

## AC2 - Questões teóricas (Especial)

- 1) O espaço de endereçamento é um conjunto de endereços disponíveis que um programa ou processo pode referenciar.
- 2) Descodificação de endereços.
- 3)
  - a) Cada byte (8 bits) tem o seu único endereço que pode ser acedido individualmente. A unidade de memória mais pequena que pode ser endereçada é o byte.
  - b) Cada bit tem o seu único endereço que pode ser acedido individualmente. A unidade de memória mais pequena que pode ser endereçada é o bit.
  - c) Cada word tem o seu único endereço que pode ser acedido individualmente. A unidade de memória mais pequena que pode ser endereçada é então uma word (sequência ordenada de bytes/bits que não podem ser acedidos individualmente).
- 4)
  - a) O barramento de dados é responsável por transferir dados entre o CPU, a memória principal e os módulos de I/O.
  - b) O barramento de endereços é responsável por transferir os endereços de memória, onde vai ser escrita ou lida a informação presente no barramento de dados.
  - c) O barramento de controlo é responsável por especificar o tipo de operação a realizar e assegurar a comunicação entre os vários elementos do sistema computacional, recolhendo também sinais de status.

### 5) Barramento de endereços

#### 6) Barramento de controlo.

## Microcontroladores (Embedded Systems)

- 7) Um cross-compiler é um programa que corre numa plataforma e gera código para uma plataforma com uma arquitetura diferente daquela onde é executado.
- 8) A função de um bootloader num sistema baseado em microcontroladores é transferir o código executável a partir de um sistema host, usado no desenvolvimento, para a memória do microcontrolador, permitindo a sua posterior exec.
- 9)
  - a) Circuitos integrados com CPU, RAM, ROM, periféricos.
  - b) frequências muito inferiores ( $< 200\text{MHz}$ ) comparado c/ um sistema de uso geral ( $> 3\text{GHz}$ )
  - c) Grande variedade de periféricos
  - d) Por norma, um custo inferior ao de um sistema de uso geral.
  - e) Baixo consumo energético, comparado a  $n \sim v \sim \gamma$
  - f) Utilizado em vários círculos em que a aplicação não necessite de elevados recursos computacionais. / Tarefas específicas.

10) Sistema computacional especializado c/ recursos disponíveis, em geral, mais limitados que num sistema computacional c/ uso geral. Tem requisitos próprios e executa apenas tarefas pre-definidas; normalmente, implementado c/ base num microcontrolador.

11) Um microcontrolador PIC32 usa internamente uma arquitetura Harvard com memórias de código e dados independentes e c/ diferentes tecnologias.

### Módulos I/O

12)

- a) Configurar o porto, como entrada ou saída.
- b) Escrever valores num porto de saída
- c) Ler valores de um porto de entrada.

13)

- a) Sim
- b) Sim
- c) RD-LAT

14)

- a) Sim
- b) RD-TRIS

15)

- a) Saída (Registro TRIS está a '0')
- b) Está a ser escrito o valor '0' no registo LAT de um determinado porto.

16)

- a) Saída (Registro TRIS está a '1')
- b) Está a ser feita a leitura do registo PORT (com 2 ciclos de atraso)

17) São utilizados flip-flops tipo D

18) Essa implementação visa resolver os problemas de meta-stabilidade decorrentes do facto de o sinal externo ser assíncrono ao sysclk.

19) Sure, not gonna happen.

### Nugões Básicas sobre periféricos

### 20) Módulo I/O

21) A CPU toma a iniciativa de estabelecer o contacto com o periférico e espera que o periférico esteja disponível para a troca de informação (polling). Durante o período de polling a CPU não realiza qualquer operação. Quando o periférico sinaliza a CPU de que este é pronto, a CPU inicia e controla a troca de informação.

22) E/S programada.

23) Referimo-nos à parte que providencia a adaptação entre as características intrínsecas do dispositivo periférico e as da CPU/memória.

24)

a) Portas tri-state

b) Porque garantem que se lê a informação do periférico (dispositivo E/S) certo, tendo em conta que estas portas só estão ativas quando os sinais CS e RD estão ativos em simultâneo, se isto não se verificar a entrada encontra-se em alta impedância.

25)

a) Operação: Leitura espaço de end. memória

b) Leitura I/O

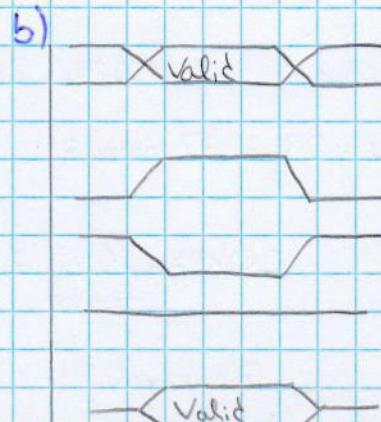
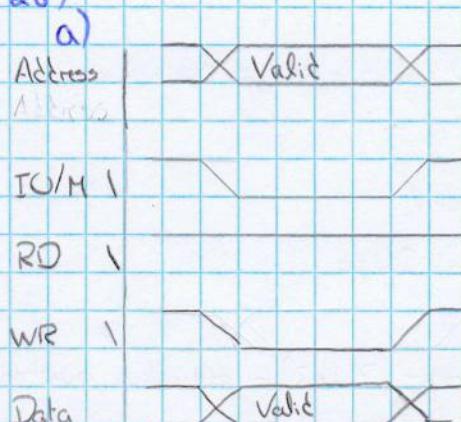
c) Escrita I/O

d) escrita I/O

e) escrita memória

f) leitura memória

26)



Interrupções

27)

a) O PIC32 tanto pode funcionar em "single-vector mode" (um vetor único p/ todas as fontes de interrupção, ident. p/ software) ou em "multi-vector mode" (as interrupções são vetorizadas, ident. p/ hardware).

b) Cada periférico possui um identificador único (vetor) e durante o processo de atendimento, na fase da ident. da fonte, o periférico gerador da interrupção identifica-se através do seu vetor. Este vetor vai ser depois usado como índice de uma tabela que especifica como chegar à RSI.

c) O controlador de interrupções permite a configuração da prioridade de cada fonte. Dessa forma a ordem de atendimento depende da valor da prioridade configurada de cada fonte. Este valor está presente nos 3 bits do registo IPC que codificam 7 níveis de prioridade (1 a 7, sendo '0' fonte desativada).

