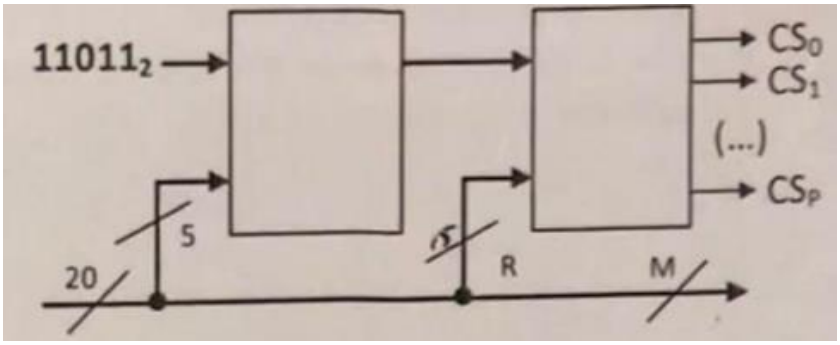
1. A função de um *bootloader* num sistema baseado em microcontrolador é:
    a. Realizar a compilação do software de alto nível (e.g. C) e iniciar a sua execução após o *reset* do sistema
    b. Executar o software e auxiliar no seu *debug* através da introdução de *breakpoints*, visualização do conteúdo de registos e de posições de memória
    c. Transferir o código executável a partir do sistema *host* usado no desenvolvimento, para a memória do microcontrolador, permitindo a sua posterior execução
    d. Interagir com o *cross-compiler* para efeitos de *debug* da aplicação
2. Um microcontrolador é um sistema computacional programável que:
    a. Disponibiliza, através dos seus portos de I/O, a generalidade dos sinais dos barramentos do microprocessador para ligação direta a sensores e atuadores de um sistema embutido
    b. Inclui, num único dispositivo integrado, CPU, memória e u conjunto variável de periféricos e portos de I/O
    c. Devido a restrições de custos, suporta um número reduzido de instruções e de registos
    d. Por questões de dimensão, não utiliza mecanismos de multiplexagem para partilha de portos de I/O....diversas funcionalidades internas
3. Quando nos referimos a um "Módulo de I/O" estamos a referir-nos:
    a. A um periférico que permite operações de escrita e leitura
    b. Ao sofware (*device-driver*) que assegura que o acesso ao periférico é transparente para as aplicações
    c. Ao tipo de conector que permite a interface entre um periférico e o canal de comunicação do mesmo com o mundo exterior
    d. À parte de um dispositivo periférico que funciona como adaptador entre as características intrínsecas do periférico e as características do CPU e do sistema de memória
4. O modelo de programação de um periférico especifica:
    a. O sub-conjunto de instruções *assembly* do CPU suportadas por esse periférico
    b. Quais 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. A funcionalidade do periférico e o seu conjunto de registos de dados, de controlo e de status

6. Na implementação da parte de dados de um porto de entrada:
    a. Devem ser usados *buffers tri-state* para que a informação presente no barramento de dados só fique disponível para o periférico quando o porto for ativado
    b. Devem ser usados *buffers tri-state* para que a informação só seja colocada no barramento de dados quando o porto for selecionado
    c. Devem ser usados *flip-flops* para armazenar o valor presente no barramento de endereços, se este coincidir com o endereço do porto
    d. Devem ser usados *flip-flops* para armazenar o valor transferido através do barramento de dados durante um ciclo de escrita
7. Suponha que os bits 7 e 6 do porto B do PIC32 estão configurados como saída e que se pretende colocar esses dois bits com o valor "**10**", respetivamente, sem alterar o valor dos restantes. Para isso, em linguagem C, pode fazer-se:
    a. LATB = LATB & 0xFFBF;
    b. LATB = (LATB & 0xFF3F) | 1 << 7;
    c. LATB = LATB | 0x0080;
    d. LATB = LATB & 0x0080;
8. O método de transferência de informação entre um CPU e um módulo de E/S, em que o programa executado no CPU inicia, monitoriza e controla a transferência de informação, designa-se por:
    a. Entrada/saída por *polling* com vectorização
    b. Entrada/saída por interrupção iniciada pelo CPU
    c. Entrada/saída programada (método *polling*)
    d. Entrada/saída por interrupção iniciada pelo periférico
9. Na organização do sistema de interrupções designada por "interrupções vetorizadas", o processador identifica o periférico gerador da interrupção:
    a. 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
    b. Por hardware num ciclo de *interrupt acknowledge* durante o qual o periférico gerador da interrupção coloca o vetor no barramento de dados
    c. Por software, antes de chamar a rotina de serviço à interrupção lendo o registo de estado de cada um dos periféricos do sistema
    d. Por software na rotina de serviço à interrupção lendo o registo de estado de cada um dos periféricos do sistema
10. Numa transferência por DMA, o mecanismo de interrupção é utilizado pelo respetivo controlador para:
    a. Sinalizar/Informar o CPU que a transferência de informação foi completada
    b. Efetuar ao CPU o pedido de cedência dos barramentos; a transferência tem início quando o DMA receber a confirmação, através do sinal *busgrant*, de que os barramentos foram libertados
    c. Informar o CPU que a transferência de informação vai ter início, permitindo desse modo que o CPU suspenda a atividade de acesso ao exterior
    d. Informar o CPU da existência de uma anomalia ocorrida durante o processo de transferência

