1. A função de um bootloader num sistema baseado num microcontrolador é:
    a. realizar a compilação do software e iniciar a sua execução após o reset do sistema
    b. transferir o código executável para o host usado no desenvolvimento para posterior assemblagem ou compilação
    c. transferir o código executável do host usado no desenvolvimento para o sistema embedded para posterior execução
    d. nenhuma das respostas está correta

Bootloader (software)

- Executam no arranque do sistema
- A comunicação com o host é efetuada por RS232 / USB
- é um programa que reside, de forma permanente, na memória do microcontrolador e que disponibiliza apenas funções básicas de transferência e execução de um programa.
- Ferramentas desenvolvidas especificamente para DETPIC32
    - bootloader - programa previamente gravado na boot flash do PIC32; lê informação do porto de comunicação e escreve na memória Flash.

2. A função de um bootloader num sistema baseado num microcontrolador é:
    a. transferir o código executável do host PC usado no desenvolvimento, para o sistema embedded para posterior execução
    b. realizar a compilação do software e iniciar a sua execução após o reset do sistema
    c. executar o software e auxiliar na sua depuração através da introdução de breakpoints, visualização do conteúdo de registos e de posições de memória
    d. interagir com o cross-compiler para efeitos de depuração da aplicação

1. Um microcontrolador é um dispositivo programável que:
    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

Microcontrolador :

- Principais características:
    - Dispositivo programável que integra, num único circuito integrado, 3 componentes fundamentais:

        - Unidade de Processamento
        - Memória(volátil e não volátil)
        - Portos de I/O
    - inclui outros periféricos(Timers, conversor A/D, serial I/O)
    - Barramentos (dados, endereços e controlo) interligam todos estes dispositivos
    - Externamente há, em geral, pinos que podem ser configurados programaticamente para diferentes funções (versatilidade)
    - Para a transferência de um programa executável para a memória do microcontrolador pode ser utilizado um dos seguintes métodos:
        - Programa-monitor (software)
        - Bootloader (software)
        - In-Circuit Debugger (hardware)
- Microcontrolador PIC32MX795F512H
    - Baseado numa arquitetura de Harvard (memória de instruções e dados separadas). Esta opção evita o hazard estrutural na implementação pipelined que aconteceria com uma única memória.

Microprocessador	**Microcontrolador**
Circuito integrado com um (ou mais) CPU	Circuito integrado inclui CPU, RAM, ROM, periféricos
Não tem memória interna (além do banco de registos)	Baixo consumo de energia
Os barramentos estão disponíveis no exterior	Utilizado em tarefas específicas (por exemplo controlo da velocidade de um motor)
Para obter um sistema completo é necessário acrescentar RAM, ROM e periféricos	Disponibiliza uma grande variedade dos periféricos e interfaces com o exterior
Pode operar a frequências elevadas (> 3GHz)	Frequência de funcionamento normalmente baixa (< 200 MHz)
Sistemas computacionais de uso geral	Rapidez de resposta a eventos externos (Sistemas de Tempo Real)

2. O sinal de seleção (CE\) de um porto mapeado na gama de endereços 0x00400...0x007FF de um processador com um espaço de endereçamento de 20 bits pode ser obtido através da expressão:

a) $CE\backslash = \sum_{i=11}^{19} Ai\backslash$
b) $CE\backslash = \sum_{i=11}^{19} Ai$
c) $CE\backslash = \sum_{i=0}^{10} Ai\backslash$
d) $CE\backslash = \sum_{i=10}^{10} Ai$

0x00400 . . . 0x007FF

0000 0000 0100 0000 0000 então $CE = \sum_{i=10}^{19} \overline{Ai} \simeq \overline{CE} = \sum_{i=11}^{19} Ai$

0000 0000 0111 1111 1111

i=10 mas só zeros

$\overline{Ai}$ → porque são zeros

3. O diagrama temporal da figura ao lado representa um ciclo de:
   a. leitura de um dispositivo em que os sinais de controlo usam lógica positiva
   b. leitura de um dispositivo em que os sinais de controlo usam lógica negativa
   c. escrita num dispositivo em que os sinais de controlo usam lógica positiva
   d. escrita num dispositivo em que os sinais de controlo usam lógica negativa

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"

Merged : apenas tem um stb
Micro-ciclo : tem dois stb1 stb2 (normalmente addr e data)
sem sinal ack é síncrono pq depende só do relógio

multiplexado porque o adress e data estão no mesmo sinal
está ativo em baixo logo escrita

6. Um árbitro de um barramento multimaster baseado em prioridade FIFO garante:
    a. a ausência de fenómenos de starvation
    b. que é sempre servido o master de maior prioridade com pedido pendente de atribuição de barramento
    c. que a atribuição do barramento é fixado pela ordem temporal inversa com que os masters fazem os seus pedidos
    d. nenhuma das respostas estão corretas

- **Prioridades dinâmicas**: As prioridades mudam ao longo do tempo, para evitar problemas tais como a impossibilidade de acesso por parte dos elementos menos prioritários (*starvation*).
  - **FIFO ou FCFS**: (***First-In-First-Out*** ou ***First-Come-First-Served***) os elementos são ordenados por ordem do pedido e o barramento é atribuído por ordem de chegada

7. Numa transferência por DMA, o respectivo 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
    d. pode usar o barramento em qualquer instante de acordo com as suas necessidades, desde que respeite um tempo mínimo entre utilizações

E/S por acesso direto à memória (DMA):

- A informação é trocada diretamente entre a memória e o periférico. O CPU não toma parte no processo de transferência
- O CPU apenas configura inicialmente o DMA e é sinalizado no final de que a transferência terminou

A transferência é exclusivamente feita por hardware pelo DMA, i.e., não é executado qualquer programa .
Quando o DMA termina a transferência de todas as palavras, gera uma interrupção.
Consiste na transferência de informação do periférico diretamente para a memória, sem utilizar o processador

Controlador de DMA

- Um DMA é um periférico que, do ponto de vista do modelo de programação, é semelhante a qualquer outro periférico
- Disponibiliza um conjunto de registos internos a que o CPU acede para configurar uma transferência de dados, nomeadamente:
  - Endereço origem da informação (endereço de leitura)
  - Endereço destino da informação (endereço de escrita)
  - Quantidade de informação a transferir (nº de bytes/words)
- Pode também ter registos com informação sobre o estado de uma transferência
- Do ponto de vista da operação realizada, o DMA é um controlador especializado na transferência de dados (cópia), de forma autónoma, em hardware, após programação feita pelo CPU
- Durante a transferência, o DMA tem a capacidade de controlar os barramentos de endereços, dados e controlo, como se fosse um CPU

- Para efetuar uma transferência de um bloco de dados de **n** palavras, o **DMA** realiza, no essencial, ao nível dos barramentos, as mesmas operações que seriam realizadas por um programa executado pelo CPU:
  - **Lê** 1 palavra (*byte* ou *word*) do dispositivo fonte (do ***source address***) para um registo interno – "**fetch**"
  - **Escreve** a palavra guardada no registo interno no passo anterior, no dispositivo destino (no ***destination address***) – "**deposit**"
  - Incrementa *source address* e *destination address*
  - Incrementa o **número de palavras transferidas**
  - Se não transferiu a totalidade do bloco, repete desde o início
- Para realizar estas tarefas o DMA necessita de **controlar os barramentos** de **endereços** e de **dados** e ainda os sinais **rd** e **wr**
- A capacidade do DMA para gerir os barramentos do sistema conflitua com capacidade idêntica do CPU

- Apenas um dos dois dispositivos, CPU e DMA, pode estar ativo no barramento de cada vez... Por defeito é o CPU
- Isto é, só pode haver, em cada instante, um "**bus master**" (só um dispositivo que é "bus master" pode controlar os barramentos)
- Quando o DMA necessita de aceder aos barramentos para realizar uma transferência, tem que primeiro ter autorização do CPU para ser o "bus master"
- O DMA só inicia a transferência de informação quando tem a confirmação, por parte do CPU, de que é "bus master"
- O controlo dos barramentos por parte do DMA é sempre temporário

- Quais são então os passos necessários à realização de uma transferência de dados por parte do DMA?

Controlador de DMA – passos para 1 transferência

- DMA pede ao CPU para ser *bus master*
- Espera até ter a confirmação do CPU
- Efetua a transferência:
  - Lê 1 palavra (*byte* ou *word*) do dispositivo fonte (do *source address*) para um registo interno
  - Escreve a palavra guardada no registo interno, no passo anterior, no dispositivo destino (no *destination address*)
  - Incrementa *source address* e *destination address*
  - Incrementa o número de palavras transferidas
  - Se não transferiu a totalidade do bloco, repete
- Retira o pedido para ser *bus master* libertando, simultaneamente, os barramentos (ou seja, as suas ligações aos barramentos de dados, de endereços e de controlo passam a funcionar como entradas)

Para se tornar "bus master" o DMA:

- Ativa o "BusReq"
- espera pela ativação do "BusGrant"(CPU ativa Bus Grant quando está em condições de libertar os barramentos)
-

8. Para a transferência de 2048 words (de 32 bits), um controlador de DMA de 32 bits, dedicado, a funcionar em modo cycle-stealing, necessita de pelo menos:
   a. 2048 bus cycles
   b. 4096 bus cycles
   c. 8192 bus cycles
   d. 16384 bus cycles

Dedicado - 1 bus cycle

Não dedicado - 2 bus cycle

- **Bloco** – o DMA assume o controlo dos barramentos até todos os dados terem sido transferidos
- **Burst** (rajada)
  - o DMA transfere até atingir o número de palavras pré-programado ou até o periférico não ter mais informação pronta para ser transferida. Se não foi transferida a totalidade da informação:
    - O periférico pode desativar o sinal Dreq o que leva o DMA a desativar o sinal BusReq e a libertar os barramentos
    - Logo que o periférico ative de novo o sinal Dreq o DMA volta a ativar o sinal BusReq e, logo que seja "bus master", continua no ponto onde interrompeu
- **Cycle Stealing**
  - O DMA assume o controlo dos barramentos durante 1 *bus cycle* e liberta-os de seguida ("rouba" 1 *bus-cycle* ao CPU) - transfere parcialmente 1 palavra (*fetch* ou *deposit*)
  - O CPU só liberta os barramentos nos ciclos em que não acede à memória (por exemplo, na fase MEM de uma instrução aritmética no MIPS, pipeline)
  - A transferência é mais lenta, mas o impacto do DMA no desempenho do CPU é nulo - o DMA aproveita os ciclos que não são, de qualquer modo, usados pelo CPU

9. Num sistema com vários dispositivos interligados por um barramento SPI, em termos de comunicação:

a. A função dos dispositivos é fixa à partida, podendo apenas existir um master, sendo os restantes obrigatoriamente slaves
b. podem existir, permanentemente, vários masters e vários slaves, configurados nessa função à partida, sendo a comunicação estabelecida entre quaisquer dois desses dispositivos
c. em cada instante apenas pode existir um master, sendo os restantes obrigatoriamente slaves, mas um slave pode tornar-se master se ganhar o processo de arbitragem com outro master de inferior prioridade
d. a função masterslave de cada dispositivo é imposta de acordo com as necessidades, pelo controlador SPI

SPI - serial peripheral interface bus
(transmissão explícita do transmissor - exemplo spi)
é usado para comunicar com uma grande variedade de dispositivos:

- sensores
- cartões de memória
- Circuitos: memórias, ADCs, DACs, Displays LCD (e.g. telemóveis), comunicação entre corpo de máquinas fotográficas e as lentes, ...
- Comunicação entre microcontroladores

Ligação a curtas distâncias
A função dos dispositivos é não é fixa à partida
Apenas pode haver um master
a função masterslave de cada dispositivo é imposta de acordo com as necessidades, pelo CPU

Descrição geral

- Arquitetura "Master-Slave" com ligação ponto a ponto
- Comunicação bidirecional "full-duplex"
- Comunicação síncrona (relógio explícito do *master*)
  - Relógio é gerado pelo *master* que o disponibiliza para todos os *slaves*
  - Não é exigida precisão ao relógio - os bits vão sendo transferidos a cada transição de relógio. Isto permite utilizar um oscilador de baixo custo no *master* (não é necessário um cristal de quartzo)
- Fácil de implementar por hardware ou por software
- Não são necessários "line drivers" (ou "transceivers") - circuitos de adaptação ao meio de transmissão