28)

a) Prologo → Salva guarda do contexto do programa que foi interrompido (registos internos da CPU → memória (stack))

b) Epílogo → Reposição do contexto do programa interrompido (memória (stack) → registos internos da CPU)

29) O overhead neste contexto é o custo adicional que o sistema tem para poder realizar transferências de informação por interrupção. Esse overhead é causado pelo número de ciclos gastos na mudança de contexto, prólogo e epílogo.

30)  $f = 10 \text{ MHz}$   
 $MIPS = 5 \Rightarrow CPI = 2$   
overhead = 20 ciclos  
RSI = 40 instruções

 $\text{Overhead} + (40 \times 2) = 20 + 80 = 100 \text{ ciclos}$ 
 $1 \text{ ciclo} = \frac{1}{10 \times 10^6} = 0,1 \cdot 10^{-6} = 0,1 \mu\text{s}$

$100 \times 0,1 \mu\text{s} = 10 \mu\text{s} \quad 10 \mu\text{s} = \frac{1}{10 \times 10^{-6}} = 0,1 \times 10^6 \text{ Hz} = 100 \text{ kHz}$

31)  $f = 40 \text{ MHz}$   
 $MIPS = 16 \Rightarrow CPI = 2,5$   
 $f_{out} = 200 \text{ kHz}$   
overhead = 75 ciclos

 $200 \text{ kHz} = \frac{1}{200 \times 10^3} = 0,005 \times 10^{-3} = 5 \mu\text{s}$ 
 $5 \mu\text{s} \times 40 \text{ MHz} = 200 \text{ ciclos total} - 75 = 125 \text{ ciclos}$ 
 $\frac{125}{2,5} + \frac{75}{2,5} = 50 + 30 = 80 \text{ interrupções.}$

32)  $f = 100 \text{ MHz}$   
 $MIPS = 33,3 \Rightarrow CPI = 3$   
 $f_{out} = 500 \text{ kHz}$   
overhead = 80 ciclos

 $500 \text{ kHz} = \frac{1}{500 \times 10^3} = 0,002 \times 10^{-3} = 2 \mu\text{s}$ 
 $100 \text{ MHz} \times 2 \mu\text{s} = 200 \text{ ciclos} - 80 = 120 \text{ ciclos}$ 
 $\frac{120}{3} + \frac{80}{3} = 40 + 27 = 67 \text{ instruções.}$

33)  $f = 10 \text{ MHz}$   
CPI = 2  
RSI = 70 instruções  
 $f_{out} = 50 \text{ kHz}$

 $50 \text{ kHz} = \frac{1}{50 \times 10^3} = 0,02 \times 10^{-3} = 20 \mu\text{s}$ 
 $20 \mu\text{s} \times 10 \text{ MHz} = 200 \text{ ciclos} - (70 \times 2) = 60 \text{ ciclos overhead}$

34)  $f = 80 \text{ MHz}$   
CPI = 2  
overhead = 40 ciclos  
RSI = 20 instruções

 $80 \text{ MHz} = \frac{1}{80 \times 10^6} = 0,0125 \times 10^{-6} = 12,5 \text{ ns/ciclo}$ 
 $40 + (2 \times 20) = 80 \text{ ciclos} \times 12,5 \text{ ns} = 1000 \text{ ns} = 1 \mu\text{s}$

$\frac{1}{1 \times 10^{-6}} = 1 \text{ MHz}$

- 35)
- Pela ordem que são colocados os periféricos na cache, relativamente ao CPU.
  - Interrupt acknowledge ("Inta")

- 36)
- 1) Interrupt request
  - Depois de detectar o Ireq, ativa o CPU, quando disponível para atender a interrupção, ativa o sinal Inta.
  - O periférico, depois de detectar o sinal Inta ativo, identifica-se colocando no barramento de endereços o vetor.
  - O CPU destina o sinal Ireq e armazena o vetor.
  - O CPU a partir do vetor do periférico e da tabela de vetores salta para a RSI.

37) Interrupções retorcidas - Daisy Chain.

38) Este sistema de interrupções tem apenas 1 entrada de interrupção e uma rotina para todas as fontes de interrupção. A rotina lê o registo de status de cada uma das periféricos até encontrar um que tenha gerado pedido de interrupção. Caso hajam vários pedidos em simultâneo, a ordem pela qual os periféricos é "questionados" determina a ordem da priorização no atendimento.

## DMA

39) 1) O DMA ativa o sinal BusReq, pedindo autorização ao CPU p/ ser bus master e espera.

2) Logo que possa, o CPU coloca os seus barramentos em alta impedância e ativa o sinal BusGrant.

3) O DMA ativa o sinal Dack e o DiskCtrl, de seguida, desativa o sinal Dreq.

4) O DMA efetua a transf. (Durante esta fase o CPU está impedido de aceder à memória).

5) O DMA termina a transf. e desativa os sinais Dack e BusReq. Ativa o sinal de interrupção.

6) O CPU após libertar o sinal BusReq desativo, desativa o sinal BusGrant e passa a ser novamente BusMaster.

7) O CPU, logo que possa, atende à interrupção gerada pelo DMA.

40) Retira o pedido para ser Bus Master, libertando os barramentos (desativa os sinais Dack e BusReq) e ativa o sinal de interrupção.

41) O sinal é gerado pelo CPU quando este deteta o sinal BusReq e o CPU está pronto para libertar os barramentos (colocá-los em alta impedância), tornando o gerador do sinal BusReq Bus Master.

42) Em modo cycle-stealing o CPU só libera os barramentos nos ciclos em que não acede à memória. O DMA é Bus Master durante 1 ciclo e depois libera-o.

1) Torna-se Bus Master;

2) Lê 1 palavra do source address;

3) Liberta os barramentos;

4) Espera;

5) Torna-se Bus Master;

6) Escreve uma palavra no dest. add.

7) Liberta os barramentos;

8) Incrementa o endereço e o num. da palavra transf.