12. Num barramento série que use a técnica de sincronização de relógio designada por “relógio codificado”:
    a. O relógio é gerado pelo recetor e enviado de forma codificada para o transmissor
    b. O relógio é gerado pelo transmissor que o envia, para o recetor, codificado nos dados
    c. O transmissor e o recetor têm os seus próprios relógios que se sincronizam mutuamente
    d. O transmissor e o recetor têm relógios independentes; o relógio do recetor é sincronizado ocasionalmente com o do transmissor por meio da receção de símbolos codificados nos dados
13. No método de sincronização dos relógios utilizado na interface RS-232C, o “erro de fase”:
    a. Introduz um desvio constante entre o instante ideal e o instante real de amostragem no recetor
    b. Introduz um desvio cumulativo e proporcional ao comprimento da trama, entre o instante ideal e o instante real de amostragem no recetor
    c. Não tem qualquer implicação no instante real de amostragem no recetor
    d. É proporcional à frequência do relógio do transmissor
14. O barramento SPI é caracterizado por ter uma arquitetura:
    a. *Master-slave* com ligação multi ponto e comunicação *full duplex*
    b. *Master-slave* com ligação ponto a ponto e comunicação *half duplex*
    c. *Multi-master* com ligação ponto a ponto e comunicação *full duplex*
    d. *Master-slave* com ligação ponto a ponto e comunicação *full duplex*
15. Na interface de comunicação I^2C, quando existe mais do que um *master* a tentar aceder simultaneamente ao barramento, a arbitragem é feita:
    a. Por bit dominante / bit recessivo e processa-se bit a bit
    b. Atribuindo o barramento ao *master* com o endereço mais alto
    c. Dando prioridade ao *master* que foi servido pela última vez há mais tempo
    d. Através do bit que determina a operação a efetuar
16. Suponha que no barramento CAN, após uma situação de meio livre, três *masters* acedem simultaneamente ao barramento. O *master* 1 produz uma mensagem com o identificador 0x34, o *master* 2 produz uma mensagem com o identificador 0x15 e o *master* 3 produz uma mensagem com o identificador 0x5A. Nessa situação:
    a. O acesso ao barramento é ganho pelo *master* 1
    b. O acesso ao barramento é ganho pelo *master* 3
    c. O acesso ao barramento é ganho pelo *master* 2
    d. Não há necessidade de arbitrar o acesso ao barramento, porque as mensagens produzidas pelos 3 *masters* têm identificadores diferentes
17. No barramento CAN a codificação das tramas de dados utiliza a técnica de *bit stuffing*. Essa técnica consiste em:
    a. Por cada 5 bits iguais é inserido um de polaridade oposta
    b. Por cada 5 bits com o valor ‘1’, é inserido um bit com o valor ‘0’
    c. Por cada bit enviado é inserido um de polaridade oposta
    d. Por cada 5 bits com o valor ‘0’, é inserido um bit com o valor ‘1’