- Arquitetura "Master-Slave"
  - O sistema só pode ter um *master*
  - O *master* é o único dispositivo no sistema que pode controlar o relógio
- Um *master* pode estar ligado a vários *slaves*: para cada comunicação, apenas 1 *slave* é selecionado pelo *master*
- O *master* inicia e controla a transferência de dados
- Sinalização:
  - **SCK** – clock
    - Relógio gerado pelo *master* que sincroniza a transmissão/receção de dados
  - **MOSI** – Master Output Slave Input (SDO no *master*)
    - Linha do *master* para envio de dados para o *slave*
  - **MISO** – Master Input Slave Output (SDI no *master*)
    - Linha do *slave* para enviar dados para o *master*
  - **SS** – Slave select
    - Linha do *master* que seleciona o *slave* com quem vai comunicar

11. No barramento SPI:
    a. só é possível transmitir dados entre dois dispositivos em modo half duplex
    b. o slave e o master utilizam relógios independentes para sincronizar as transferências de informação
    c. as transferências de info são sincronizadas pelo relógio da masterd
    d. o master usa o relógio do slave para sincronizar as transferências de informação

só é possível transmitir dados entre dois dispositivos em modo full duplex
master e slave utilizam o mesmo relógio

10. Na interface I2C 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 de seleção através do qual é transferido o endereço do slave
    d. um barramento de endereços de 7 bits a partir do qual cada dispositivo decodifica o seu próprio endereço

I^2C – Caraterísticas básicas

- Transferência série bidirecional, *half-duplex*, orientada ao byte
- As transferências envolvem sempre uma relação *master/slave*
- *Master* pode ser transmissor ou recetor ("master-transmitter" ou "master-receiver")
- O barramento de comunicação necessita apenas de dois fios:
  - Serial data line (SDA)
  - Serial clock line (SCL)
- Cada dispositivo ligado ao barramento é endereçável por software usando um endereço único previamente atribuído
- Endereços de 7 bits ("standard mode") - alguns endereços reservados, 112 disponíveis
- Endereçamento de 10 bits também disponível
- Barramento *multi-master* com deteção de colisões e arbitragem, evitando a corrupção de informação se dois ou mais *masters* iniciarem simultaneamente uma transferência

I^2C – terminologia

- **Transmitter** – dispositivo que envia dados para o barramento
- **Receiver** – dispositivo que recebe dados do barramento
- **Master** – o dispositivo que inicia a transferência, gera o sinal de relógio e termina a transferência
- **Slave** – o dispositivo endereçado pelo master
- **Multi-master** – mais do que um *master* pode tentar, ao mesmo tempo, controlar o barramento sem corromper a comunicação em curso
- **Arbitragem** – procedimento para assegurar que, se mais do que um *master* tentar, simultaneamente, controlar o barramento, apenas a um é permitido continuar, sem perturbação da comunicação iniciada pelo *master* vencedor
- **Sincronização** – procedimento para sincronização dos sinais de relógio de dois ou mais dispositivos

12. Na interface I2C o endereçamento/seleção dos dispositivos é realizado através de:

a. um sinal específico ativado pelo dispositivo que comanda a transferência
b. informação transmitida na linha de relógio pelo dispositivo que comanda a transferência
c. um barramento de endereços de 7 bits a partir do qual cada dispositivo decodifica o seu próprio endereço
d. informação transmitida na linha de dados pelo dispositivo que comanda a transferência

13. A interface I2C é caracterizada fisicamente por apresentar:

a. uma linha de relógio e duas linhas de dados (uma para transmitir e outra para receber)
b. uma linha de relógio, uma linha de seleção e uma linha de dados
c. uma linha de relógio e uma linha de dados
d. uma linha de relógio, uma linha de seleção e duas linhas de dados (uma para transmitir e outra para receber)

IC2 apenas tem 2 linhas

1. O IC2 é um bus:

a. Síncrono
b. Assíncrono

11. A técnica “bit dominante/bit recessivo” é utilizada em:

a. RS232 e CAN
b. I2C e SPI
c. CAN e I2C
d. SPI e CAN

IC2 usa bit dominante e bit recessivo
CAN usa bit dominante e bit recessivo

2. Qual o tipo de memória usado nos SSD:

a. SRAM
b. Flash Memory
c. EPROM

12. Um cabo USB, de acordo com a versão 2.0 da norma, possui os seguintes condutores:

a. VBUS e GND para alimentação; D+ e D- para transmissão não-diferencial em modo full-duplex
b. VBUS e GND para alimentação; D+ e D- para transmissão diferencial em modo full-duplex
c. VBUS e GND para alimentação; D+ e D- para transmissão diferencial em modo half-duplex
d. VBUS e GND para alimentação; D+ e D- para transmissão não-diferencial em modo half-duplex

Todas as novas versões mantêm a compatibilidade com as versões anteriores, com exceção do USB 4.0 que apenas é compatível com as versões USB 3.x e USB 2.0.
Características principais:

- 3 taxas de transmissão(versão 2.0):
  - low-speed, 1.5 Mbit/s – para comunicação com dispositivos de interação com o utilizador (rato, teclado, joystick, etc.)
  - full-speed, 12 Mbit/s – para comunicação com dispositivos que não requeiram taxas elevadas (e.g. impressoras, áudio, ...)
  - high-speed, 480 Mbit/s – para comunicação com dispositivos de alto débito (e.g. vídeo, discos, ...)
- comunicação **half-duplex**(USB 1.0 e 2.0)
- USB 3.X tem as anteriores mais estas taxas de transmissão:
  - USB 3.0: Super-speed mode – 5 Gbit/s.
  - USB 3.1: Super-speed plus mode – 10 Gbit/s
- Comunicação "**full-duplex**" no USB 3.x quando são usados os "Super-speed" modes
- Os dados são transmitidos em **modo diferencial** em par entrançado (nível lógico na receção é discriminado pela diferença de tensão entre os dois sinais)
- O USB usa **comunicação síncrona orientada ao bit**
- Tipologias:
  - Ligação física em árvore
  - Máximo de 7 níveis (incluindo "root hub") e de 127 dispositivos
  - Hub (concentrador): permite que vários dispositivos possam partilhar uma única porta USB
  - Arquitetura **"master-slave"** (i.e., cada dispositivo apenas envia dados após permissão explícita do "host controller"
- "**Bit-stuffing** " quando nº de "1"s consecutivos **>= 6**
- É inserido um "0" após 6 "1"s consecutivos

14. No protocolo USB p/transferir grandes quantidades de dados com garantia de entrega são usadas:

a. transferências de controlo
b. transferência ”bulk”
c. transferências isócronas
d. transferências de interrupção

Tipo de transferências:

- USB suporta 4 tipos de transferências de dados
- **Não periódicas** – surgem sem qualquer período definido, sob iniciativa do host-controller, sem garantia de taxa de transmissão ou de latência:
  - **Control** Transfers
  - **Bulk** Transfers (massivo)
    - Transferência de volumes de dados elevados (instantes de transmissão irregulares)
    - Permitem a transmissão de dados com garantias de entrega, usando mecanismos de retransmissão em caso de deteção de uma situação de erro (CRC – Cyclic Redundancy Check)
    - Não são dadas garantias de largura de banda ou de latência, sendo este tráfego tratado em regime de "best effort"
    - discos, pen-drives, scanners, impressoras
- **Periódicas** – Na fase de enumeração é definido o período entre envios consecutivos de informação, garantindo-se um limite máximo para a latência:
  - **Isochronous** Transfers (isócrono)
    - Permitem a transferência de informação com uma **largura de banda garantida** e **latência limitada**.
    - Parâmetros são negociados após a ligação do dispositivo ao barramento, na fase de enumeração
    - Os eventuais erros de transmissão não são corrigidos, isto é, não há retransmissão de informação
    - É, das quatro, o único tipo de transferência que **não faz controlo de erros**
    - aplicações de áudio e vídeo; i.e., aplicações onde garantir o ritmo de transmissão é mais importante que garantir que todos os dados chegam ao destino
  - **Interrupt** Transfers
    - O host-controller interroga periodicamente os dispositivos para saber se algum tem informação para transferir (é um polling feito pelo hostcontroller)
    - A regularidade com que o polling é feito depende do valor da latência negociada na fase de enumeração no arranque do periférico
    - O período de polling pode ser configurado entre 1 e 255 ms

        - São transmitidas baixas quantidades de informação (até 64 bytes)
        - No caso em que ocorre um erro de comunicação, a retransmissão é feita no período de polling seguinte
        - ratos, teclados, joysticks, etc.
- Cada pipe suporta apenas um tipo
- São utilizadas pelo host-controller para o envio de mensagens de comando e configuração ou consulta de estado

15. A topologia das ligações físicas, no protocolo USB, é:
    a. um anel com um máx de 7 dispositivos
    b. uma estrela com um máx de 7 dispositivos
    c. uma árvore com um max de 127 níveis
    d. uma árvore com um máximo de 7 níveis

em árvore 7 níveis e 127 dispositivos

13. O número total de pinos (excluindo as linhas de alimentação) de um circuito integrado de uma memória dinâmica DRAM de 2Mx8, com um sinal único de controlo de leitura/escrita é:

a. 21 c. 31
b. 22 d. 32

$2M \times 8 = 2^1 \times 2^{20} \times 2^3 = 2^{21} \times 8 = 2^{2 \times 10,5} \rightarrow 11$ bits $+ 8$

$11 + 8 + 3 = \boxed{22}$

2 de escolha linha e/ou coluna

1 controlo

DRAM E SRAM - memória volátil - info armazenada perde-se quando o circuito é desligado da alimentação

SRAM :

- Vantagens:
    - Rápida
    - Informação permanece até que a alimentação seja cortada
- Desvantagens:
    - Implementações típicas: 6 transistores / célula
    - Baixa densidade, elevada dissipação de potência
    - Custo/bit elevado
- escrita : por informação em bit (1) (e bit\ (0)) ; ativar a linha select

- leitura : ativa select ; valor armazenado é detectado pela diferença de tensão entre as linhas bit e bit\

SRAM - Bloco funcional

• Diagrama lógico (interface assíncrona)

• Tabela de verdade

CE\	OE\	WE\	Operação
1	X	X	High-Z
0	1	1	High-Z
0	X	0	Escrita
0	0	1	Leitura

- Ciclo de leitura:
  - Cycle Time: tempo de acesso mais qualquer tempo adicional necessário antes que um segundo acesso possa ter início
  - Access Time: tempo necessário para os dados ficarem disponíveis no barramento de saída da memória
  - Taxa de transferência: taxa a que os dados podem ser transferidos de/para uma memória (1 / cycle_time)
- para aumentar memoria:
  - aumentar dimensão da palavra : $2^m \times n \rightarrow 2^m \times (n \times p)$ "p" circuitos
  - aumentar o número total de posições e memória: 2^k circuitos $2^m \times n \rightarrow 2^{m+k} \times n$

DRAM:

- Vantagens:
  - Implementações típicas: (1 transistor + 1 condensador) / célula
  - Alta densidade, baixa dissipação de potência
  - Custo/bit baixo
- Desvantagens:
  - Informação permanece apenas durante alguns milissegundos (necessita de refresh regular – daí a designação "dynamic")
  - Mais lenta (pelo menos 1 ordem de grandeza) que a SRAM
- leitura:
  - Pré-carregar a linha "bit" a VDD/2;
  - Ativar a linha "select";
  - Valor lógico detectado pela diferença de tensão na linha bit, relativamente a VDD/2;
  - Restauro do valor da tensão no condensador (write)
- escrita:
  - Colocar dado na linha "bit"
  - Ativar a linha "select"
- Refresh da célula: Operação interna idêntica a uma operação de "Read"
- O endereço de acesso à memória é dividido em 2 partes:
- Address:

Row Address	Column Address

- O barramento de endereços é multiplexado: primeiro é enviado o endereços de linha e depois é enviado o endereço de coluna
- A multiplexagem no tempo é feita com 2 strobes independentes •
    - RAS – Row Address Strobe
    - CAS – Column Address Strobe

- Melhorias de desempenho da DRAM
    - **Fast Page Mode :** Adiciona sinais de temporização que permitem acessos repetidos ao buffer de linha
    - **Synchronous DRAM (SDRAM)** : Adiciona um sinal de relógio à interface DRAM, para facilitar a sincronização de transferências múltiplas; Múltiplos bancos, cada um com o seu buffer de linha
    - **Double Data Rate (DDR SDRAM)** : Transferência de dados tanto no flanco ascendente como no flanco descendente do sinal de relógio (duplica a taxa de transferência de pico);
    - **Estas técnicas melhoram a largura de banda, mas não a latência**

