferido para a cache. Em que linha da cache e em que posição na linha se encontra? Qual o conteúdo do campo ***Tag*** correspondente a essa linha?

    3 7 5 B 2 A 5 C
    0011 0111 0101 1011 0010 1010 0101 1100 No. linha = 1010 010 = 82
    Posição da palavra na linha = 1 11 = 7

    A pergunta podia tambem ser interpretada como pedindo:
    Posição do byte na linha = 1 1100 = 28