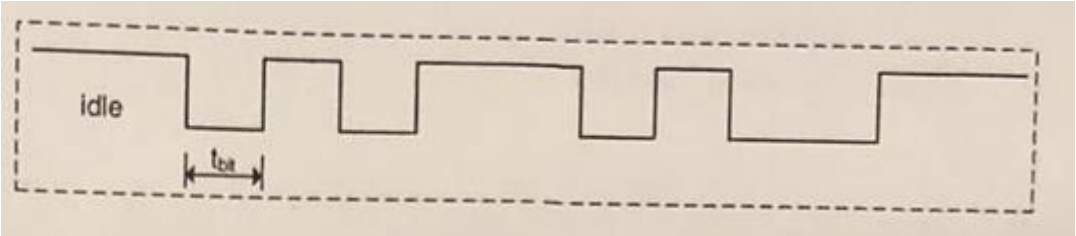
19. Para construir um módulo de memória SRAM de 256k x 16 bits, são necessários:
    a. 8 circuitos de 32k x 8 bits
    b. 16 circuitos de 16k x 8 bits
    c. 32 circuitos de 16k x 4 bits
    d. 32 circuitos de 32k x 4 bits
20. Uma memória dinâmica (tecnologia DRAM) de 4Gx64, implementada com matrizes quadradas, o número de bits do barramento de endereços é:
    a. 16
    b. 6
    c. 64
    d. 32
21. Considere uma memória DRAM em que o *address(high)* é composto por 5 bits, o *address(low)* é composto por 7 bits e a memória é composta por 8 planos de células de 1 bit. Neste caso estamos perante uma memória de:
    a. 4 kbit com matrizes de 128 linhas por 128 colunas
    b. 32 kbit com matrizes de 256 linhas por 128 colunas
    c. 4 kByte com matrizes de 128 linhas por 32 colunas
    d. 4 kByte com matrizes de 32 linhas por 128 colunas
22. Num determinado sistema com um espaço de endereçamento de 16 bis foi implementado um descodificador de endereços usando a seguinte expressão lógica (lógica negativa): **CE\ = A15 + A14 + A12**. Com este descodificador pode ser selecionada uma memória:
    a. De 4k posições, na gama 0x3000 a 0x3FFF
    b. De 4k posições, na gama 0xC000 a 0xCFFF
    c. De 2k posições, na gama 0xE000 a 0xE7FF
    d. De 8k posições, na gama 0xC000 a 0xDFFF
23. O número de comparadores necessário para a implementação de uma cache parcialmente associativa de 64 kB de 4 vias e blocos de 64 bytes é
    a. 1
    b. 8
    c. 16
    d. 4
24. O *dirty bit* é usado numa *cache* quando esta usa uma política de escrita:
    a. *Write-back*, para indicar que o respetivo bloco não está a ser usado
    b. *Write-back*, para indicar que a informação armazenada no respetivo bloco foi alterada
    c. *Write-through*, para indicar que a informação armazenada no respetivo bloco foi alterada na memória principal
    d. *Write-through* para indicar que o respetivo bloco não está a ser usado

26. Na técnica designada por "memória virtual", o endereço da memória física é obtido através:
    a. Da tradução do *physical page number* no *virtual page number* e sua concatenação com o *page offset* do endereço produzido pelo CPU
    b. Da tradução do *virtual page offset* no *physical page offset* e sua concatenação com o *virtual page number* do endereço produzido pelo CPU
    c. Da tradução do *physical page offset* no *virtual page offset* e sua concatenação com o *virtual page number* do endereço produzido pelo CPU
    d. Da tradução do *virtual page number* no *physical page number* e sua concatenação com o *page offset* do endereço produzido pelo CPU
27. Considere um espaço de endereçamento de 20 bits e o circuito gerador de sinais de seleção programável da figura (igual ao que estudou nas aulas teóricas). Na situação apresentada e considerando que a linha de seleção **CS_1** está ativa na gama **0xD8400** a **0xD87FF**, podemos concluir que este circuito gera:
    a. 16 linhas de seleção
    b. 64 linhas de seleção
    c. 32 linhas de seleção
    d. 8 linhas de seleção



28. Considere uma memória DRAM de 1Mx8, implementada com matrizes quadradas, que utiliza um ciclo de refrescamento do tipo *RAS only*. Sabendo que o parâmetro *cycle time* do ciclo *RAS only* é 40 ns, o tempo necessário para fazer um refrescamento completo à memória é, aproximadamente:
    a. 41 µs
    b. 328 µs
    c. 320 ms
    d. 40 ns
29. Considere um *timer* em que a relação entre as frequências de entrada e de saída é uma constante "**k**" configurável. Considere ainda que se usaram dois desses timers e se ligaram em cascata (i.e. em série). Supondo que a frequência à entrada do primeiro *timer* é 1 MHz, para obter à saída do segundo *timer* uma frequência de 200 Hz, as constantes dos dois timers, "**k1**" e "**k2**", poderão ter os seguintes valores:
    a. K1 = 200, K2 = 250
    b. K1 = 1000, K2 = 200
    c. K1 = 200, K2 = 25
    d. K1 = 3500, K2 = 1500