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

a. 4Mx64
b. 8Mx32
c. 16Mx8
d. 32Mx16

64 circuitos

$4M \times 2 = 2^2 \times 2^{20} \times 2^1 = 2^{23}$

$64 = 2^6$ $\quad 23 + 6 = 29$

a) $4M \times 64 = 2^2 \times 2^{20} \times 2^6 = 2^{28}$

b) $8M \times 32 = 2^3 \times 2^{20} \times 2^5 = 2^{28}$

c) $16M \times 8 = 2^4 \times 2^{20} \times 2^3 = 2^{27}$

d) $32M \times 16 = 2^5 \times 2^{20} \times 2^4 = \boxed{2^{29}} = 29$ bits

O número total de words que a memória pode armazenar (Número total de bits = word size * nº words)

- Exemplos:
- 1k x 8
  - 8 bits / word
  - 1k = 2^{10} → 10 linhas de endereço → 1.024 endereços
- 1M x 4
  - 4 bits / word
  - 1M = 2^{20} → 20 linhas de endereço → 1.048.576 endereços

(decoder igual mas com saída em vez de endereços)

15. 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-back para indicar que a informação armazenada no respetivo bloco foi alterada
    c. write-through para indicar que o respetivo bloco não está a ser usado
    d. write-back para indicar que o respetivo bloco não está a ser usado

Políticas de escrita:

- **Write-through:**
  - Todas as escritas são realizadas simultaneamente na cache e na memória principal
  - Se endereço ausente na cache, atualiza apenas a memória principal (**write-no-allocate**)
  - A memória principal está sempre consistente
- **Write-back :**
  - Valor escrito apenas na cache; novo valor é escrito na memória quando o bloco da cache é substituído
  - "**Dirty bit**" (este bit é ativado quando houver uma escrita em qualquer endereço do bloco presente na linha da cache)
  - Se endereço ausente na cache, carrega o bloco para a cache e atualiza-o (**write-allocate**)
  - Mais complexo do que "write-through"
- **Write-back** - Escrita em disco página a página; Página só é escrita em disco quando necessita de ser retirada da memória física
- **Dirty bit :**
  - Se "dirty bit" = 0, a página não necessita de ser escrita em disco aquando da sua substituição (overwrite)
  - Se "dirty bit" = 1, a página que vai ser substituída foi alterada. Antes de ser substituída, essa página é copiada para o disco; após essa operação, a nova página é então copiada do disco para a zona da memória que ficou livre
- **Write-allocate -** Escrita numa página que não reside em memória física: carrega essa página para a memória e escreve

16. Numa cache com associatividade de 4 de 8kBytes e 128 linhas, o número de comparadores necessários para comparar o campo tag de um endereço de acesso à memória é:

a. 4
b. 128
c. 512
d. 8192

3 formas de organizar os sistemas de cache:

- Cache totalmente associativa ("fully associative") = nº de linhas
- Cache com mapeamento direto ("direct mapped") - 1 comaparador ou linha
- Cache parcialmente associativa ("set associative") = associatividade

20. O número de comparadores necessário para a implementação de uma cache de mapeamento direto é de 16Kbytes, blocos de 64 bytes e 256 linhas:

a. 256
b. 1
c. 16
d. 64

17. A técnica de memória virtual permite:

a. a utilização de memória cache no processador para aumentar a dimensão aparente da memória física do sistema
b. que a dimensão da memória física disponível exceda o limite do espaço de endereçamento de um processo
c. implementar mecanismos de proteção através da independência dos espaços de endereçamento de cada processo
d. todas as restantes respostas estão corretas

14. 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 excede o limite de memória física disponível
c. Implementar mecanismos de proteção dada a independência dos espaços de endereçamento de cada processo.
d. Todas estão certas

Memória virtual :

- É uma abstração que permite uma utilização eficiente do sistema de memória em sistemas multi-processo
- Motivações
  - Eficiência na utilização da memória

        - Memória deve ser partilhada pelos vários processos em execução
        - Em memória apenas deve residir a informação necessária
    - Segurança
        - Devem existir mecanismos de segurança que impeçam que um processo altere as zonas de memória dos outros processos
    - Transparência
        - Um processo deve ter acesso à memória de que necessita
        - Um processo deve correr como se toda a memória lhe pertencesse
    - Partilha de memória
        - Vários processos devem poder aceder à mesma zona de memória

18. A tradução de endereços virtuais em endereços físicos consiste na tradução de:

a. physical page number no virtual page number e sua justaposição com o page offset no endereço produzido pelo CPU
b. physical page offset no virtual page offset e sua justaposição com o page number no endereço produzido pelo CPU
c. virtual page offset no physical page offset e sua justaposição com o page number no endereço produzido pelo CPU
d. virtual page number no physical page number e sua justaposição com o page offset no endereço produzido pelo CPU

- Tradução
    - Uma *Page Table* por processo
    - O Virtual Page Number (VPN), obtido do endereço gerado pelo processador, forma o índice de acesso à *Page Table*
- Obtenção do endereço físico da página
    - A entrada da tabela correspondente ao VPN contém informação sobre a página
    - Se "valid bit = 1" a página reside na memória
        - O endereço de acesso é obtido da entrada da tabela correspondente ao VPN concatenado com o "page offset"
    - Se "valid bit = 0" a página reside no disco
        - *Page Fault*
        - Copiar para a memória a página pretendida (pode envolver a substituição de uma página residente em memória)
        - "valid bit" passa a "1"
- Proteção
    - Os atributos de acesso à página (Read, Read/Write, Execute) são verificados em cada tradução (para cada VPN)

19. As páginas de memória virtual possuem normalmente:

a. a mesma dimensão dos blocos da cache para tornar partido das transferências por DMA
b. a mesma dimensão dos blocos da cache para tornar partido das transferências por interrupção
c. uma dimensão de alguns milhares de bytes para tirar partido das transferências por DMA
d. uma dimensão de alguns milhares de bytes para tirar partido das transferências por interrupção

20. Num dado processador um endereço virtual é representado com 32 bits, dos quais 10 bits são usados para o page offset. Este processador é usado num sistema com 1GBytes de memória física. Nestas circunstâncias, o número de páginas virtuais e físicas é, respetivamente:

a. 4M e 1M
b. 4M e 1k
c. 4G e 1M
d. 4G e 1k

endereço virtual representado por

$32 \text{ bits} - \underset{\text{page offset}}{10 \text{ bits}} = 22 \text{ bits} = 2^{22} = \underset{\text{memória virtual}}{4M}$

Sistema tem 1GB de memória física $= 2^{30} = 30$ bits

$30 - \underset{\text{page offset}}{10} = 20 = 2^{20} = 1M \rightarrow$ memória física

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

a. Estruturas de dados do tipo FIFO como meio de comunicação com a aplicação de alto nível, sendo o envio e receção de caracteres da UART processados por interrupt.
b. Interrupções para sinalizar a aplicação de alto nível de que foi recebido um novo caractere ou que a UART está disponível para o envio de um caractere
c.
d. Buffer circular para receção e envio de um caracter por polling

- Realização de um *device driver* para uma UART RS232 (*Universal Assynchronous Receiver Transmitter*) para executar num sistema com microcontrolador (i.e., sem sistema operativo)

- Princípio de operação
  - Desacoplamento da transferência de dados entre a UART e a aplicação realizada por meio de FIFOs (um FIFO de transmissão e um de receção). Do ponto de vista da aplicação:
    - A **transmissão** consiste em copiar os dados a enviar para o FIFO de transmissão do *device driver*
    - A **receção** consiste em ler os dados recebidos que residem no FIFO de receção do *device driver*
  - A transferência de dados entre os FIFOS e a UART é realizada por interrupção, i.e., sem intervenção explícita da aplicação

- Um FIFO pode ser implementado através de um *buffer* circular

Transferência entre FIFO e UART feita por interrupção

11. Sabendo que num dado sistema computacional cada Bus Cycle quer para leitura quer para escrita, demora 100ns. Para a transferência de 512 words (de 32 bits), um controlador de DMA de 16 bits, não dedicado, a funcionar em modo bloco, necessita de:

a. 51,2 µs
b. 102,4 µs
c. 204,8 µs
d. Nenhuma das respostas está correta

100 ns 512 palavras (32 bits) controlador DMA 16 bits

$T = 100\,ns \times 2 \times 2 \times 512 = 400 \times 512 = 204{,}800$

escrita e leitura

não dedicado
2 bus-cycle
ou
32 bits/16 bits
= 2

512×400
000
000
2048
204800

$= 204{,}8\ \mu s$

21. Considere um timer de 16 bits, com reset síncrono, com uma frequência de entrada de 1MHz, que funciona, em modo alternado, com duas constantes de divisão KA e KB. Utilizando o timer como divisor de frequência, e supondo que o tempo a “1” do sinal é determinado pela constante KA, para se obter à saída um sinal com um período de 2ms e duty-cycle de 25%, as constantes KA e KB deverão valer, respetivamente:

a. 500 e 1500
b. 1500 e 500
c. 499 e 1499
d. 1499 e 499

reset sincrono → tem -1
$f_{entrada} = 1MHz$ KA e KB
Quando "1" → KA $T_{saída} = 2ms$ e duty-cycle = 25%
$t_{on} = KA$

- $\text{duty-cycle} = \frac{t_{on}}{T} \times 100 \Leftrightarrow 25 = \frac{t_{on}}{2ms} \times 100$

$\simeq 0{,}25 \times 2 = t_{on} \Leftrightarrow 0{,}5 = t_{on} \Leftrightarrow t_{on} = 0{,}5 - 1 = t_{on} = 500 - 1 = 499\ KA$

reset sincrono

- $T = t_{on} + t_{off} \Leftrightarrow 2 = 0{,}5 + t_{off} \Leftrightarrow 1{,}5 = t_{off} \simeq$
$t_{off} = 1500 - 1$ → reset sincrono
$= 1499\ KB$

22. Considere um sistema baseado num CPU a funcionar a uma frequência de 10MHz, com uma taxa de execução de 2,5 MIPS que processa por interrupção eventos externos periódicos. Se a latência máxima no atendimento a uma interrupção for de 5 ciclos de relógio, e a rotina de serviço à interrupção tiver 30 instruções, a máxima frequência a que esses eventos podem ocorrer é, respetivamente:

a. 71kHz
b. 80kHz
c. 200kHz
d. 285kHz

23. Considere um CPU a funcionar a uma frequência de 50MHz ligado a uma memória com um tempo de acesso de 38ns. O CPU suporta transferências do 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 5ns e o descodificador que gera o sinal de seleção para a memória apresenta um atraso de propagação de 7ns. 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. 3

$f = 50MHz$ $t_{acesso} = 38ns$ Sen i-síncrono

$t_{buffer} = 5ns$ $t_{atraso\ decoder} = 7ns$

- $T = \frac{1}{f} = \frac{1}{50 \times 10^6} = 0{,}02 \times 10^{-6} = 20 \times 10^{-9}\ s$
- atraso = buffer + sinal sel + setup = 5 + 7 + 3 = 15 ns $= 15 \times 10^{-9}\ s$
- t = atraso + acesso memória $= 15_{ns} + 38_{ns} = 53\ ns = \underline{53 \times 10^{-9}\ s}$
- Comparar com ciclos

t = nº ciclos × $T = 2{,}5 \times 20 = \underline{50 \times 10^{-9}\ s}$

Como o dos ciclos < então $\boxed{0}$.