43) Enquanto que em modo bloco o DMA assume o controlo dos barramentos até todos os dados terem sido transferidos, em modo "Burst" o DMA transfere rajadas de palavras até atingir um número de palavras pré-programadas ou até o periférico não ter mais informação a transferir. Se os dados não foram todos transferidos, opera-se reset no sinal Dreq reiniciando todo o processo de transferência.

44) DMA é dedicado (2 bus cycle p/palavra); 1 bus cycle = 1 ciclo relógio, o que indica que

$$f = 500 \text{ MHz} \quad 1 \text{ ciclo clk} : \frac{1}{500 \times 10^9} = 0,002 \times 10^{-9} = 2 \text{ ns} \times 2 = 4 \text{ ns}$$

$$512 \text{ words } 32 \text{ bits} \quad 512 \text{ words} \times 4 = 2048 \text{ ns}$$

$$b) \text{ctrl } 16 \text{ bits} \quad 1 \text{ ciclo clk} : \frac{1}{500 \times 10^9} = 1 \text{ ns} \times 2 = 2 \text{ ns} / 16 \text{ bits} \times 2 = 4 \text{ ns} / 32 \text{ bits}$$

$$f = 500 \text{ MHz} \quad 512 \text{ words } 16 \text{ bits} \quad 4 \text{ ns} \times 512 = 2048 \text{ ns}$$

$$c) 1024 \text{ ns}$$

$$d) \text{ctrl } 16 \text{ bits} \quad 1 \text{ ciclo clk} : \frac{1}{500 \times 10^9} = 2 \text{ ns} \times 2 = 4 \text{ ns} / 16 \text{ bits}$$

$$f = 500 \text{ MHz} \quad 2 \text{ k words } 16 \text{ bits} \quad 4 \text{ ns} \times 2048 = 8192 \text{ ns}$$

$$e) 2048 \text{ ns}$$

45) 1 bus cycle = 2 ciclos relógio ; DMA não dedicado

- a) ctrl 32 bits : 1 ciclo clk :  $\frac{1}{500 \times 10^6} = 1 \text{ ns}$ , 1 bus cycle =  $2 \text{ ns} \times 2 = 4 \text{ ns}$   
 $f = 500 \text{ MHz}$   
512 words 32 bits  $4 \text{ ns} \times 512 = 2048 \text{ ns}$
- b) ctrl 16 bits : 1 ciclo clk :  $(2 \text{ ns} \times 2) \times 2 = 8 \text{ ns} \times 2 = 16 \text{ ns / 32 bits}$   
 $f = 500 \text{ MHz}$   
2K de 32 bits  $16 \times 2048 = 32768 \text{ ns}$
- c) 1 ciclo clk :  $\frac{1}{500 \times 10^6} = 1 \text{ ns} \times 2 = 2 \text{ ns} \times 2 = 4 \text{ ns} \times 2 = 8 \text{ ns / 32 bits}$   
 $8 \text{ ns} \times 256 = 2048 \text{ ns}$
- d) 1 ciclo clk :  $\frac{1}{500 \times 10^6} = (2 \text{ ns} \times 2) \times 2 = 8 \text{ ns / 16 bits}$   
 $8 \text{ ns} \times 2048 = 16384 \text{ ns}$

46) Vai levarar metade do tempo tendo em conta que só precisa de 1 bus cycle por palavra

47) DMA não dedicado ; Cycle-stealing (1 ciclo relógio entre op); 1 bus cycle = 2 ciclos relógio

- a) ctrl 32 bits : 1 ciclo clk :  $\frac{1}{250 \times 10^6} = 0,004 \times 10^{-6} = 4 \text{ ns}$   
 $f = 250 \text{ MHz}$   
512 words 32 bits  $1 \text{ palavra} = (\text{fetch} + \text{wait} + \text{deposit} + \text{wait}) = (4 \times 2) + 4 + (4 \times 2) + 4 = 24 \text{ ns}$   
 $\hookrightarrow 1 \text{ bus cycle} = 2 \text{ ciclos de clk}$   
 $512 \times 24 = 12288 \text{ ns}$
- b) ctrl 16 bits : 1 ciclo clk :  $\frac{1}{16 \text{ Hz}} = 1 \text{ ns} ; 2 \text{ ns / palavra 32 bits}$   
 $f = 1 \text{ GHz}$   
512 words de 32 bits  $1 \text{ palavra} = \text{fetch} + \text{wait} + \text{deposit} + \text{wait} = 2 \times 2 + 2 + 2 \times 2 + 2 = 12 \text{ ns}$   
 $512 \times 12 \text{ ns} = 6144 \text{ ns}$
- c) ctrl 16 bits : 1 ciclo clk :  $\frac{1}{500 \times 10^6} = 0,002 \times 10^{-6} = 2 \text{ ns}$   
 $f = 500 \text{ MHz}$   
2K words de 16 bits  $1 \text{ palavra} = 2 \text{ ns} \times 2 + 2 \text{ ns} + 2 \text{ ns} \times 2 + 2 \text{ ns} = 12 \text{ ns}$   
 $2048 \times 12 \text{ ns} = 24576 \text{ ns.}$

48) DMA dedicado ; modo bloco

- a) ctrl 32 bits : 1 ciclo clk :  $\frac{1}{500 \times 10^6} = 2 \text{ ns} \rightarrow 1 \text{ bus cycles}$   
 $f = 500 \text{ MHz}$   
512W de 32 bits 512 bus cycles
- b) ctrl 16 bits : 1 ciclo clk :  $1 \text{ ns} \Rightarrow 1 \text{ palavra 32 bits} = 2 \text{ ns}$   
 $f = 1 \text{ GHz}$   
4K W de 32 bits  $\frac{1}{2 \text{ bus cycles}} = 2 \text{ ns}$   
 $4 \times 2 = 8 \text{ K bus cycles}$
- c) ctrl 16 bits : 1 ciclo clk :  $1 \text{ ns} \Rightarrow 1 \text{ palavra 32 bits} = 2 \text{ ns}$   
 $f = 1 \text{ GHz}$   
512W 32 bits  $\frac{1}{2 \text{ bus cycles}} = 2 \text{ ns}$   
 $512 \times 2 = 1024 \text{ bus cycles}$
- d) ctrl 16 bits : 1K bus cycles.  
 $f = 500 \text{ MHz}$   
1K W 16 bits