30. Considere um sistema de supervisão, baseado no protocolo I2C, que recolhe periodicamente informação proveniente de 30 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 50kHz, necessita para adquirir os valores de todos os sensores (cada um implementado num *slave* distinto) é:
    a. 20 µs
    b. 12 ms
    c. 0.6 ms
    d. 6.6 ms
31. Um dispositivo com interface RS232C, configurado para transmitir 7 bits de dados, paridade par e 2 stop bits, produz a trama da figura que é recebida por outro dispositivo RS232C incorretamente configurado para 8 bits de dados, paridade ímpar e 1 stop bit, mas com o mesmo *baudrate*. Para a trama apresentada, o recetor:
    a. Não vai detetar qualquer erro, mas o valor recebido não é igual ao valor transmitido
    b. Vai detetar um erro de *framming*
    c. Não vai detetar qualquer erro e recebe corretamente o valor transmitido
    d. Vai detetar um erro de paridade



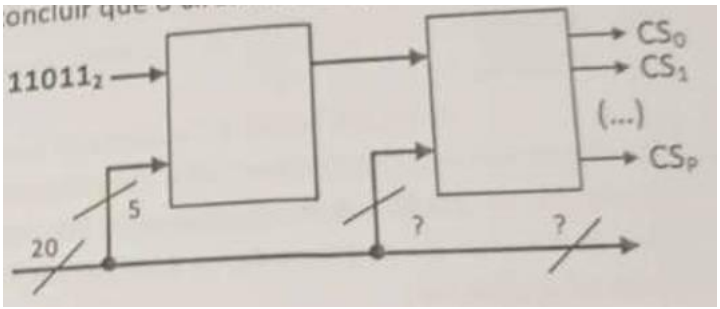
32. Considere um controlador de DMA não dedicado de 32 bits (i.e., com barramento de dados de 32 bits), a funcionar a 120 MHz. Suponha que são necessários 2 ciclos de relógio (= 1 T_{BC}) para efetuar uma operação de leitura ou escrita. A taxa de transferência de pico desse DMA (expressa em Bytes/s), em modo *cycle-stealing* e com um tempo mínimo entre operações elementares de 2 T_{BC} é:
    a. 120 MByte/s
    b. 12.5 MByte/s
    c. 40 MByte/s
    d. 60 MByte/s
33. Considere um espaço de endereçamento de 32 bits e uma memória *cache* de 128 kByte, com uma organização parcialmente associativa com 8 vias, e blocos de 32 bytes. O número de linhas de *cache* é:
    a. 8
    b. 1024
    c. 32
    d. 512

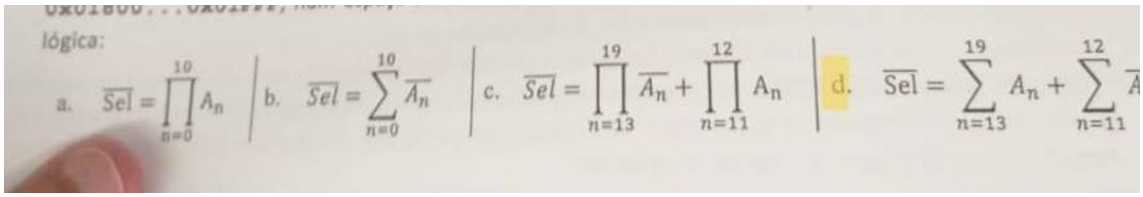
35. Um sistema computacional com um sistema de memória *byte-addressable*, possui um espaço de endereçamento virtual de 16M e um espaço de endereçamento físico de 2M. Sabendo que a *page table* tem 8k entradas, podemos concluir que o sistema de memória virtual está organizado em páginas de:
    a. 1 kByte
    b. 2 kBytes
    c. 4 kBytes
    d. 512 Bytes
36. 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. ..... apenas pode ser efetuado por instruções que transferem 1 byte de informação.....
    c. Cada *word* de 32 bits é armazenada em 4 posições de memória consecutivas de 1 byte.....
    d. O barramento de endereços e de dados têm obrigatoriamente que ter a mesma dimensão.....
37. Na arquitetura de um sistema computacional, o *Address Bus* permite:
    a. Especificar o tipo de operação efetuada sobre a memória ou sobre o periférico
    b. Identificar, na memória ou num periférico, a origem/destino da informação....
    c. Transferir dados entre.....
    d. .....
38. O *bus matrix*, usado no PIC32, permite:
    a. O acesso do CPU a uma memória RAM para transferência simultânea de dados e instruções
    b. O acesso do CPU a uma memória FLASH para transferência simultânea de dados e instruções.
    c. A transferência direta de dados ou instruções da memória RAM para a FLASH (ou o contrário) sem intervenção de qualquer outro dispositivo
    d. O acesso do CPU a uma memória FLASH para leitura de dados constantes e à mesma memória FLASH para leitura de instruções
39. Um compilador-cruzado (*cross-compiler*) é um programa que corre numa plataforma e:
    a. Simula o funcionamento de uma aplicação numa plataforma diferente
    b. Permite o *debug* de uma aplicação que corre numa plataforma diferente
    c. Gera código que pode ser executado na mesma plataforma em que é gerado
    d. Gera código para uma plataforma com uma arquitetura diferente daquela onde é executado