24. 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 1 stop bit, mas com o mesmo baud rate. Nestas circunstâncias o recetor:

a. vai detetar um erro de paridade
b. vai detetar uma trama inválida devido a um número incorreto de stop bits
c. vai detetar um erro de paridade e uma trama inválida devido a um número incorreto de stop bits
d. não vai detectar qualquer erro

paridade par = 0
paridade ímpar = 1

19. Um dispositivo com interface RS232 é configurado para transmitir com 7 bits de dados, paridade par, e 2 stop bits, produz a trama seguida que é recebida por outro dispositivo RS232 incorretamente configurado para 8 bits de dados, paridade par e 1 stop bit mas com o mesmo baudrate. Nestas circunstâncias o recetor:

a. Vai detetar uma trama inválida devido a um número incorreto de stop bits
b. Vai detetar um erro de paridade
c. Vai detetar um erro de paridade e uma trama inválida devido a um número incorreto de stop bits
d. Não vai detetar qualquer erro

Sincronização

- Entre instantes de sincronização o desvio dos relógios depende da estabilidade/precisão dos relógios do transmissor e do recetor
- Exemplo em que a receção não é corretamente efetuada devido a um desvio da frequência dos relógios do transmissor e do recetor

- Neste exemplo o recetor detetaria dois erros:
  - **Erro de paridade** (se o bit D6 for detetado como 1): o bit de paridade devia ser 0 e é lido como 1
  - **Erro de framing**: é detetado o nível lógico 0 no instante em que era

25. Considere um processador com um espaço de endereçamento de 32 bits e uma memória cache com associatividade de 2, de 16kByte e blocos de 32 bytes. A dimensão, em bits, dos campos tag, set, e byte é:

a. tag: 19; set: 8; byte: 5
b. tag: 18; set: 9; byte: 5
c. tag: 13; set: 14; byte: 5
d. tag: 14; set: 16; byte: 2

espaço endereçamento 32 bits
associatividade 2 = 2 de 16 Kbyte blocos 32 bytes

- 16 Kbyte / 2 = 8 K = $2^3 \times 2^{10}$ = 13 bits (linhas da memoria cache)
- blocos 32 bytes → 2^5 = 5 bits para byte
- 13 - 5 = 8 bits para set
- endereçamento 32 bits - 13 = 19 bits para tag

20. Considere um sistema de supervisão, baseado no protocolo I2C, 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 400KHz, necessita para adquirir os valores de todos os sensores (cada um implementado num slave distinto) é, aproximadamente:

a. 0,40ms
b. 0,45ms
c. 0,50ms
d. Nenhuma das respostas está correta

10 sensores de 8 bits
f = 400 KHz

- 1 start bit + 7 endereço + 1 wr + 1 ack + 8 bits dados + 1 bit $\overline{Ack}$ + 1 stop bit = 20 bits
- $T = \frac{1}{400K} = 2,5\ \mu s$
- 2,5 × 20 × 10 sensores = 500 µs ≃ 0,5 ms

26. O número total de bits de armazenamento (dados e controle) necessário para a implementação de uma memória cache com associatividade de 2, de 16kByte e blocos de 32bytes, com política de escrita do tipo write-back, num espaço de endereçamento de 32 bits, é:

a. 140288
b. 140800
c. 141312
d. 141824

27. Num sistema que suporta um nível de cache e memória virtual:

a. no espaço de armazenamento secundário (disco) estão armazenadas as páginas de memória virtual mais recentemente acedidas e na memória cache estão armazenados os blocos dessas páginas mais recentemente acedidos
b. os blocos da cache e as páginas de memória são tipicamente da mesma dimensão
c. enquanto é efetuado o processamento de um page fault de um processo, o processador pode estar ocupado a executar outro processo
d. todas as restantes respostas estão corretas

b esta mal porque não é cache e memória é : Os espaços de endereçamento virtual e físico são divididos em blocos; os blocos têm a mesma dimensão nos dois espaços de endereçamento

24. O número mínimo de bits de controlo (excluindo os de dados) necessários para a implementação de uma memória cache com mapeamento direto, de 64Kbyte de dados organizados em blocos de 16 Bytes, com política de escrita do tipo write-through, num espaço de endereçamento de 32 bits é:

a. 64K
b. 68K
c. 72K
d. Superior a 100K

mapeamento direto → 1 cache 64 Kbytes
16 bytes
espaço endereçamento

- $64K = 2^{16} = 16$ bits
- 16 bytes → $2^4 = 4$ bit p/byte
- $32 - 16 = 16$ bits p/tag
- $16 - 4 = 12$ bits p/groupset → $2^{12} = 4K = 4096$ linhas de cache
- 1 bit valid + 16 tag = 17 bits de controlo
- $4096 \times 17 = \boxed{68K}$

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

a. depende da implementação sendo sempre muito inferior ao número máximo de entradas da page table
b. igual ao número máximo de páginas virtuais
c. igual ao número máximo de páginas virtuais de memórias usadas pelo processo em execução
d. igual ao número de entradas da page table

Tradução de endereços

- A *Page Table* contém um número de entradas igual ao número máximo de páginas virtuais (para o exemplo dado anteriormente, a tabela teria 2^{20} entradas). Por esta razão não é necessária uma "tag"

28. Num sistema de memória virtual de 32 bits em que a dimensão da página é de 2kBytes, a page table de cada processo tem um total de:

a. 4k entradas
b. 1M
c. 2M
d. nenhuma das anteriores

mem. vir → 32 bits

dim. pág → 2Kbytes

page table tem um total de:

- $2K = 2^{11} = 11$ bits
- $32 - 11 = 21 = 2^{21} = \boxed{2M}$

3. O modelo de programação de um periférico específica:

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. as arquiteturas e as ferramentas de desenvolvimento com as quais o periférico pode ser usado
d. o conjunto de registos, respetivos campos e modos de acesso suportados pelo periférico

1. O modelo de programação de um periférico especifica:
   a. a funcionalidade do periférico e o conjunto de registos de dados de controlo e de status
   b. o subconjunto de instruções assembly do CPU suportadas por esse periférico
   c. os sinais elétricos usados na ligação do periférico a dispositivos externos, tais como sensores e atuadores
   d. as arquiteturas e as ferramentas de desenvolvimento com as quais o periférico pode ser usado

4. 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

5. O diagrama temporal da figura do lado representa um ciclo de:
   a. leitura de um dispositivo em que os sinais de controlo usam lógica negativa
   b. leitura de um dispositivo em que os sinais de controlo usam lógica positiva
   c. escrita num dispositivo em que os sinais de controlo usam lógica negativa
   d. escrita num dispositivo em que os sinais de controlo usam lógica positiva

6. O sinal de seleção “Sel” (ativo alto) de um porto mapeado na gama de endereços 0x0000...0x07FF de um processador com um espaço de endereçamento de 16 bits pode ser obtido através da expressão:

a) $Sel = \prod_{i=0}^{11} A_i$ b) $Sel = \prod_{i=11}^{15} A_i$ c) $Sel = \prod_{i=0}^{11} A_i$ \ d) $Sel = \prod_{i=11}^{15} A_i$ \

0x0000 ... 0x07FF

0000 0000000 0000

0000 0111 1111 1111

16 - 5 = 11

$Sel = \prod_{i=11}^{15} \overline{A_i}$

7. Numa transferência assíncrona:
   a. assume-se que o dispositivo externo responde à velocidade do CPU e, consequentemente, não existem sinais de protocolo envolvidos no handshake da transação existem sinais de controlo
   b. 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 semi-sincrono
   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

Transferência ponto a ponto

- O processo de transferência de informação pode ser descrito como uma sequência de **ações elementares**.

Transferência ponto a ponto

- Protocolo da ligação:
  - Especifica a **sequência de ações elementares** que coordenam a informação e o respetivo controlo
  - A ativação de cada ação elementar do protocolo está associada com o estado ou mudança de estado de uma variável booleana (sinal)
  - Essa variável pode estar associada a uma linha física ou estar codificada: N linhas podem servir para codificar 2^N ações.
- Ciclo:
  - **Sequência completa de ações elementares** que permitem transferir uma "unidade" de informação da fonte para o destino.
- Exemplo:

  - **VALID**: Sinal ativado pela fonte e usado pelo destino para armazenar INFO. É ativado quando as linhas de INFO já estão estáveis.

Transferência múltipla SÍNCRONA

- Não utiliza sinais de reconhecimento (i.e. não há qualquer sinal que indique que a informação tenha sido consumida pelo destino)

- A lógica de controlo da FONTE assume que a informação foi armazenada pelo DESTINO ao fim de um tempo t_A
- A FONTE tem controlo total das temporizações, sincronizando a transferência

Transferência múltipla ASSÍNCRONA ("handshaken")

- Utiliza sinais de protocolo para assegurar a correta troca de informação entre a origem e o destino
- Pode ser utilizada na comunicação com dispositivos muito lentos (quando o tempo que decorre desde o pedido da operação até à sua conclusão é longo e variável)

- Solução intermédia – protocolo **semi-síncrono**:
  - Opera, por defeito, de forma síncrona mas torna-se "handshaken" quando houver um pedido explícito, por parte do *slave*, para alteração da temporização das ações
  - Um dispositivo lento pode, através da ativação de um sinal adequado, atrasar a conclusão da operação; no caso em que o sinal não é ativado a transferência é síncrona (só os dispositivos lentos necessitam de ativar esse sinal)
- O protocolo semi-síncrono permite a coexistência de dispositivos rápidos e de dispositivos lentos no mesmo sistema

4. Numa transferência semi-síncrona:
   a. assume-se que o dispositivo externo responde à velocidade do CPU, e consequentemente, não existem sinais de protocolo envolvidos na transação
   b. 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
   c. O CPU prolonga o ciclo de leitura/escrita até que o dispositivo externo sinalize que a operação pretendida foi completada
   d. nenhuma das respostas está correta

29. Numa transferência do tipo síncrono:
   a. É necessário um sinal de Acknowledge gerado pelo periférico. assincrono , sincrono Não tem sinais
   b. A transferência efetua-se apenas nas transições do sinal de relógio. quando o sinal esta valido não so no clk
   c. O periférico sinaliza o fim da transferência através de um sinal deWait. semi-sincrono
   d. Nenhuma das anteriores está correta.

8. A figura do lado corresponde ao diagrama temporal de:

   a. uma operação de escrita numa transferência síncrona com dados e endereços multiplexados numa configuração micro-ciclo
   b. uma operação de leitura numa transferência síncrona com dados e endereços não multiplexados numa configuração merged
   c. uma operação de leitura numa transferência assíncrona com dados e endereços multiplexados numa configuração micro-ciclo e strobes independentes
   d. uma operação de escrita numa transferência assíncrona com dados e endereços não multiplexados numa configuração merged e strobes independentes

9. A técnica de entrada/saída de dados por interrupção:
   a. permite transferir eficientemente (com elevado throughput) grandes quantidade de informação
   b. permite mascarar a latência do periférico
   c. consiste na execução de um ciclo de polling que é interrompido quando o periférico estiver pronto para realizar a transferência
   d. consiste na interrupção do periférico sempre que o CPU pretende transferir informação

Técnicas de transferência de informação

1. O CPU inicia e controla a transferência de informação:
   - E/S programada (***programmed I/O***)
     - O CPU toma a iniciativa – aguarda se necessário, inicia e controla a transferência de informação (**POLLING**)
   - E/S por interrupção (***interrupt driven I/O***)
     - O periférico sinaliza o CPU de que está pronto para trocar informação (leitura ou escrita). O CPU inicia e controla a transferência

2. O CPU não toma parte na transferência de informação:
   - E/S por acesso directo à memória (**DMA**)
     - A informação é trocada diretamente entre a memória e o periférico. O CPU não toma parte no processo de transferência
     - O CPU apenas configura inicialmente o DMA e é sinalizado no final de que a transferência terminou

E/S por interrupção

- Na técnica de E/S por interrupção quando o periférico está pronto para disponibilizar/receber informação sinaliza o CPU
- Uma interrupção, depois de reconhecida, faz com que o CPU abandone temporariamente a execução do programa em curso para executar a rotina que dá seguimento à interrupção gerada
  - A rotina associada à interrupção designa-se por **rotina de serviço à interrupção** ou ***interrupt handler***
- A transferência é também efetuada pelo CPU mas o tempo de espera é eliminado, uma vez que a interrupção ocorre quando o periférico está pronto para a troca de informação
- Esta técnica mascara o problema da longa latência descrito no exemplo de leitura de informação de uma unidade de disco (slide 3)

3. Quando é usada a técnica de entrada/saída de dados por interrupção:
   a. o periférico faz um pedido de interrupção ao CPU após a conclusão da transferência de dados
   b. o periférico faz um pedido de interrupção ao CPU quando estiver pronto para transferir dados
   c. o CPU interrompe a execução do programa para configurar o controlador de DMA que fará a transferência propriamente dita
   d. o DMA verifica através de um ciclo de polling se o periférico está pronto para transferir os dados e quando tal se verifica gera um pedido de interrupção do CPU

10. Quando é usada a técnica de entrada/saída de dados por DMA:
   a. o CPU verifica através de um ciclo de polling ao registo de dados do controlador de DMA se a transferência já foi concluída
   b. o periférico faz um pedido de interrupção ao CPU quando estiver pronto para transferir os dados
   c. o CPU configura o controlador de DMA que fará por software a transferência propriamente dita
   d. o CPU configura o controlador de DMA que fará por hardware a transferência propriamente dita

13. Suponha que pretende interligar, através de um protocolo/interface série, dois sistemas computacionais que distam de alguns metros numa linha de produção com elevados níveis de interferência eletromagnética, um ligado a um sensor e outro correspondendo a um computador de controlo. O standard mais adequado a este cenário de aplicação é:
   a. SPI
   b. I2C
   c. CAN
   d. RS232