49) DNA é dedicado 32 bits  
 $f = 100 \text{ MHz}$   
 Cycle stealing  
 2 ciclos p/ operação  
 wait = 2 ciclos

$1 \text{ ciclo relogio} = \frac{1}{100 \times 10^6} = 10 \text{ ns}$

$1 \text{ palavra} = 4 \text{ bus cycles} (1 \text{ fetch + wait + deposit + wait})$   
 $= 4 \times (2 \times 10) = 80 \text{ ns}$

$1 \text{ word} = 4 \text{ bytes} \Rightarrow T_{\text{byte}} = \frac{80 \text{ ns}}{4} = 20 \text{ ns}; f_{\text{byte}} = \frac{1}{20 \times 10^{-9}} = 0,05 \times 10^9 = 50 \text{ MB/s}$

50)

a) ctrl 32 bits  
 $f = 120 \text{ MHz}$   
 $1 \text{ bus cycle} = 2 \text{ ciclos clk}$   
 wait = 2 bus cycle = 4 ciclos clk

$1 \text{ ciclo relogio} = \frac{1}{120 \times 10^6} = 8,3 \text{ ns}$

$1 \text{ palavra} = 2 \times 8,3 + 4 \times 8,3 + 2 \times 8,3 + 4 \times 8,3 = 99,6 \text{ ns}$

$1 \text{ word} = 4 \text{ bytes} \Rightarrow T_{\text{byte}} = \frac{99,6}{4} = 24,9 \text{ ns}; f_{\text{byte}} = \frac{1}{24,9 \times 10^{-9}} = 40 \text{ MB/s}$

b) ctrl 32 bits  
 $f = 80 \text{ MHz}$

$T_{\text{bc}} = 2 \text{ ciclos clk}$   
 wait = 3  $T_{\text{bc}}$  = 6 ciclos

$1 \text{ ciclo clk} = \frac{1}{80 \times 10^6} = 12,5 \text{ ns}$

$1 \text{ palavra} = 2 \times 12,5 + 6 \times 12,5 + 2 \times 12,5 + 6 \times 12,5 = 25 + 75 + 25 + 75 = 200 \text{ ns}$

$T_{\text{byte}} = \frac{200}{4} = 50 \text{ ns}; f = \frac{1}{50 \times 10^{-9}} = 0,02 \times 10^9 = 20 \text{ MB/s}$

c) ctrl 16 bits  
 $f = 120 \text{ MHz}$

$T_{\text{bc}} = 2 \text{ ciclos clk}$

wait = 2  $T_{\text{bc}}$  = 4 ciclos clk  
 $1 \text{ palavra} = 2 \times 8,3 + 4 \times 8,3 + 2 \times 8,3 + 4 \times 8,3 = 99,6 \text{ ns} / 16 \text{ bits} \Rightarrow 199,2 \text{ / 32 bits}$

$T_{\text{byte}} = \frac{199,2}{4} = 49,8 \text{ ns}; f_{\text{byte}} = \frac{1}{49,8 \times 10^{-9}} \approx 20 \text{ MB/s}$

d) ctrl 16 bits  
 $f = 200 \text{ MHz}$

$T_{\text{bc}} = 1 \text{ cc}$

wait = 1  $T_{\text{bc}}$  = 1 cc  
 $1 \text{ palavra} = 1 \times 5 \text{ ns} = 20 \text{ ns} / 16 \text{ bits} \Rightarrow 40 \text{ ns} / 32 \text{ bits}$

$T_{\text{byte}} = \frac{40}{4} = 10 \text{ ns}; f_{\text{byte}} = \frac{1}{10 \times 10^{-9}} = 100 \text{ MB/s}$

51) DNA é dedicado 32 bits

$f = 100 \text{ MHz}$

$T_{\text{bc}} = 2 \text{ cc}$

a)  $T = \frac{1}{100 \times 10^6} = 0,01 \times 10^{-6} = 10 \text{ ns}$        $1 \text{ palavra} = 2 \times 10 \text{ ns} + 2 \times 10 \text{ ns} = 10 \text{ ns}$

$T_{\text{byte}} = \frac{40 \text{ ns}}{4} = 10 \text{ ns}; f_{\text{byte}} = \frac{1}{10 \times 10^{-9}} = 0,1 \times 10^9 = 100 \text{ MB/s}$

b) wait = 1 cc

$1 \text{ palavra} = 2 \times 10 \text{ ns} + 10 \text{ ns} + 2 \times 10 \text{ ns} + 10 \text{ ns} = 60 \text{ ns}$        $T_{\text{byte}} = \frac{60 \text{ ns}}{4} = 15 \text{ ns}$

$f_{\text{byte}} = \frac{1}{15 \times 10^{-9}} = 0,067 \times 10^9 = 67 \text{ MB/s}$

c) a)  $T = 10\text{ns}$ ,  $1\text{ palavra} = 2 \times 10\text{ns} = 20\text{ns}$   
 $T_{byte} = \frac{20\text{ns}}{5} = 4\text{ns}$ ,  $f_{byte} = \frac{1}{4 \times 10^{-9}} = 250\text{MB/s}$

b)  $1\text{ palavra} = 2 \times 10\text{ns} + 10\text{ns} = 30\text{ns}$   
 $T_{byte} = \frac{30}{7,5} = 4\text{ns}$ ,  $f_{byte} = \frac{1}{4 \times 10^{-9}} = 250\text{MB/s}$ .

52) Timers

52)  $f_{out1} = \frac{f_{in}}{K_1}$

$$f_{out} = \frac{f_{out1}}{K_2} = \frac{\frac{f_{in}}{K_1}}{K_2} = \frac{f_{in}}{K_1 \cdot K_2}$$

53) Duty-cycle é a percentagem de tempo em que uma determinada sinal está ativo.

Uma aplicação prática da manipulação deste valor é o controle das rotações de um motor DC.

54)  $f_{out} = \frac{f_{in}}{(K+1)}$

a)  $f_{out} = \frac{20\text{MHz}}{2000} = 20\text{kHz}$

b)  $f_{out} = \frac{40\text{MHz}}{1250} = 32\text{kHz}$

c)  $f_{out} = \frac{80\text{MHz}}{32768} = 2,5\text{kHz}$

d)  $f_{out} = \frac{2\text{MHz}}{1024} = 1,95\text{kHz}$

55) A divisão por 2 permite obter um duty-cycle de 50%.