41. Na implementação da parte de dados de um porto de saída devem ser usados:
    a. Buffers tri-state para que a informação presente no barramento de dados só fique disponível para o periférico quando o porto for ativado
    b. Flip-flops para armazenar o valor transferido através do barramento de dados durante um ciclo de escrita
    c. Buffers tri-state para que a informação só seja colocada no barramento de dados quando o porto for selecionado
    d. Flip-flops para armazenar o valor presente no barramento de endereços, se este coincidir com o endereço do porto
42. A descodificação de endereços consiste em:
    a. Representar um endereço em binário de forma a utilizar o menor número de linhas do barramento
    b. Determinar em função do endereço gerado pelo periférico, qual o CPU ou memória que deve ser selecionada
    c. Ocupar a totalidade do espaço de endereçamento do processador com memórias e periféricos
    d. Determinar, em função do endereço presento no barramento, qual o periférico ou memória que deve ser selecionada
43. Suponha que os bits 7 e 6 do porto B do PIC32 estão configurados como saída e que se pretende atribuir a esses dois portos o valor 1 e 0, respetivamente, sem alterar o valor dos restantes. Para isso, em linguagem C, pode fazer-se:
    a. LATB = (LATB & 0xFF3F) | (1 << 7);
    b. LATB = (LATB & 0xFFBF) | (0 << 6);
    c. LATB = (LATB | 0x0080) & (0 << 6);
    d. LATB = (LATB & 0x0000) | (1 << 7);
44. Na implementação de um porto de I/O do PIC32, o registo com a designação "PORT" está associado a um conjunto de dois *flip-flops* D ligados em série (*shift register* de dois andares). O objetivo desta organização é:
    a. Criar um atraso temporal de dois ciclos de relógio relativamente às alterações do registo TRIS
    b. Assegurar que o registo LAT é adequadamente escrito numa operação "Read Modify Write"
    c. Assegurar que há uma adequada sincronização entre o valor lógico presente na entrada e o relógio interno e prevenir ocorrência, internamente, de fenómenos de meta-estabilidade
    d. Garantir que há tempo para colocar em alta impedância o *buffer tri-state* de saída quando se comuta o porto de modo de saída para modo de entrada
45. O método de transferência de informação entre um CPU e um periférico, em que o programa executado no CPU inicia, monitoriza e controla a transferência de informação, designa-se por:
    a. Entrada/saída por *polling* com vectorização
    b. Entrada/saída por interrupção iniciada pelo CPU
    c. Entrada/saída por interrupção iniciada pelo periférico
    d. Entrada/saída programada (método de *polling*)

47. Num sistema que implemente "interrupções vetorizadas", a sequência de operações efetuada pelo CPU na fase de atendimento a uma interrupção é, pela ordem indicada, a seguinte:
    a. Salto para a RSI, identificação da fonte, determinação do endereço da RSI, salvaguarda do endereço de retorno
    b. Determinação do endereço da RSI, identificação da fonte, salvaguarda do endereço de retorno, salto para a RSI
    c. Salvaguarda do endereço de retorno, identificação da fonte, determinação do endereço da RSI, salto para a RSI
    d. Salvaguarda do endereço de retorno, salto para a RSI, identificação da fonte
48. ....., o conjunto de instruções designado por "prólogo" destina-se, no essencial, a:
    a. ....a tabela de vetores de modo a impedir que novos pedidos de interrupção sejam atendidos
    b. Salvaguardar, na stack, o contexto atual do programa interrompido, i.e, registos internos do CPU
    c. Identificar a fonte de interrupção (quando isso é feito por *software*) e obter o endereço inicial da RSI
    d. Regressar ao programa ....
49. Na organização do sistema de interrupções designada por "identificação da fonte por software", o processador identifica o periférico gerador da interrupção:
    a. 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
    b. Antes de saltar para a rotina de serviço à interrupção lendo o registo de estado de cada um dos periféricos do sistema
    c. Na rotina de serviço à interrupção lendo o registo de estado de cada um dos periféricos do sistema
    d. Num ciclo de *interrupt acknowledge* durante o qual o periférico gerador da interrupção coloca o vetor no barramento de dados
50. Quando nos referimos a uma "**secção crítica**", num trecho de código executado por um CPU, estamos a referir-nos:
    a. A uma sequência de instruções cuja execução não pode ser interrompida por uma interrupção, por usar/atualizar um recurso partilhado com o código da rotina de serviço à interrupção
    b. À secção mais importante do código em execução e que não pode deixar de ser executado
    c. A um conjunto consecutivo de instruções onde não é possível chamar uma função
    d. A uma secção de código que, em certas circunstâncias, pode gerar uma exceção