14. No standard RS-232 um dos tipos de erro de comunicação que é detetado é o erro de framing. Esse erro ocorre quando o recetor:

a. recebe um número de bits a "1" que não corresponde à paridade programada
b. recebe um bit de paridade diferente do programado
c. deteta um número de bits no campo de dados diferente do programado
d. recebe como stop bit um bit com o nível lógico "0"

Erro framming - deteta nivel lógico “0” no instante que era esperado stop bit
Erro de paridade - bit deveria ser “0” e é lido com 1
Erro de fase - aumento do fator de sobreamostragem

10. Na interface RS-232 uma das fontes de erro do instante de amostragem dos vários bits no recetor é o “erro de fase”. Esse erro diminui:

a. com o aumento do número de stop bits
b. com a diminuição do tamanho da trama
c. com a diminuição do fator de sobreamostragem
d. com o aumento do fator de sobreamostragem

15. A interface SPI permite a comunicação entre:

a. um master e vários slaves, numa ligação em daisy-chain, sendo o sinal de relógio implícito
b. um master e vários slaves, numa ligação com sinais de seleção individuais, sendo o sinal de relógio explícito
c. vários masters e vários slaves, numa ligação com sinais de seleção individuais, sendo o sinal de relógio implícito
d. vários masters e vários slaves, numa ligação com sinais de seleção individuais, sendo o sinal de relógio explícito

Ligação a curtas distâncias
Arquitetura "Master-Slave" com ligação ponto a ponto:

- O sistema só pode ter um master
- O master é o único dispositivo no sistema que pode controlar o relógio
- Um master pode estar ligado a vários slaves: para cada comunicação, apenas 1 slave é selecionado pelo master

Comunicação bidirecional "**full-duplex**"
Comunicação síncrona:

- Relógio é gerado pelo master que o disponibiliza para todos os slaves
- Não é exigida precisão ao relógio - os bits vão sendo transferidos a cada transição de relógio. Isto permite utilizar um oscilador de baixo custo no master

Fácil de implementar por hardware ou por software
Arquiteturas de ligação:

- slaves independentes

    - Sinais de seleção ("slave select") independentes
    - Em cada instante apenas um SSx\ está ativo, isto é, apenas 1 slave está selecionado
    - O número máximo de slaves está limitado pelo número de linhas de seleção disponibilizadas pelo master
- Daisy Chain (cascata)
    - Sinal "slave select" comum, SDO/SDI ligados em cascata
    - A saída de dados de cada slave liga à entrada de dados do seguinte
    - O conjunto dos slaves é visto pelo master como um único dispositivo de maior dimensão

RELÓGIO: explícito - SPI implícito - RS232
codificado - USB, Eth

21. Num espaço de endereçamento de 16 bits, um descodificador implementado através da expressão lógica "Sel\ = A15 + A13\ + A12 + A11\", descodifica a(s) seguinte(s) gama(s) de endereço(s):

a. 0x2000 a 0x27FF, 0x6000 a 0x67FF
b. 0x2800 a 0x2FFF, 0x6800 a 0x6FFF
c. 0x9000 a 0x97FF, 0xD000 a 0xD7FF
d. 0x9800 a 0x9FFF, 0xD800 a 0xDFFF

$\overline{SEL} = A_{15} + \overline{A_{13}} + A_{12} + \overline{A_{11}} \rightarrow SEL = 0x101xxx\ldots$

a) 0010 X000 ... a 0010 0111 ...

b) 0010 1000 ... a 0010 1111 ...
0110 1000 ... 0110 1111 ...

c) 100 X000 ...

d) 1001 X000

22. Num porto de entrada constituído por buffers tri-state (na ligação entre os sinais externos e o barramento de dados do processador), o sinal de Enable ativo alto é obtido a partir dos sinais Sel\ e RD\ (ambos ativos baixos) de acordo com a expressão lógica:

a. Enable = Sel\ . RD\
b. Enable = (Sel\ . RD\)\
c. Enable = Sel\ + RD\
d. Enable = (Sel\ + RD\)\

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

Se for por hardware pelo CPU :

- Identificação da fonte de interrupção (nos casos em que tal é efetuado por hardware) e obtenção do endereço da RSI
- Salvaguarda do contexto atual do CPU (valor corrente do PC e de flags de estado associadas ao sistema)
- Desativação das interrupções
- Carregamento no PC do endereço da RSI
- Execução da RSI até encontrar a instrução de retorno
- Execução da instrução de retorno da RSI (Reposição do contexto salvaguardado e reativação das interrupções; Regresso ao programa interrompido, executando a instrução que teria sido executada se não tivesse acontecido a interrupção)

Por Software (RSI - rotina de serviço a interrupção):

- Salvaguarda do contexto do programa que foi interrompido(registos internos do CPU → memória (stack) ("PRÓLOGO"))
- ações associadas ao processamento da interrupção.
- Reposição do contexto do programa interrompido(registos internos do CPU ← memória (stack) ("EPILOGO"))
- Conclusão da RSI com a instrução de retorno

24.1. Num sistema de interrupções vectorizadas:

a. A identificação da fonte é realizada por hardware.
b. Os periféricos podem estar agrupados em cadeia
c. A cada periférico é atribuído um vector único
d. todas as anteriores

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

Interrupções vetorizadas

- CPU tem apenas 1 entrada de interrupção
- A identificação da fonte é feita por hardware
- Cada periférico possui um identificador único, designado por **vetor**
- Durante o processo de atendimento, na fase de identificação da fonte, o periférico gerador da interrupção identifica-se através do seu vetor
- Este vetor vai ser usado depois como índice de uma tabela que especifica como chegar à RSI

- **Uma RSI para cada vetor de interrupção**

- A identificação da fonte de interrupção é feita por hardware num processo genericamente designado por "Interrupt acknowledge cycle"
- Periféricos podem estar organizados numa estrutura daisy chain (A ordem de colocação dos periféricos na cadeia, relativamente ao CPU, determina a sua prioridade)

25. Numa transferência por DMA, em modo bloco, quando o controlador de DMA pretende dar início a uma transferência:

a. ativa o sinal busreq durante um número fixo de ciclos de relógio, efetuando de seguida a transferência
b. ativa o sinal busreq, efetuando a transferência logo que se torne no bus master
c. gera uma interrupção que é interpretada pelo CPU como um pedido de cedência dos barramentos; a transferência é efetuada quando o DMA reconhecer a ativação do sinal busgrant
d. sinaliza o CPU, através da linha busreq, que vai dar início à transferência e inicia-a de imediato; o sinal busgrant é utilizado pelo CPU para suspender a atividade do DMA

27. Considere um timer de 16 bits, com reset assíncrono, com uma frequência de entrada de 1MHz, que funciona, em modo alternado, com duas constantes de divisão KA e KB. Utilizando o timer como divisor de frequência, e supondo que o tempo a "1" do sinal é determinado pela constante KB, para se obter à saída um sinal com um período de 10ms e duty-cycle de 40%, a constante KA deverá valer:

a. 4000
b. 3999
c. 6000
d. 5999

reset assíncrono → não tem -1 no fim

f = 1MHz KA e KB ton = KB. $T_{saída}$ = 10ms = 10000 µs

duty-cycle = 40%

• $T = \frac{1}{1\times10^6} = 1\times10^{-6}$ s = 1×10^{-9} ns

• duty-cicle = $\frac{t_{on}}{T}\times100 \Leftrightarrow 40 = \frac{t_{on}}{1\times10^{-9}}\times100 \Leftrightarrow t_{on} = 0{,}4\times1\times10^{-9}$

• $t_{on} = 0{,}4\times10^{-9}$ ns ⇔ KB = 4000 µs (são 3 para frente)

• T = 4000 + Ka ⇔ 10 000 - 4000 = Ka ⇔ Ka = 6000 µs

28. 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 15 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 4 ns e o descodificador que gera o sinal de seleção para a memória apresenta um atraso de propagação de 6 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

f = 100 MHz tacesso = 15 ns semi-síncrono

setup = 3 ns buffer = 4 ns sel = 6 ns

• $T = \frac{1}{100\times10^6} = 0{,}01\times10^{-6} = 10\times10^{-9}$ s.

• atraso = buffer + setup + select = 3 + 4 + 6 = 13 ns = 13×10^{-9} s

• t = atraso + tacesso = 13 + 15 = 28 ns = 28×10^{-9} s

• Comparar c/ciclos:

t = n°ciclos × T = 2,5 × 10 = 25×10^{-9} s

• como nciclos < então 0.

2. Na arquitetura de um sistema computacional, o data bus permite:
   a. identificar, na memoria externa/periférico, a origem/destino dos dados
   b. especificar o tipo de operação efetuada sobre a memoria
   c. transferir dados entre a memoria externa/periféricos e os registos do CPU
   d. transferir o código máquina das instruções para o program counter

Data bus faz a ligação entre o CPU e os periféricos

5. O overhead da transferência de informação por interrupção é devido a:
   a. retorno ao programa interrompido
   b. salto para a rotina de serviço à interrupção
   c. salvaguarda e reposição do contexto do programa interrompido,i.e, registos internos do cpu.
   d. configuração do controlador de interrupções

Overhead do método de transferência por interrupção

- O ***overhead*** global do método de transferência por interrupção é causado pela mudança de contexto:
  - A rotina de serviço à interrupção tem que, à entrada, **salvaguardar o contexto do programa interrompido**
  - Antes de abandonar a RSI tem que **repor o contexto salvaguardado**
  - A título de exemplo, estas 2 operações requerem, no MIPS, cerca de 40 instruções
- Em sistemas computacionais mais evoluídos, outro aspeto negativo da mudança de contexto é a que resulta da, muito provável, mudança da informação nas memórias cache (a ver mais tarde)
  - A RSI poderá utilizar zonas de memória diferentes das do programa interrompido, o que obriga à atualização das memórias *cache* com o consequente impacto no número de ciclos de relógio gastos
  - Por outro lado, o regresso ao programa interrompido tem uma consequência semelhante, obrigando à atualização das memórias cache, desta vez com as zonas de memória que o programa estava a utilizar antes de ocorrer a interrupção

Overhead - nº de ciclos do clk gastos pelo CPU em tarefas não relacionadas diretamente com transferência de informação, depende do nº de vezes do polling

8. Considere um timer em que a relação entre as freq. De entrada e de saída é dada por (k+1) em que k é uma constante configurável. Se a freq de entrada do timer for 20MHz e se se pretendes obter á saída um sinal com um período de 2ms, o valor de k deve ser:
   a. k = 20 x 10^9
   b. k = 1999
   c. k=39999
   d. k = 9999

$f_{in} = (K+1) f_{out}$ $\quad f_{in} = 20MHz$ $\quad T_{saída} = 2\,ms = 2 \times 10^{-3}\,ns$

- $T_{inicial} = \frac{1}{20 \times 10^6} = 0{,}05 \times 10^{-6}\,ns = 50 \times 10^{-9}\,s$

- $T_{saída} = \frac{1}{f_{out}} \Leftrightarrow 2 \times 10^{-3} = \frac{1}{f_{out}} \Leftrightarrow f_{out} = 0{,}5 \times 10^{3}\,Hz$

$20 \times 10^6 = (K+1)\,0{,}5 \times 10^3 \Leftrightarrow 40 \times 10^3 = K+1 \Leftrightarrow$

$\Leftrightarrow K = 40000 - 1 \Leftrightarrow \boxed{K = 39999}$

16. No barramento CAN , cada trama de dados contém:

a. a identificação do nó produtor de informação
b. a identificação do nó destino da informação
c. a identificação do conteúdo de informação
d. a identificação dos dois nós: o produtor e o destino da informação

17. No barramento CAN a codificação das tramas de dados utiliza a técnica de "bit stuffing".Essa técnica :

a. por cada 5 bits iguais é inserido um de polaridade oposta
b. por cada bit enviado é inserido um de polaridade oposta
c. por cada 5 bits diferentes é inserido um a '0'
d. por cada 5 iguais é inserido um a '1'

- "**Bit-stuffing**"
  - Por cada 5 bits iguais é inserido 1 de polaridade oposta
  - Garante tempo máximo entre transições da linha de dados, assegurando que há transições suficientes para manter os relógios dos nós sincronizados

18. Numa memoria estática SRAM:

a. as células necessitam de refrescamento regular
b. cada célula de 1 bit é implementada com um mínimo de 6 transístores
c. o tempo de acesso aumenta com o valor do endereço
d. existe um único barramento para os dados e endereços de modo a diminuir o número de pinos do CPU

- **SRAM – Static RAM**
  - Vantagens:
    - Rápida
    - Informação permanece até que a alimentação seja cortada
  - Inconvenientes:
    - Implementações típicas: 6 transistores / célula
    - Baixa densidade, elevada dissipação de potência
    - Custo/bit elevado