56)  $T_{CLK} = 20\text{nHz}$

Rescaler = 4

PR = 2499

OCk = 83k

$$f_{out} = f_{out\text{presc}} = \frac{T_{CLK}}{\frac{Pres}{PR+1}} = \frac{\frac{20 \times 10^9}{4}}{2500} = \frac{20 \times 10^9}{10000} = \frac{200000000}{10000000}$$

$T_{out} = \frac{1}{2 \times 10^3} = 50\text{ms}$

$OCk = \frac{(PR+1) \times \text{duty-cycle}}{100} \Rightarrow \text{duty-cycle} = \frac{OCk \times 100}{PR+1} = \frac{834 \times 100}{2500} = 33,36\%$

57)  $f_{out} = 85\text{Hz}$

Timer T<sub>E</sub>

PBclk = 50MHz

a)  $85 =$

$$(ad) = \frac{f_{out\text{pres}}}{PR+1} \Rightarrow f_{out} = \frac{PBclk}{PR+1} \Rightarrow f_{out} = \frac{PBclk}{Pres \times (PR+1)} \Rightarrow \left[ \begin{array}{l} PBclk \\ Pres \times (PR+1) \\ f_{out} \times (PR+1) \end{array} \right]$$

$$\Rightarrow Pres = \left[ \frac{50\text{MHz}}{85 \times 65537} \right] \Rightarrow Pres = 9 \Rightarrow Pres \min = 16$$

$$b) f_{out} = \frac{f_{osc}}{PR+1} = \frac{50MHz}{16} \Rightarrow PR+1 = 36769 \Rightarrow PR = 36763$$

58)  $K_{pres} = ?$

$$a) K_{pres} = 9 \Rightarrow K_{min} = 64$$

$$b) 85 = \frac{50MHz}{64 \times (PR+1)} \Rightarrow PR+1 = \frac{50MHz}{64 \times 85} \Rightarrow PR = 9190$$

59)  $f_{out} = 100Hz$

$$\text{duty cycle} = 25\%$$

$$PBclk = 40MHz$$

$$a) K_{prescaler} \leq \left\lceil \frac{PBclk}{(65535+1) \times f_{out}} \right\rceil = \left\lceil \frac{40 \times 10^6}{65536 \times 100} \right\rceil = 7$$

$$K_{min} = 8$$

$$b) PR = \frac{40MHz}{8 \times 100} - 1 \Rightarrow PR = 62500$$

$$OC5RS = \frac{(PR+1) \text{ duty cycle}}{100} = \frac{62500 \times 25}{100} = 15625$$

$$c) \text{Resolution} = \log_2 (62500) = 15,93 = 16$$

60)  $TCLK = 20MHz$   $PR \in OCK = ?$

$$K = 8$$

$$f_{out} = 200Hz$$

$$\text{duty cycle} = 25\%$$

$$f_{out} = \frac{TCLK}{K \times (PR+1)} \Rightarrow PR+1 = \frac{TCLK}{f_{out} \times K} \Rightarrow PR = 12499$$

$$OCK = \frac{(12499+1) \times 25}{100} = 3125$$

61)  $PBclk = 20MHz$

$$f_{out} = \frac{1}{15 \times 10^3} = \frac{1000}{15} = \frac{200}{3}$$

$$K_{pres} = \left\lceil \frac{50 \times 10^6}{(65535+1) \times 200/3} \right\rceil = 11,62 = 12 \Rightarrow K_{min} = 64$$

$$\frac{200}{3} = \frac{\frac{50 \times 10^6}{64}}{PR+1} \Rightarrow PR+1 = \frac{50 \times 10^6}{64 \times (\frac{200}{3})} \Rightarrow PR = 1718,75 = 1719$$

$$62) f_{out} = \frac{f_{in}}{K+1}$$

$$a) f_{in} = 20MHz$$

$$f_{out} = \frac{1}{5 \times 10^3} = \frac{1000}{5} = 200Hz$$

$$K+1 = \frac{20 \times 10^6}{200} \Rightarrow K+1 = 1000000 \Rightarrow K = 999999$$

$$b) f_{in} = 25MHz$$

$$f_{out} = \frac{1}{1 \times 10^3} = 1000Hz$$

$$K+1 = \frac{250000000}{1000} \Rightarrow K = 250000$$

$$c) f_{in} = 40 \text{ MHz}$$

$$f_{out} = \frac{1}{250 \times 10^3} = 0,004 \times 10^{-3} = 4 \text{ Hz}$$

$$K+1 = \frac{40000000}{4} = K = 9999999$$

63)

$$a) f_{in} = 100 \text{ kHz}$$

$$\text{counter 16 bits} \Rightarrow \text{contar até } 2^{16} = 65536$$

$$T = \frac{1}{100 \times 10^3} = 10 \text{ ms} \times 65536 = 0,655 \text{ s}$$

OU

$$f_{out} = \frac{f_{pbclk}}{PR+1} = \frac{100 \text{ kHz}}{65537} = 1,526 \text{ Hz}$$

$$T_{at} = \frac{1}{1,526} = 0,655 \text{ s}$$

(Contador incrementa até PR  $\Rightarrow$  reset)

$$b) f_{in} = 20 \text{ kHz}$$

$$\text{counter 10 bits} \Rightarrow 2^{10} = 1024$$

$$T = \frac{1}{20 \times 10^3} = 0,05 \times 10^{-3} = 50 \mu\text{s} \times 1024 = 0,0512 \text{ ms}$$

$$c) f_{in} = 50 \text{ MHz}$$

$$\text{counter 24 bits} \Rightarrow 2^{24} = 16777216$$

$$T = \frac{1}{50 \times 10^6} = 0,02 \times 10^{-6} = 20 \text{ ns} \times 16777216 = 0,3355 \text{ s}$$

$$64)$$

$n_{min}$	$2^n = PR+1 \Rightarrow n_{min} = \log_2 (PR+1)$	$PR+1 = \frac{f_{in}}{f_{out}} = \frac{100 \text{ kHz}}{5,88 \text{ Hz}} = 17006,80277$
$f_{in} = 100 \text{ kHz}$	$T_{at} = [10 \text{ ms} \dots 170 \text{ ms}] ; f_{out} = [100 \text{ Hz} \dots 5,88 \text{ Hz}]$	$n_{min} = \lceil \log_2 (17006,80) \rceil = \lceil 15,05 \rceil = 16 \text{ bits}$

$$65) f_{in} = 250 \text{ kHz}$$

$$T_{at} = [25 \text{ ms} \dots 180 \text{ ms}] ; f_{out} = [8 \text{ Hz} \dots 2,1 \text{ Hz}]$$