52. Quando é usada a técnica de transferência de dados por DMA para transferir informação entre a memória e um periférico:
    a. O CPU configura o controlador de DMA com os endereços de origem e destino e o número de *words* a transferir, e o DMA faz depois a transferência dos dados
    b. O periférico faz um pedido de interrupção ao controlador de DMA após a conclusão da transferência de dados
    c. O CPU verifica, através de um ciclo de *polling* ao registo de estado do controlador de DMA, se a transferência já foi concluída
    d. O DMA verifica, através de um ciclo de *polling* ao registo de estado do periférico, se existem mais dados para serem transferidos
53. 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, e inicia de seguida a transferência
    b. Ativa o sinal *busreq*, efetuando a transferência logo que se torne o *bus master*
    c. ...uma interrupção que é interpretada pelo CPU como um pedido de cedência dos barramentos e a transferência é efetuada quando o DMA reconhecer a ativação do sinal *busgrant*
    d. O *busgrant* é utilizado pelo CPU para suspender a atividade do DMA
54. Num sistema de comunicação série que use transmissão síncrona:
    a. O sinal de relógio não é transmitido, nem há recuperação do relógio no recetor
    b. O sinal de relógio é codificado nos dados, ou é transmitido de forma explícita através de um sinal adicional
    c. Os relógios do transmissor e do recetor não precisam de ser sincronizados
    d. Existe obrigatoriamente, para além da linha de dados, uma linha através da qual é transmitido o sinal de relógio
55. Considere um espaço de endereçamento de 20 bits e o circuito gerador de sinais de seleção programável da figura (igual ao que estudou nas aulas teóricas). Na situação apresentada e considerando que a linha de seleção **CS_1** está ativa na gama **0xD8200** a **0xD83FF**, podemos concluir que o circuito da figura gera:
    a. 16 linhas de seleção
    b. 64 linhas de seleção
    c. 8 linhas de seleção
    d. 32 linhas de seleção



57. O sinal de seleção "**Sel**" (lógica negativa) de uma memória de 2k endereços mapeada na gama de endereços 0x01800…0x01FFF, num espaço de endereçamento de 20 bits, pode ser obtido através da expressão lógica:
    a. 

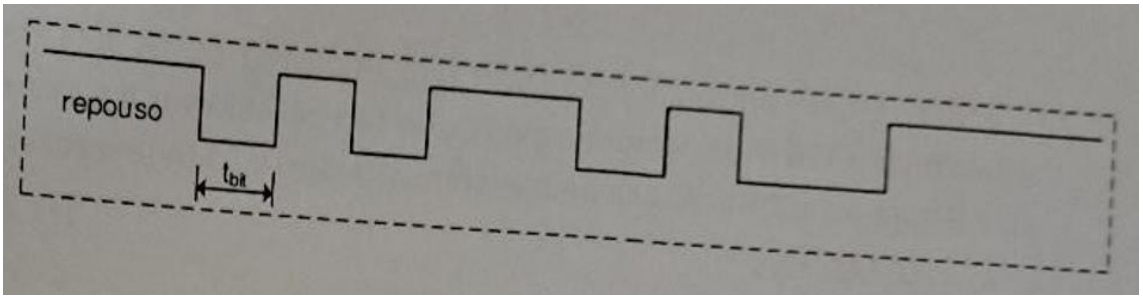
58. Considere um controlador de DMA **não dedicado** de 32 bits (i.e. com barramento de dados de 32 bits) a funcionar a 180 MHz. Suponha que são necessários 2 ciclos de relógio (= 1 T_{BC}) para efetuar uma operação de leitura e escrita. A taxa de transferência de pico desse DMA (expressa em Bytes/s), em modo ***cycle-stealing*** e com um tempo mínimo entre operações elementares de 2 T_{BC} é:
    a. 60 MByte/s
    b. 120 MByte/s
    c. 12.5 MByte/s
    d. 40 MByte/s
59. Na arquitetura de um sistema computacional, o *Data Bus* permite:
    a. Identificar, na memória externa/periférico, a origem/destino dos dados
    b. Especificar o tipo de operação efetuada sobre a memória
    c. Transferir o código máquina das instruções para o *program counter*
    d. Transferir dados entre a memória externa/periféricos e os registos do CPU
60. Suponha que pretende configurar os bits 7 e 6 do porto B do PIC32 como saída e entrada, respetivamente, **sem alterar a função dos restantes**. Em linguagem C, isso pode ser feito através da instrução:
    a. TRISB = (TRISB & 0xFF7F) | 0xFFFF;
    b. TRISB = (TRISB | 0x0040) & 0x0080;
    c. TRISB = (TRISB & ~(1 << 7)) | 1 << 6;
    d. TRISB = (TRISB & 0x0040) | 0x0080;
61. Considere um *timer* em que a relação entre as frequências de entrada e de saída é uma constante k configurável. Se colocar dois desses *timers* em cascata (i.e, ligados em série) com constantes de divisão **k1** e **k2**, a relação entre o período do sinal de relógio à entrada do primeiro *timer* (**Tin**) e o período à saída do segundo *timer* (**Tout**) pode ser calculada como:
    a. Tout = Tin / (k1 + k2)
    b. Tout = Tin * (k1 * k2)
    c. Tout = Tin / (k1 * k2)
    d. Tout = Tin / (k1 + k2)
62. O sinal de seleção "**Sel**" (em lógica positiva) 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. **Resposta c)**
    b. 