19. Numa memoria DRAM de 256Mx1, implementada numa matriz quadrada, o número de barramento de endereços é:

a. 8
b. 1
c. 27

$256M \times 1 = 2^8 \times 2^{20} \times 1 =$

$= 2^{28} = 2^{2 \times 14} \Rightarrow$ 14 bits

matriz quadrada

d. 14

22. Na técnica normalmente designada “memoria virtual” o número de entradas da page table é:

a. igual ao número de entradas da memoria TLB
b. igual ao número máximo de páginas físicas
c. igual ao número de páginas de memoria física usadas pelo processo em execução
d. igual ao número máximo de páginas virtuais

23. O valid bit de uma entrada da page table, quando ativo, indica que:

a. a respetiva página foi acedia recentemente
b. a respetiva página esta residente em memoria física???fisica ou não ?
c. a informação da respetiva página foi alterada
d. a respetiva pág. esta residente em disco

- Se "valid bit = 1" a página reside na memória
  - O endereço de acesso é obtido da entrada da tabela correspondente ao VPN concatenado com o "page offset"
- Se "valid bit = 0" a página reside no disco
  - *Page Fault*
  - Copiar para a memória a página pretendida (pode envolver a substituição de uma página residente em memória)
  - "valid bit" passa a "1"

- "**Page Table register**" – registo que contém o endereço de início da *Page Table* do processo ativo
- "**valid bit**" – bit que indica se a página está ou não em memória
- "**access bits**" – atributos de acesso à página

25. Num sistema que suporte memoria virtual e cache qual das seguintes situações pode ocorrer acesso á memoria

a. TLB miss, Page table miss, cache hit
b. TLB miss, Page table hit, cache miss
c. TLB hit, Page table miss, cache miss
d. TLB hit, Page table miss, cache hit

DICA: tem de dar HIT na page table

16. Num processador que detenha sinais específicos para implementar Isolated I/O, o acesso aos periféricos:

a. Pode apenas ser efetuado por instruções específicas para I/O.
b. Pode apenas ser efetuado pelas mesmas instruções de acesso à memória. Memory mapped I/O
c. Pode ser efetuado por ambos os tipos de instruções, dependendo da forma como esteja projetado o hardware.

d. Nenhuma das anteriores está correta.

21. A equação lógica CE\ = A18\ + A17 + A16\ + A15 + X em que X = IO/M* seleciona um bloco de memória de:

a. 128K
b. 32K
c. 16K
d. Nenhuma das anteriores está correta.

endereço de 20 bits
A18, A17, A16 e A15 já definidos logo
20 - 4 = 16 bits
2^{16} = 64 K //

25. Num porto de entrada constituído por buffers tri-state, o sinal de ativação ativo baixo (EN\) é obtido a partir dos sinais CE\ e RD\ de acordo com a expressão lógica:

a. En\ = (CE\ + RD\)\
b. En\ = (CE\ . RD\)\
c. En\= CE\ . RD\
d. En\ = CE\ + RD\

$EN = CE \cdot RD$ $\overline{EN} = \overline{(CE \cdot RD)}$
$\Leftrightarrow EN\backslash = CE\backslash + RD\backslash$

27. Uma ligação ponto a ponto dispõe dos seguintes sinais: STROBE, ACKNOWLEDGE, R/W*. Em princípio deverá estar a utilizar um protocolo:

a. Síncrono
b. Semi-sncrono
c. Handshaken
d. Nenhuma das anteriores

STROBE E R/W é comum a todos

ACK - assincrono (hadshaken)
Wait - semi-sincrono

28. Os protocolos síncronos são vantajosos quando:
    a. Todos os dispositivos presentes no sistema são rápidos
    b. Todos os dispositivos presentes no sistema são lentos
    c. Os dispositivos presentes no sistema apresentam uma velocidade homogénea
    d. Os dispositivos presentes no sistema apresentam uma velocidade heterogénea

é a c. pois a velocidade dos protocolos síncronos dependem da unidade + lenta

34. Considere-se um CPU que funciona a uma frequência de 500 MHz e o ciclo de leitura corresponde ao da figura. Pretende-se ligar a este CPU uma memória RAM com um tempo de acesso de 15 ns. Sendo uma transferência de tipo semi-síncrono, qual o número de wait-states que é necessário introduzir para que a operação decorra com sucesso? (Na figura tem 2 ciclos)
    a. 2
    b. 4
    c. 6
    d. 8

- $T = \frac{1}{500 \times 10^6} = 0{,}002 \times 10^{-6} = 2 \times 10^{-9}$ s
- $t_{acesso} = 15$ ns $/2 = 7{,}5$ → são precisos 8 ciclos.
- 8 - 2 ciclos = 6 ciclos

não sei porque este 2.

50. Um watchdog é um circuito que permite:
    a. Fazer reset ao CPU após uma certa temporização.
    b. Receber linhas de interrupção de periféricos e controlar a sua ligação ao CPU.
    c. Que um controlador de DMA fique indefinidamente a controlar o barramento após ele lhe ser atribuído.
    d. É evidente que é um animal doméstico.

"Watchdog Timer" (temporizador "cão de guarda")

- Sistemas baseados em microprocessador podem assegurar funções de controlo críticas que não podem falhar
- Como garantir que um crash do microprocessador não compromete o funcionamento global do sistema?
- Um "watchdog timer" tem como função monitorizar a operação do microprocessador e, em caso de falha, forçar o seu reinício
- Situação mais comum: se o processador não atuou a entrada de *Reset* do "watchdog timer" ao fim de um tempo pré-determinado o "watchdog timer" força o *Reset* do microprocessador

52. Considere um watchdog timer, com uma frequência de entrada de 1 MHz, construído a partir de um contador de 8 bits que, sempre que a contagem atinge o valor máximo, força o reset do processador que esta a monitorizar. O programa a correr nesse processador tem que, periodicamente, colocar o valor do contador a 0. De modo a impedir o reset do processador, o período de atuação no watchdog timer não pode ser superior a, aproximadamente:

a. 0,25 ms
b. 0,5 ms
c. 1,0 ms
d. 2 ms

$f_{in} = 1MHz \qquad K = 2^8$ (8bits)

$f_{out} = \frac{f_{in}}{2^8} \Leftrightarrow f_{out} = \frac{1 \times 10^6}{2^8} \Leftrightarrow f_{out} = \frac{1 \times 2^{20}}{2^8}$

$\boxed{10^6 = 2^{20}} \Leftrightarrow f_{out} = 1 \times 2^{12} = 1 \times 2^2 \times 2^{10}$

$= 4 \times 10^3$ Hz

$T = \frac{1}{4 \times 10^3} = 0{,}25 \times 10^{-3}$

$\boxed{10^3 = 2^{10}}$

59. A fetch e o deposit duram 2 ciclos cada. Considerando uma frequência de 5MHz, 1024 palavras para transferir de 32 bits cada e um barramento de dados de 8 bits, a duração da transferência se o controlador de DMA for dedicado funcionando em bloco e a duração se for em modo cycle-stealing, é respetivamente:

a. 3,3 ms e 4,9 ms

b. 4,9 ms e 3,3 ms
c. 1,6 ms e 4,9 ms
d. 1,6 ms e 3,3 ms

Dma dedicado - fetch OU deposit
DMA não dedicado - fetch E deposit
DMA em Cycle-Stealing - fetch espera um ciclo faz deposit espera + 1 ciclo

$f = 5MHz$ 1024 palavras (32 bits cada) barramento 8 bits

- $T = \frac{1}{5 \times 10^6} = 0,2 \times 10^{-6} = 200\,ns$ (200×10^{-9})
- bytes p/transferência → $1024 \times 32 / 8 = 4096$ bytes
- é dedicado → $2T$

  $t = 2 \times 200 \times 10^{-9} \times 4096 = \underline{1,6\,ms}$
- modo cycling

  $t = 2T + T + 2T + T = 6T$

  $t = 6 \times 200 \times 10^{-9} \times 4096 = \underline{4,9\,ms}$

71. No protocolo SPI o relógio é:
    a. Explícito, gerado pelo Master.
    b. Explícito, gerado pelo Slave
    c. Implícito
    d. Nenhuma das anteriores

75. A ligação de periféricos em USB é feita com uma topologia de:
    a. Mesh
    b. Tiered Star
    c. Barramento
    d. Qualquer das anteriores

77. A especificação USB suporta diversos tipos de transferências, entre elas:
    a. Controlo pequena dimensao
    b. Isócronas largura de banda larga garantida e latencia limitada
    c. Bulk grandes
    d. Todas as anteriores

87. Numa memória estática (SRAM) de 64Mx8 o número de transístores que constitui a área de armazenamento é aproximadamente:

a. 537x10^6
b. 67 x10^6
c. 403 x10^6
d. 3220 x10^6

94. O número de bits dos barramentos de endereços e de dados de uma memória dinâmica de 16Mx32 é respetivamente:

a. 24 e 32
b. 32 e 24
c. 12 e 32
d. 16 e 32

endereços → m dado → n $2^{2m} \times n$

$16M \times 32 = 2^4 \times 2^{20} \times 32 = 2^{24} \times 32$

$= 2^{2 \times 12} \times 32$

$\boxed{m = 12 \quad n = 32}$

98. Numa memória cache, caso os dados pretendidos não se encontrarem num bloco de nível primário ocorre:

a. Um hit
b. Um miss
c. O envio de um sinal para o CPU a dizer que a informação está inválida
d. Nenhuma das anteriores

100. Considere um processador com um espaço de endereçamento de 32 bits e uma memória cache de mapeamento direto com 512 linhas de 128 bytes cada uma. A dimensão, em bits, dos campos tag, group e byte é:

a. Tag: 7; Group: 16; Byte: 9
b. Tag: 9; Group: 7; Byte: 16
c. Tag: 16; Group: 9; Byte: 7
d. Nenhuma das anteriores

32 bits 512 de 128 bytes
linha

- nº linhas = 512 = 2^9 → group 9 bits
- dimensão = 128 bytes ⇒ 2^7 → 7 bit p/byte
- 32 − (9+7) = 16 bits p/tag

104. O que é uma Memória Virtual?

a. Um tipo de memória, onde os seus endereços são diretamente apontados pelo CPU
b. Um protocolo que separa endereços e dados de memória física

c. Uma técnica que permite aumentar a dimensão aparente da memória física do sistema
d. Nenhuma das anteriores

105. Se uma página que o CPU pretenda aceder não está em memória:
a. É gerado “page fault”
b. O CPU gera uma exceção e a informação é transferida do disco para a memória
c. Elabora uma tarefa que pode durar milhões de ciclos de relógio
d. Todas as anteriores

Um sistema possui um espaço de endereçamento virtual de 4Gbytes, páginas de memória de 8kBytes e 512Mbytes de memória física. Considere tambem:

- **Que num dado instante está a executar um processo cujo *Page Table Register* possui o valor 0x01230000**
- **Que cada entrada da *Page Table* possui 32 bits, está alinhada em endereços múltiplos de 4 e contém a seguinte informação:**

Valid, Dirty, Read, Write, Execute flags [31:27]	*Bits não usados [26:16]*	*PPN [15:0]*

- **O conteúdo de algumas posições da memória principal a seguir indicados:**

Endereço	Valor
...	
0x01230008	0xB0000002
0x0123000C	0xF0000003
0x01230010	0xB8000001
0x01230014	0xA0000000
...	

28. Num acesso à memoria, o CPU produz o endereço 0x00006000, o qual é traduzido no seguinte endereço físico:
a. 0x0000600C
b. 0x0000200C
c. 0x0000100C
d. nenhuma das respostas estão corretas

mem. virtual → 4 Gb
pag. memória → 8 Kb
mem. física → 512 Mb

pag.mem → 8 K = 2^{13} = 13 bytes → 0 0000 0000 0000

• Sobram
0 0 0000 0000 011 x4 = 000 0001100 = ?