$PR+1 = \frac{250 \text{ kHz}}{2,1} = 119047,62$
$n_{min} = \lceil \log_2 (119047,62) \rceil = \lceil 16,86 \rceil = 17 \text{ bits}$

$$66) T_{at} = [50 \text{ ms} \dots 150 \text{ ms}] ; f_{out} = [20 \text{ Hz} \dots 6,67 \text{ Hz}]$$

$$PR+1 = 2^k$$

$f_{in} = ?$

$$PR+1 = \frac{f_{in}}{f_{out}} \Rightarrow f_{in} = (PR+1) \cdot f_{out} \Rightarrow f_{in} = 65536 \cdot 6,67 = 437125,12 \text{ Hz}$$

Nótes Básicas de Barramentos / Decote de endereços.

$$67) \text{Address bus} = 16 \text{ bits}$$

$$\text{bloco mem} = 1 \text{ KByte}$$

$$A_{15} A_{14} A_{13} A_{12} A_{11} A_{10} \dots A_0$$

$$1 \text{ e } 0 \text{ e } 0 \text{ e } 0 \text{ e } \dots$$

$$10 \text{ bits}$$

$$a) i) CS = A_{15} \cdot \bar{A}_{13} \cdot \bar{A}_{11}$$

$$\begin{matrix} 0 & 1 \\ \text{Lógica positiva: } & \bar{A}_{11} \cdot A_{10} \end{matrix}$$

$$ii) CS = \bar{A}_{15} + \bar{A}_{13} + \bar{A}_{11}$$

$$\text{negativa: } A_{11} + \bar{A}_{10}$$

$$b) \text{Gamma base: } 1000 \ 0000 \ 0000 \ 0000 \dots 1000 \ 0011 \ 1111 \ 1111 = [0x8000 \dots 0x83FF]$$

A <sub>15</sub>	A <sub>14</sub>	A <sub>13</sub>	A <sub>12</sub>	A <sub>11</sub>	A <sub>10</sub>
1	0	0	0	0	0
1	0	0	0	1	0
1	0	1	0	0	0
1	0	1	0	1	0

$0x8000 \dots 0x83FF$

A <sub>15</sub>	A <sub>14</sub>	A <sub>13</sub>	A <sub>12</sub>	A <sub>11</sub>
1	1	0	0	0
1	1	0	0	1
1	1	0	1	0
1	1	0	1	0

$0xC000 \dots 0xC3FF$

$0xC400 \dots 0xC7FF$

$0xD000 \dots 0xD3FF$

$0xD400 \dots 0xD7FF$

68) Address bus = 16 bits

bloco mem - 4KB  $\Rightarrow$  12 bits

$$\bar{CS} = A_{15} + \bar{A}_{13} \Leftrightarrow CS = \bar{A}_{15} \cdot A_{13}$$

- a)  $0\ 0\ 1\ 0\ 0x2000 \dots 0x2FFF$  ( $0010\ 0000\ 0000\ 0000\dots 0010\ 1111\ 1111$ )  
 $0\ 0\ 1\ 1\ 0x3000 \dots 0x3FFF$   
 $0\ 1\ 1\ 0\ 0x6000 \dots 0x6FFF$   
 $0\ 1\ 1\ 1\ 0x7000 \dots 0x7FFF$
- b)  $0x200F \dots 0x300F \dots 0x600F \dots 0x700F$

69) Add bus = 16 bits

RAM 1KB = 10 bits  $\Rightarrow$  6 bits seleção :  $CS = \bar{A}_{12} \cdot \bar{A}_{11} \cdot \bar{A}_{13} \cdot \bar{A}_{12} \cdot \bar{A}_1 \cdot \bar{A}_0$

Porto :  $CS = \bar{A}_{15} \cdot \bar{A}_{14} \cdot \bar{A}_{13} \cdot \bar{A}_{11} \cdot A_{10} \cdot \bar{A}_9 \cdot A_8 \cdot \bar{A}_7 \cdot A_6 \cdot \bar{A}_5 \cdot \bar{A}_4 \cdot \bar{A}_3 \cdot \bar{A}_2 \cdot \bar{A}_1 \cdot \bar{A}_0$

Dispositivo :  $CS = \bar{A}_{15} \cdot \bar{A}_{14} \cdot \bar{A}_{13} \cdot \bar{A}_{12} \cdot \bar{A}_{11} \cdot A_{10} \cdot A_9 \cdot \bar{A}_8 \cdot \bar{A}_7 \cdot \bar{A}_6 \cdot \bar{A}_5 \cdot \bar{A}_4 \cdot \bar{A}_3$

ROM 2KB = 11 bits  $\Rightarrow$  5 bits seleção :  $CS = A_{15} \cdot A_{14} \cdot A_{13} \cdot A_{12} \cdot A_{11}$

70)  $0x001002 \Rightarrow 20$  bits = K

S = 3 bits e N = 32 bits

N = K + R + S  $\Rightarrow$  R = 32 - 20 - 3  $\Rightarrow$  R = 9 bits.

S:  $2^S$  = sub-gamas / linhas de seleção  
R:  $2^R$  = Dimensão de cada sub-gama

	K	S	R
$0x001002000 \dots 0x0010021FF$	1	1	1111 1111
$0x001002200 \dots 0x0010023FF$	1	1	1111 ...
$0x001002400 \dots 0x0010025FF$	1	1	0101 ...
$0x001002600 \dots 0x0010027FF$	1	1	0111 ...
$0x001002800 \dots 0x0010029FF$	1	1	1001 ...
$0x001002A00 \dots 0x001002BFF$	1	1	1011 ...
$0x001002C00 \dots 0x001002DFF$	1	1	1101 ...
$0x001002E00 \dots 0x001002FFF$	1	1	1111 ...