64. Numa transferência por DMA, o sinal *BusRequest* é utilizado pelo respetivo controlador para:
    a. Informar o CPU que a transferência de informação vai ter início, permitindo desse modo que o CPU suspenda a atividade de acesso ao exterior
    b. Efetuar ao CPU o pedido de cedência dos barramentos; a transferência tem início quando o DMA receber a confirmação, através do sinal *BusGrant*, de que os barramentos foram libertados
    c. Informar o CPU da existência de uma anomalia ocorrida durante o processo de transferência
    d. Sinalizar o CPU que a transferência de informação foi completada
65. No standard RS232C um dos tipos de erro de comunicação que é detetado é o erro de *framming*. Esse erro ocorre quando o recetor:
    a. Recebe um número de bits a 1 que não corresponde à paridade programada
    b. Recebe como "stop bit" um bit com nível lógico 0
    c. Recebe um bit de paridade diferente do programado
    d. Deteta um número de bits no campo de dados diferente do programado
66. Num barramento SPI é possível 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 explicito
    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
67. No barramento SPI o *master* seleciona o *slave* com quem vai comunicar através de:
    a. Informação transmitida na linha de dados
    b. Um sinal de seleção através do qual é transferido o endereço desse *slave*
    c. Um sinal de seleção que ativa antes de iniciar a transferência
    d. Um barramento de endereços de 7 bits a partir do qual cada dispositivo descodifica o seu próprio endereço
68. Na interface I^2C o primeiro byte gerado pelo *master*, numa comunicação contém a seguinte informação:
    a. O endereço do *master*, o endereço do *slave* com quem quer comunicar e a operação a realizar
    b. Os parâmetros de comunicação a utilizar na transferência (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* com quem quer comunicar e a operação a realizar

70. Suponha que pretende interligar, através de um protocolo/interface série, dois sistemas computacionais que distam de algumas dezenas de 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. CAN
    c. I^2C
    d. RS232C
71. Suponha que no barramento CAN, após uma situação de meio livre, três *masters* acedem simultaneamente ao barramento. O *master* 1 produz uma mensagem com o identificador 0x4C, o *master* 2 produz uma mensagem com o identificador 0x2A e o *master* 3 produz uma mensagem com o identificar 0x61. Nessa situação:
    a. Não há necessidade de arbitrar o acesso ao barramento, porque as mensagens produzidas pelos 3 *masters* têm identificadores diferentes
    b. O acesso ao barramento é ganho pelo *master* 2
    c. O acesso ao barramento é ganho pelo *master* 1
    d. O acesso ao barramento é ganho pelo *master* 3
72. Num *device driver* para uma UART (porta série RS232C) utilizam-se tipicamente:
    a. Estruturas de dados do tipo FIFO como meio de comunicação com a aplicação de alto nível, sendo o envio e a receção de caracteres da UART processados por interrupção
    b. Interrupções para sinalizar a aplicação de alto nível de que foi recebido um novo caracter ou que a UART está disponível para o envio de um novo caracter
    c. Interrupções geradas pela aplicação de alto nível para proceder ao envio de novos caracteres
    d. Buffers circulares como meio de comunicação com a aplicação de alto nível, sendo o envio e a receção de caracteres da UART processados por *polling*
73. Considere uma memória **DRAM** em que o barramento de endereços tem uma dimensão de 7 bits e a zona de armazenamento é constituída por 8 matrizes de células de 1 bit com 32 linhas. Podemos então concluir que estamos perante uma memória de:
    a. 4 kBytes
    b. 2 kBytes
    c. 16 kBytes
    d. 8 kBytes
74. Considere uma memória *cache* de mapeamento direto de 16 kByte, com blocos de 32 bytes, ligada a um processador com um espaço de endereçamento de 20 bits. Nesta *cache*, cada linha da *tag memory* armazena:
    a. 5 bits
    b. 7 bits
    c. 6 bits
    d. 8 bits