• Somando ao 0x01230000
0x01230010 → 0x B800 0001
ver tabela

c) 0x0000100C

29. O processo em execução pode aceder ao endereço virtual 0x0000A010 para:
a. leitura
b. escrita

c. leitura e escrita
d. nenhuma das respostas está correta

Não sei fazer porque tem 13 bits não sei como fica o de cima consegui porque o final so havia 1 xD

30. No endereço virtual 0x00008008 do processo em execução encontra-se:
    a. uma word do segmento de dados estáticos inicializados
    b. uma word do segmento de dados estáticos não inicializados
    c. uma word da stack
    d. o código máquina de uma instrução

Um sistema possui um espaço de endereçamento virtual de 4Gbytes, páginas de memória de 4KBytes e 1GByte de memória física. Considere tambem:

- Que num dado instante está a executar um processo cujo *Page Table Register* possui o valor 0x01230000
- Que cada entrada da *Page Table* possui 32 bits, está alinhada em endereços múltiplos de 4 e contém a seguinte informação e flags:

Valid, Dirty, Read, Write, Execute flags [31:27]	*Bits não usados [26:18]*	*PPN [17:0]*

Endereço	Valor
...	
0x0123000C	0xB000600C
0x01230010	0xF0005007
0x01230014	0xA8000005
0x01230018	0x9000000E
...	

29) Num acesso à memória, o CPU produz o endereço 0x0000600C. Qual o endereço físico que é traduzido?
a) 0x0000500C
b) 0x0000E00C
c) 0x00005007
d) Nenhuma das respostas está certa

30) O processo em execução pode aceder ao endereço virtual 0x00004014 para:
a) Leitura
b) Escrita
c) Leitura e Escrita
d) Nenhuma está certa

31) No caso de ser necessário substituir a página física que contém o endereço 0x00005001 por uma nova página, haverá necessidade de a salvaguardar previamente?
a) Não, porque a respectiva flag Dirty não está activa
b) Não, porque a respectiva flag Write não está activa
c) Sim, porque a respectiva flag Dirty está activa
d) Sim, porque a respectiva flag Write está activa

Resolução em baixo

men. virtual → 4Gb
pag. memória → 4Kb
men física → 16b

(29) 0x 0000600C.

• p.m → 4K = 2^{12} = 12 bits → 0x00C

• 00006 x 4 = 0000 0000 0000 0000 0110 x 4 = 0000 0000 0000 0000 1000

= 00018

• somar ao 0x01230000

= 0x01230018 $\xrightarrow{\text{tabela}}$ 0x9000 000E

→ end. físico = 0x0000E00C (b)

(30) 0x00004014

• 12 bit → 014

• 0000 0000 0000 0000 0100 x 4 = 00010

• soma 0x01230010 $\xrightarrow{\text{tabela}}$ 0xF0005007

1111

Ver esquema → primeira

valid

Dirty

Recs

write

(c)

(31) 0x00005001

12 bits → 001

0000 0101 x 4 = 0000 0001 0100 = 00014

• soma 0x01230000 = 0x01230014

↓ tabela

0xA000 000

↓

1010

↓

Como o dirty está a zero não é preciso salvaguardar

(a)

--
2020
--
1. Num sistema computacional com um espaço de endereçamento de memória de N bits, o conjunto de endereços possíveis será:

a. $[2_{N-1},..., 0]$
b. $[2_{N},..., 0]$
c. $[2^{N} - 1,..., 0]$
d. $[2^{N-1},..., 0]$

2. Dizer-se que, num sistema computacional, a memória apresenta uma organização do tipo byte-addressable significa que:

a. cada posição de memória é identificada com um endereço de 1 byte
b. o barramento de dados tem a dimensão de 1 byte
c. os barramentos de endereços e de dados tem obrigatoriamente a mesma dimensão
d. uma word de 32 bits é armazenada em 4 posições de memória consecutivas de 1 byte

não encontro nada sobre isto

3. No PIC32, os registos TRIS, LAT e PORT do PIC32 permitem, respetivamente:

a. o output e input de informação e o controlo da direcionalidade de um porto digital
b. o input e output de informação e o controlo da direcionalidade de um porto digital
c. o controlo da direcionalidade, o input e output de informação de um porto digital
d. o controlo da direcionalidade, o output e input de informação de um porto digital

4. Quando, num periférico, é usada a técnica entrada/saída de dados por software(programada):

a. o periférico faz um pedido de interrupção ao CPU após a conclusão da transferência de dados
b. o CPU verifica através de um ciclo de polling se o periférico está pronto para transferir os dados
c. o periférico faz um pedido de interrupção ao CPU quando estiver pronto para transferir os dados
d. o CPU interrompe a execução do programa para configurar o controlador de DMA que faz a propriamente dita

O CPU inicia e controla a transferência de informação:

- E/S programada (***programmed I/O***)
  - O CPU toma a iniciativa – aguarda se necessário, inicia e controla a transferência de informação (**POLLING**)

E/S programada

- O CPU tem que esperar que o periférico esteja disponível para a troca de informação. Essa espera é efetuada num ciclo de verificação da informação de status do periférico, designado por ***POLLING***
- Uma parte substancial do tempo de processamento do CPU pode ser desperdiçado no ciclo de *polling*
- É uma técnica apropriada quando a velocidade do dispositivo periférico não diminui drasticamente a capacidade de processamento do CPU
- O ***overhead*** deste método de transferência (i.e., o número de ciclos de relógio gastos pelo CPU em tarefas que não estão diretamente relacionadas com a transferência de informação – pode ser dada em %) depende do número de vezes que o ciclo de *polling* for executado
- Uma solução para eliminar o tempo perdido no ciclo de *polling* consiste na utilização da técnica de **E/S por interrupção**

5. A função de um bootloader num sistema baseado em microcontrolador é:

a. transferir o código executável para a memória do microcontrolador a partir de um sistema de desenvolvimento, permitindo a sua posterior execução
b. realizar a compilação do software de alto nível(i.e. C) e iniciar a sua execução após o reset
c. executar o software, e auxiliar no seu debug através da introdução de breakpoints, visualizar registos e de posições de memória
d. interagir com o cross-compiler para efeitos de debug da aplicação

apenas funções básicas de transferência e execução de um programa.

6. Para construir um módulo de memória SRAM de 512k x 8 bits, são necessários:

a. 4 circuitos de 64k x 8 bits
b. 8 circuitos de 64k x 4 bits
c. 8 circuitos de 64k x 8 bits
d. 16 circuitos de 64k x 8 bits

⑥ $512K \times 8 \rightarrow 2^9 \times 2^{10} \times 2^3 = 22$ bits

a.) $2^2 \times 2^6 \times 2^{10} \times 2^3 = 21$ bits

b) $2^3 \times 2^{16} \times 2^2 = 21$ bits

c) $2^3 \times 2^{16} \times 2^3 = 22$ bits

d) $2^4 \times 2^{16} \times 2^3 = 23$ bits

7. O diagrama temporal da figura do lado representa um ciclo de:

a. leitura de um dispositivo mapeado no espaço de endereçamento de memória
b. escrita num dispositivo mapeado no espaço de endereçamento de memória
c. escrita num dispositivo mapeado no espaço de endereçamento I/O
d. leitura de um dispositivo mapeado no espaço de endereçamento I/O

sei que é memória, não sei ver se é escrita ou leitura

8. Numa memória estática SRAM:

a. as células necessitam de refrescamento regular
b. cada célula de 1 bit é implementada com um mínimo de 6 transistores
c. o tempo de acesso aumenta com o valor do endereço
d. existe um único barramento para dados e endereços por forma a diminuir o número de pinos do dispositivo

Memória do tipo RAM (volátil)

- **SRAM – Static RAM**
  - Vantagens:
    - Rápida
    - Informação permanece até que a alimentação seja cortada
  - Inconvenientes:
    - Implementações típicas: 6 transistores / célula
    - Baixa densidade, elevada dissipação de potência
    - Custo/bit elevado

9. Numa transferência por DMA, em modo bloco, quando o controlador de DMA pretende dar início a uma transferência:

a. ativa o sinal *busreq*, iniciando a transferência logo que reconheça a ativação do sinal bus Grant
b. ativa o sinal *busreq* durante um número fixo de CPU que vai iniciar uma transferência de dados
c. gera uma interrupção sinalizando desse modo o CPU que vai iniciar uma transferência de dados
d. sinaliza o CPU através da linha *busreq,* que vai dar início a transferência e inicia-a de imediato. O sinal *busgrant* é utilizado pelo CPU para suspender a atividade do DMA

10. Alguns timers que estudou tem, como último andar do temporizador um divisor por dois. A finalidade desse divisor por dois é:

a. dividir por dois o período do sinal de saída do timer
b. assegurar um sinal de saída com um duty cycle de 50%
c. aumentar a resolução do timer
d. aumentar a precisão do timer

11. Numa RSI, o conjunto de instruções designado por "prólogo" destina-se, no essencial, a:

a. alterar a tabela de vetores de modo a impedir que novos pedidos de interrupção sejam atendidos
b. identificar a fonte de interrupção(nos casos em que tal é feito por hardware) e obter o endereço inicial da RSI
c. regressar ao programa interrompido reativando as interrupções
d. copiar para a stack o contexto do programa que foi interrompido pela interrupção, i.e., registos internos do CPU.

Processamento de interrupções pela RSI

- Ações gerais que devem ser implementadas na Rotina de Serviço à Interrupção (software):
  1. Salvaguarda do contexto do programa que foi interrompido: registos internos do CPU → memória (*stack*) ("**PRÓLOGO**")
  2. .. ações associadas ao processamento da interrupção..
  3. Reposição do contexto do programa interrompido: registos internos do CPU ← memória (*stack*) ("**EPILOGO**")
  4. Conclusão da RSI com a instrução de retorno (do tipo "Return From Interrupt / exception" – "**`eret`**" no caso do MIPS)

- **Latência da interrupção**: define-se como o tempo que decorre desde a ocorrência do evento que desencadeia a interrupção até à execução da primeira instrução "útil" da Rotina de Serviço à Interrupção (pontos 1 a 4 do slide anterior mais eventual conclusão de secção crítica do código, mais a salvaguarda do contexto)

12. Um barramento paralelo de tipo síncrono:

a. inclui um sinal de relógio e o protocolo de comunicação é síncrono ou semi-síncrono
b. não inclui um sinal de relógio e o protocolo de comunicação é “handshaken”
c. permite a ligação de uma grande variedade de dispositivos operando com relógios de diferentes frequências
d. assegura “*fairness*” no acesso ao barramento (quando há mais do que um master)

Barramentos "Multi-Master" – Conceitos

- **"Starvation"**:
  um elemento não tem acesso ao barramento devido a constantes pedidos provenientes de elementos de prioridade superior

- "**Fair**":
  um sistema de arbitragem que garante o acesso ao barramento por todos os elementos, evitando "starvation". Pode basear-se numa das seguintes técnicas:
  - Aumento da prioridade dos elementos que esperaram para além de um dado intervalo
  - Diminuição da prioridade dos elementos já servidos
  - Redistribuição das prioridades de forma regular ou aleatória
  - Não aceitação de novos pedidos até os pendentes serem satisfeitos

  ("**Fairness**" - até um dispositivo com a mais baixa prioridade não deverá ficar impedido de aceder ao barramento)

--página que se vê mal pode conter erros

13. Diz-se que um barramento de comunicação série usa comunicação assincrona quando:

a. o relógio não é explicitamente transmitido e por isso, o relógio do receptor é recuperado o ponto das transições de nível lógico na linha de dados
b. não é enviado o relógio na transmissão, nem há recuperação do relógio na receção
c. o módulo de recessão pode gerar um sinal para atrasar o envio de dados do emissor
d. o sinal de relógio é codificado nos dados xxxinerxx??(f clocking)?