71)

a)

N = 20 bits

S =  $\log_2(4) = 2$  bits ( $2^2 = 4$  linhas de seleção/sub-gama)

R =  $\log_2(8K) = 13$  bits ( $2^{13} = 8K$ , dim da cada sub-gama)

K = N - S - R = 5 bits

b) Nove

c) 1º:  $[0000\ 0000\ 0000\ 0000\dots 0000\ 0111\ 1111\ 1111] \Rightarrow [0x0000\dots 0x07FFF]$

2º:  $[0x08000\dots 0x0FFFF]$

última:  $[0xF8000\dots 0xFFFF]$

d)

1º memória:  $[1111\ 1000\ 0000\ 0000\dots 1111\ 1001\ 1111\ 1111] \Rightarrow [0xF8000\dots F9FFF]$

2º mem:  $[0xFAC00\dots 0xFBFFF]$

3º mem:  $[0xFC000\dots 0xFDFFF]$

4º mem:  $[0xFE000\dots 0xFFFF]$

$$e) \text{ 0x3AC45} = \underbrace{\text{0011}}_K \underbrace{\text{1010}}_S \underbrace{\text{1100}}_R \text{ 0100 0101}$$

endereços da gama: [0x38000... 0x3FFFF]

endereços da memória: [0x3A000... 0x3BFFF]

70) a)

73)

74)  $N = 16$  bits

$$\bar{CE} = A_{15} + A_{14} + \bar{A}_2 \Rightarrow CE = \bar{A}_{15} \cdot \bar{A}_{14} \cdot A_2$$

$$\begin{matrix} 0 & 0 & 0 & 1 & \dots \\ \hline 0x1000 & \dots & 0x1FFF \\ 0x3000 & \dots & 0x3FFF \end{matrix}$$

75)  $N = 32$  bits

$$S = \log_2 64 \Rightarrow S = 6 \text{ bits}$$

$$R = \log_2 16K \Rightarrow R = 14 \text{ bits}$$

$$K = 32 - 14 - 6 = 12 \text{ bits}$$

a) 12 bits Gama: 0x00000000... 0x000FFFFF → ?

b)  $N = 32$  bits Gama: 0x00000000... 0xCCCCFFFF → ?

$$S = 6 \text{ bits}$$

$$R = \log_2 (1K) = 10 \text{ bits}$$

$$K = 32 - 10 - 6 = 16 \text{ bits}$$

76)  $f_{\text{in}} = 200 \text{ MHz}$   
 $\text{CPI} = 2,5$

10 instruções

32 bits

$$T_{\text{in}} = \frac{1}{200 \times 10^6} = 0,005 \times 10^{-6} = 5 \text{ ns / ciclo}$$

$$2,5 \times 10 = 25 \times 5 \text{ ns} = 125 \text{ ns / 32 bits} = 4 \text{ bytes}$$

$$T_{\text{byte}} = \frac{125 \text{ ns}}{4} = 31,25 \text{ ns / byte}$$

$$f_{\text{byte}} = \frac{1}{31,25 \times 10^{-9}} = 32 \text{ MB/s.}$$

77) 16 circuitos c/  $17 \times k$  bits =  $16 \cdot 17 \times k$  bits

1 byte = 8 bits logo  $8 \cdot 17 \times k$  bits

78)  $512 \text{ K} / 64 \text{ K} = 8$  circuitos de  $64 \text{ K}, 8$  bits

79)  $N = 16$  bits

$A_{15} A_{14} A_{13} A_{12} A_{11} A_{10} \dots$   
 $x \quad x \quad 0 \quad 0 \quad 0 \quad 0 \quad \dots$

a)  $2^2 = 4 \Rightarrow 3$  réplicas

b) Os espaços de endereçamento são não-contíguas. Não sei explicar porquê mas acho que i por os 2 bits livres da descontinuidade estarem à esquerda (MSB) dos 4 bits de seleção e não à direita (LSB).

80)

a)  $\underbrace{0 \times 0000 \dots 0}_{\text{0000 0000}} \underbrace{\times 0FFF}_{\text{0000 0000 0000 1111 1111}} \Rightarrow R = 11 \text{ bits}$  ( $0$  que seleciona 3 os 9 msb)

$\underbrace{0000 0000}_{\text{0000 0000}} \underbrace{0100}_{\text{0000 0000}} \underbrace{0000}_{\text{0000}}$

$$\bar{Sel} = \prod_{i=11}^{15} A_i + \sum_{i=11}^{15} A_i$$

b)  $\bar{Sel} = \prod_{i=11}^{15} A_i + \sum_{i=11}^{15} A_i$

c)  $16 \text{ K} \Rightarrow R = 14 \text{ bits}$

$$\underbrace{0 \times 8000 \dots 0}_{\text{1000 1100}} \underbrace{\times 0FFF}_{\text{1000 1111 1111 1111}} = \underbrace{1000 1100}_{\text{1000 1100}} \underbrace{0000 0000}_{\text{0000 0000}}$$

$$\bar{Sel} = \prod_{i=16}^{18} A_i + \bar{A}_{15} + \bar{A}_4$$

d)  $\underbrace{0 \times 0000 \dots 0}_{\text{0000 0000 0000 1111 1111}} \times 03FF$

$$Sel = \prod_{i=10}^{15} A_i$$

e)  $\underbrace{0 \times 9000 \dots 0}_{\text{1001 ...}} \times 9FFF$

$$Sel = A_{16} \cdot \bar{A}_{15} \cdot \bar{A}_{13} \cdot A_2$$

81)

a)  $(0x1 \dots$

e)  $0x10$

$$[0x1000 \dots 0x1FFF] [0x3000 \dots 0x3FFF] [0x20000000 \dots 0x2FFFF \dots]$$

b)  $0x11 \dots$

$$[0x60000000 \dots 0x6FFFF \dots]$$

$$[0x3000 \dots 0x3FFF] [0x7000 \dots 0x7FFF]$$

c)  $0x01 \dots$

$$[0x10000000 \dots 0x1FFFFFF] [0x500000 \dots 0x5FFFFFF]$$

## 82) Organização em barramentos de dados.

- a) A prioridade é pre-programada e é atribuída ao elemento com maior prioridade.
- b) A prioridade é atribuída de forma a que o último elemento a fazer o pedido seja o primeiro a ser selecionado.
- c) os elementos são ordenados por ordem da pedida e o barramento é atribuído por ordem de chegada.

c) Acesso ao barramento - atribuição rotativamente, um master quando termina a transferência, passa o controlo ao master seguinte.