76. Na técnica designada por "memória virtual", o número de entradas do TLB é:
    a. Igual ao número de entradas da *page table*
    b. Igual ao número máximo de páginas virtuais
    c. Igual ao número máximo de páginas virtuais de memória usadas pelo processo em execução
    d. Dependente da implementação sendo sempre muito inferior ao número de entradas da *page table*
77. Considere um controlador de DMA **não dedicado** de 64 bits (i.e, com barramento de dados de 64 bits), a funcionar a 120 MHz. Suponha ainda que são necessários 4 ciclos de relógio (= 1 T_{BC}) para efetuar uma operação de leitura ou escrita. A taxa de transferência desse DMA (expressa em Bytes/s), a funcionar em **modo "bloco"** é:
    a. 160 MByte/s
    b. 120 MByte/s
    c. 64 MByte/s
    d. 100 MByte/s
78. O número de comparadores necessário para implementar uma *cache* parcialmente associativa de 128 kB com 256 linhas e blocos de 64 Bytes é:
    a. 1
    b. 4
    c. 8
    d. 16
79. Considere uma memória DRAM de 4Mx64, implementada com matrizes quadradas que utiliza um ciclo de refrescamento do tipo *RAS only*. Sabendo que o parâmetro *cycle time* do ciclo *RAS only* é 100 ns, o tempo necessário para fazer um refrescamento completo à memória é, aproximadamente:
    a. 205 µs
    b. 102 µs
    c. 200 ms
    d. 50 ns
80. Considere um sistema de supervisão, baseado no protocolo I2C, que recolhe periodicamente informação proveniente de 40 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 500 kHz, necessita para adquirir os valores de todos os sensores (cada um implementado num *slave* distinto) é:
    a. 1.6 ms
    b. 0.2 ms
    c. 4 ms
    d. 2.2 ms

82. Um determinado sistema de memória consegue responder a um acesso a uma posição de memória que se encontra na *cache* em 4 ciclos de relógio e a uma posição de memória que se encontra na memória central em 50 ciclos de relógio. Para este sistema de memoria se o tempo médio de acesso for 14 ciclos de relógio, isso significa um *hit ratio* na *cache* de:
    a. 94%
    b. 54%
    c. 80%
    d. 90%
83. Um dispositivo com interface RS232C, configurado para transmitir 7 bits de dados, paridade par e 2 stop bits, produz a trama da figura que é recebida por outro dispositivo RS232C configurado com o mesmo *baudrate*, mas com 7 bits de dados, sem paridade e 2 stop bit. Para a trama apresentada, o recetor:
    a. Não vai detetar qualquer erro, mas o valor recebido não é igual ao valor transmitido
    b. Não vai detetar qualquer erro e recebe corretamente o valor transmitido
    c. Vai detetar um erro de *framming*
    d. Vai detetar um erro de paridade



84. Considere um espaço de endereçamento de 32 bits e uma memória *cache* de 128 kByte, com uma organização parcialmente associativa com 4 vias, e blocos de 32 bytes. O número de linhas da *cache* é:
    a. 8
    b. 512
    c. 32
    d. 1024
85. Um sistema computacional com um sistema de memória *byte-addressable*, possui um espaço de endereçamento virtual de 16M e um espaço de endereçamento físico de 2M organizado em páginas de 1 kByte. O espaço de memória ocupado pela *Page Table*, sabendo que cada entrada tem 16 bits, é:
    a. 32 kBytes
    b. 16 kBytes
    c. 64 kBytes
    d. 256 kBytes

87. Um sistema computacional com uma memória *byte-addressable*, possui um espaço de endereçamento virtual de 4G e um espaço de endereçamento físico de 1G. A *Page Table* tem 1M entradas, está alinhada em endereços múltiplos de 4 e cada entrada tem 32 bits. Sabendo que o *Page table Register* tem o valor **0x3FF28000**, quando for gerado o endereço virtual **0x0006F5C**, o endereço da *Page Table* acedido para obter a tradução para o endereço físico será:
    a. 0x3FF2805C
    b. 0x3FF28008
    c. 0x3FF28F5C
    d. 0x3FF28F5C
    e. 0x3FF28018