14. O barramento SPI é caracterizado por:

a. ter uma arquitetura master-slave com ligação ponto a ponto e comunicação **full duplex**
b. ter uma arquitetura master-slave com ligação multi-ponto e comunicação **full duplex**
c. ter uma arquitetura multi-master com ligação multi-ponto e comunicação **half duplex**
d. ter uma arquitetura multi-master com ligação ponto a ponto e comunicação **half duplex**

Descrição geral

- Arquitetura "Master-Slave" com ligação ponto a ponto
- Comunicação bidirecional "full-duplex"
- Comunicação síncrona (relógio explícito do *master*)
  - Relógio é gerado pelo *master* que o disponibiliza para todos os *slaves*
  - Não é exigida precisão ao relógio - os bits vão sendo transferidos a cada transição de relógio. Isto permite utilizar um oscilador de baixo custo no *master* (não é necessário um cristal de quartzo)
- Fácil de implementar por hardware ou por software
- Não são necessários "line drivers" (ou "transceivers") - circuitos de adaptação ao meio de transmissão

15. Na interface de comunicação I2C o primeiro byte de uma comunicação contém a seguinte informação:

a. o endereço do master , o endereço do slave com o qual este quer comunicar e a operação a realizar
b. os parâmetros de comunicação a utilizar (taxa de transferência e número total de bits a transferir)
c. o endereço do master e a operação a realizar
d. o endereço do slave e a operação a realizar

- O primeiro byte transmitido pelo *master* contém:
  - 7 bits: **endereço do *slave***
  - 1 bit: qualificação da operação (RD / WR\)

- Qualificador da operação:
  - **RD/WR\ = 0**: o *master* é o **transmissor** (escreve dados na linha SDA)
  - **RD/WR\ = 1**: o *master* é o **recetor** (lê dados da linha SDA)

- Cada *slave* lê o endereço da linha SDA; se o endereço lido coincide com o seu próprio endereço:
  - comuta para o estado "transmissor" se o bit RD/WR\ for igual a "1"
  - comuta para o estado "recetor" se o bit RD/WR\ for igual a "0"

16. Na interface RS-232C o desvio de frequência entre os sinais de relógio do transmissor e recetor pode originar um erro, no instante de amostragem dos vários bits. Esse erro cresce de forma:

a. inversamente proporcional ao fator de sobre-amostragem
b. diretamente proporcional ao número de stop-bits
c. diretamente proporcional à dimensão da trama
d. inversamente proporcional a taxa de transmissão

17. No protocolo USB, para transferir grandes quantidades de dados assegurando um valor de latência máximo e largura de banda requerida são usadas:

a. transferências de controlo
b. transferências isócronas
c. transferências "bulk"
d. transferências de interrupção

18. As principais características/objetivos de um device driver são:

a. permitir o uso de buffers circulares entre o CPU e o dispositivo periférico
b. permitir um acesso mais rápido a um dispositivo periférico através de implementação de rotinas de serviço a interrupção
c. criar um nível de abstração, uniformização de acesso e independência entre aplicações/SO e o hardware do dispositivo periférico
d. descrever ao programador o modelo de programação de um dispositivo periférico

Introdução

- As aplicações/Sistemas Operativos (SO) **não podem conhecer todos os tipos de dispositivos** passados, atuais e futuros com um nível de detalhe suficiente para realizar o seu controlo a baixo nível!
- **Solução**: Criar uma camada de abstração que permita o acesso ao dispositivo de forma independente da sua implementação
- ***Device driver***
  - Um programa que permite a outro programa (aplicação, SO) interagir com um dado dispositivo de hardware
  - Implementa a camada de abstração e lida com as particularidades do dispositivo controlado
  - Como o *Device Driver* tem de lidar com os aspetos específicos da implementação física, o seu fornecimento é sempre assegurado pelo fabricante
- **Aspetos-chave:**
  - Abstração, uniformização de acesso, independência entre aplicações/SO e o hardware

19. Num barramento CAN:

a. cada tipo de mensagem tem um id único
b. cada dispositivo ligado ao barramento tem um id único
c. um master pode enviar duas tramas simultâneas para dois slaves, desde que os ids das tramas seja único
d. uma mensagem produzida por um master pode ser recebida apenas por um e um só slave

Introdução

- O CAN é um barramento "multi-master": qualquer nó do barramento pode produzir informação e iniciar uma transmissão
- Uma vez que dois nós podem querer aceder simultaneamente ao barramento para transmitir, tem que haver uma forma de arbitrar o acesso
- Comunicação bidirecional "half-duplex"
- A informação produzida é encapsulada em tramas
- Transmissão em "broadcast": um transmissor pode enviar informação para todos os nós ao mesmo tempo
- No CAN cada mensagem tem um ID único; esse ID determina a prioridade da mensagem e, consequentemente, a prioridade no acesso ao barramento

20. O número de comparadores necessários para a implementação de uma cache de 16kB, que usa uma arquitetura de <u>mapeamento direto,</u> blocos de 64 bytes e 256 linhas:

a. 256
b. 16
c. 1
d. 64

21. Num sistema com memória virtual, o valid bit de uma entrada de uma page table, quando **não ativo** significa que:

a. a respetiva página foi acedida recentemente
b. a respetiva página está residente em memória física
c. a informação da respetiva página foi alterada
d. a respetiva página não está residente em memória física

22. Numa memória virtual, o TLB(translation-Lookaside Buffer):

a. armazena a totalidade das entradas da page table que tenham o valid bit a '1'
b. destina-se a acelerar a tradução de endereços virtuais em endereços físicos
c. é normalmente baseado numa arquitetura do tipo mapeamento direto
d. serve para obter de forma mais rápida o conteúdo de uma VPN na Page Table

Translation-lookaside buffer (TLB)

- Os acessos à memória virtual contêm localidade espacial e temporal
- Uma forma de resolver o problema é ter uma parte da *Page Table* numa memória rápida, semelhante a uma cache, onde se encontram as entradas da tabela mais recentemente utilizadas
- Esta memória é normalmente designada por "translation-lookaside buffer" - TLB
- A TLB é tipicamente implementada como uma memória associativa com procura paralela
- Páginas residentes em disco não são referenciadas na TLB

Translation-lookaside buffer (TLB)

- Em cada acesso, no processo de tradução, verifica-se se o número da página virtual (VPN) está na TLB
- Se VPN está na TLB: "Hit" – o endereço da página física é obtido de imediato da TLB
- Se VPN não está na TLB: "Miss" – a *Page Table* é verificada e pode originar, ou não, um *Page Fault.*
  - Se o acesso à *Page Table* não originar um *Page Fault*, o endereço físico e os atributos da página são copiados da *Page Table* para a TLB (pode envolver a substituição de uma entrada na TLB)
  - Se o acesso à *Page Table* originar um *Page Fault*, é necessário fazer todo o processamento do *Page Fault* que culmina com a atualização da *Page Table*. A seguir o endereço físico e os atributos da página são copiados da *Page Table* para a TLB (pode envolver a substituição de uma entrada na TLB)

Grupo II

23. Considere um timer tipo 2 como o da figura abaixo(semelhante aos dos PIC32) com a sua saída ligada a uma unidade OC. Admita que a frequência do relógio TCLK é de 20MHz e que a frequência no pino O1 é de 200Hz. Assuma ainda que o fator de divisão prescaler é o menor que for possível para estas condições. Para obter na saída O1 um sinal periódico com uma frequência de 200MHz e um duty cycle de 25%

a. os valores armazenados respectivamente em PR e OCK devem ser 49999 e 12500
b. os valores armazenados respectivamente em PR e OCK devem ser 12499 e 3125
c. os valores armazenados respectivamente em PR e OCK devem ser 24999 e 6250
d. os valores armazenados respectivamente em PR e OCK devem ser 12499 e 9375

i) calcular prescaler

$2^{16}-1$ → nº possibilidades +1 na formula → nº maior possível

$$\text{fator prescaler} = \frac{\frac{20MHz}{2^{16}}}{200\ Hz} = 1,52$$

ii) outra maneira de calcular o prescaler é tentativas

$\frac{20MHz}{200} = 100\,000 - 1 = 99999$ > nº maior logo X

↳ pg é síncrono

1 ↑ possível prescaler

$\frac{100\,000}{2} - 1 = 49999$ < nº maior logo ✓

2 ↑ possível prescaler → este é o prescaler

$50000 \times 0,25 = 12500$

↓ o resultado do PR+1

↓ duty cycle

PR = 49999

OCK = 12500

$\frac{50000}{4} = 50\,000$ | 4 → 12500

10
20
00

24. Suponha que pretende implementar um circuito gerador de sinais de seleção programável (semelhante ao que estudou nas aulas teóricas e ilustrado na figura abaixo) que permita gerar 64 linhas de seleção, cada uma delas ativa em 32k endereços consecutivos, num espaço de endereçamento de 32 bits. Ao Bloco 1 da figura devem ser ligado N bits, correspondendo a gama:

a. A14 a A0
b. A20 a A0
c. A31 a A21
d. A31 a A15

64 linhas de seleção bloco 2 → n → 2^n

$2^n = 64$ logo $n = \boxed{6 \text{ bits}}$ → dimensão do R

$32K = 2^{15}$ → $M = 15$ → estes são os bit ⊖ significativos

$15 + 6 = 21$ bits usados

$32 - 21 = 11$ bits

↳ $N = 11$ que são os ⊕ significativos

logo, A31 a A21

25. Considere um controlador DMA de 32 bits <u>não dedicado</u> , a funcionar em modo cycle-stealing, em que um bus cycle é realizado em 3 ciclos de relógio e o tempo mínimo entre operações elementares é de 2 ciclos de relógio. Se a frequência de funcionamento do DMA for 250MHz, o tempo mínimo necessário para transferir 500 palavras de 32 bits é, aproximadamente:

a. 16µs
b. 20µs
c. 500ns
d. 25µs

não dedicado → 2 ciclos

i) 3 ciclos + 2 ciclo + 3 ciclos + 2 ciclo = 10 ciclos
pedir — espera — repete pq é não dedicado

ii) $T = \frac{1}{f} = \frac{1}{250 \times 10^6} = 0{,}004 \times 10^{-6} = 4 \times 10^{-9} = 4\,ns$

iii) $t_{transferência\ para\ 1\ palavra} = 4\,ns \times 10 \text{ ciclos} = 40\,ns$

iv) $t_{para\ 500\ palavras} = 40 \times 500 = 20\,000\,ns \simeq 20\,\mu s$

26. Considere um sistema baseado num CPU a funcionar a uma frequência de 200 MHz com uma taxa de execução média de 50 MIPS ($50x10^6$ instruções por segundo, CPI = 4) que processa, por interrupção, eventos externos periódicos. Se o overhead total do atendimento a à interrupção for de 160 ciclos de relógio, e a rotina de serviço a interrupção tiver 60 instruções, a máxima frequência a que esses eventos podem ocorrer é aproximadamente:

a. 50MHz
b. 2MHz
c. 909 kHz
d. 500kHz

27. Considere um espaço de endereçamento de 32 bits e uma memória cache, parcialmente associativa capacidade de 64 kByte, associatividade de 8 e blocos de 64 bytes. O número de bits do do campo "group" do endereço é:

a. 10
b. 7
c. 6
d. 5

cache → 64K/8 = 8K = 2^{13} → 13 bits
blocos → 64 → 2^6 = 6 bits para byte
& set/group → 13 − 6 = 7 bits
tag → 32 − 13 = 19 bits

28. Suponha um sistema de memória organizada em 2 níveis que responde a um acesso a uma posição de memória que se encontra na cache em 1 ciclo de relógio e a um acesso a uma posição de memória que se encontra na memória principal em 51 ciclos de relógio. Para que o tempo médio de acesso ao sistema de memória seja 11 ciclos de relógio o hit ratio da cache deverá ser, aproximadamente:

a. 85%
b. 90%
c. 75%
d. 80%

1 ciclo a fazer acesso se tiver na cache (hit)
51 ciclos se não tiver na cache (fazer copia e aceder) (miss)

hitRatio × 1 + (1 − hitRatio) × 51 = 11 ⟺
hitRatio + 51 − 51hitRatio = 11 ⟺
−50 hitRatio = −40 ⟺ hitRatio = $\frac{4}{5}$ = 0,8 = 80%

29. Um sistema computacional com uma memória byte-addressable, possui um espaço de endereçamento virtual de 4G, um espaço de endereçamento físico de 1G e páginas de memória de 4 kBytes. Admita ainda que o Page Table Register tem o valor 0x3FF28000, que cada entrada da Page Table está alinhada em endereços múltiplos de 4 e que o seu conteúdo tem 32 bits. Quando é gerado o endereço virtual 0x00006058, o endereço da Page Table acedido para obter a tradução para o endereço físico será:

a. 0x3FF28058
b. 0x3FF28006
c. 0x3FF28018
d. 0x3FF28000

pag. memória = 4 kbytes = 2^{12} = 12 bits

end. virtual → 0x00006058
12 bits

PNN = 00006 x 4 = shift 2

000001 1000 → 018

Soma → 0x3FF28000
= 0x3FF28018

c

30. Admita agora que, no sistema descrito na questão 29, o conteúdo de cada entrada da Page Table tem o formato descrito na figura abaixo. Quando o CPU acede ao endereço virtual 0x000C80FC para ler uma instrução, o conteúdo da page table para esse endereço é 0xA7FE2030. Nesse caso , o gestor de memória:

Valid, Dirty, Read, Write, Execute [31:27]	Bits não usados [26:18]	PPN [17:0]

a. traduz o endereço virtual para o endereço físico de acordo com a informação da page table
b. não traduz o endereço virtual para o endereço físico e gera uma exceção porque a página respetiva não tem permissão de escrita
c. não traduz o endereço virtual para o endereço físico e gera uma exceção porque a página respetiva não tem permissão de execução
d. não traduz o endereço virtual para o endereço físico e gera uma exceção porque a página respetiva não tem o dirty bit ativo

A7... - 10100111...