83) Protocolo semi-síncrono, opera de forma síncrona mas turna-se "handshaking" quando houver um pedido explícito, por parte do slave, para alterar a temporização das ações.

84) Barramento paralelo de tipo síncrono e caracterizado pela existência de um sinal clock e a transmissão de dados em N linhas.

São suportadas trans. do tipo:

- Read-modify-write
- Read-after-write
- Block

85) Transferências assíncronas são caracterizadas pela utilização de sinais protocolo (valid/accept) que asseguram a correta troca de informação entre a origem e o destino. Esta flexibilidade temporal permite a comunicação em sistemas c/ diversos dispositivos slaves com diferentes frequências.

86) barramento de 16 bits ; microciclo

a) 2 ciclos/entregamento + 2 ciclos / transf. de dados

4 ciclos

b) 1 ciclo/entregamento + 2 ciclos / transf. de dados

3 ciclos

c) 2 ciclos/entregamento + 1 ciclo / transf. de dados

3 ciclos

d) 3 ciclos/entregamento + 1 ciclo / transf. de dados

4 ciclos

87) Starvation acontece quando um elemento não tem acesso ao barramento devido a constantes pedidos provenientes de elementos de prioridade superior.

$$f_{in} = 200\text{MHz}$$

ciclo c/ 10 instruções

$$f_{in} = \frac{1}{200 \times 10^6} = 0,005 \cdot 10^{-9} = 5\text{ns}$$

CPI = 2,5

$$10 \text{ instruções} \Rightarrow 25 \text{ ciclos} \Rightarrow 125\text{ns} / 32 \text{ bits} = 175\text{ns} / 4 \text{ bytes}$$

$$T_{byte} = \frac{125\text{ns}}{4} = 31,25\text{ns} ; f_{byte} = \frac{1}{31,25 \cdot 10^{-9}} = 32\text{GB/s.}$$

88)

a) Leitura

- b) Transferência síncrona
- c) Micro-ciclo

90)

a)  $f = 500\text{MHz}$   $T = \frac{1}{500 \times 10^9} = 0,002 \times 10^{-9} = 2\text{ns / ciclo}$

$T_{acc} = 12\text{ns}$ ;  $T_{decod} = 2,5\text{ns}$

$T_{total} = 14,5\text{ns} \approx 15\text{ns} / 2\text{ns} = 7,5\text{ ciclos}$

$7,5 - 4 = 3,5 \times 2 = 7\text{ns}$

4 wait-states.

Aula 11el2  
rever

b)  $f = 200\text{MHz}$   $T = 5\text{ns}$

$T_{valit} = 12,5\text{ns}$

$T_{acc} = 35\text{ns}$ ;  $T_{decod} = 7\text{ns}$

$T_{total} = 35 + 7 + 12,5 = 54,5\text{ns} / 5\text{ns} = 11\text{ ciclos}$

São necessários 9 wait-states.

91)

- a) A fase de entregaamento engloba a transf. de dados.
- b) O entregaamento e a transf. de dados são tratados como op. autónomas com sinais de controlo e informação separados.
- c) Operação atómica, intuiçável, que consegue ler uma posição da memória e escrever na mesma posição da memória um valor novo (que seja completamente novo ou alterado).
- d) Permite ler (voltar) após a escrita de um determinado valor.
- e) Permite a transf. de um bloco de dados c/ apenas 1 ciclo de entregaamento. A primeira transf. requer um ciclo de entregaamento mas as transfs. subsequentes apenas incluem ciclos de dados.

92)

- a) escrita: escrita (dados disponíveis antes do strobe) ] ponto vista leitura: leitura (dados disponíveis depois do strobe). ] CPU

b) Síncrona

c) mui. de entregos e dados

d) escrita/leitura c/ protocolo síncrono e entregaamento micro-ciclo em barramento multiplexado.

93) Round-Robin, prioritário-dinâmico (FIFS)

94) @ Ao elemento de maior prioridade.

## Barramento de Comunicação Série

95)

- a) É mais simples de implementar
- b) Menos cablagem
- c) Diminuição de custos e interligação
- d) Possibilita a transmissão a longas distâncias
- e) Taxas de transmissão mais elevadas

96)

- a) barramento inclui sinal de relógio
- b) barramento não inclui sinal de relógio mas inclui sinais extra de protocolo para sinalizar princípio e fim da transmissão.

97) Partilhar o mesmo sinal de relógio.

Ex: Relógio explícito do transmissor; Relógio explícito do receptor; Relógio explícito mutuamente sincronizado; Relógio codificado.

98) A informação transmitida resulta de um exclusivo entre o bit a transmitir com o sinal de relógio.

99) Full-duplex: sistema de comunicação bidirecional em que a transmissão de info. ocorre simultaneamente nos dois sentidos.

Half-duplex: sistema de comunicação em que a transmissão e informação, ocorre à vez em apenas um sentido.

100)

- a) SPI, RS232C, USB 3.x
- b) I2C, CAN, USB 1.0 e 2.0

101) Orientação ao bit: info. organizada em tramas; As tramas são constituídas por um símbolo de sincronização seguido de uma sequência de bits.

Ex. CAN; USB

Orientação ao byte: O envio de um byte é uma op. atómica. Poder ser enviados bytes que não são informação de dados servem para o receptor estruturar a info. recebida.  
Ex. I2C

### Protocolo SPI

102)

- a) Master-slave com ligação ponto-a-ponto.
- b) Comunicação síncrona (relógio explícito do master)
- c) Comunicação bidirecional full-duplex.

103)

$$a) \left( \frac{1}{16 \times 10^3} \right) \cdot 20 = 160 \mu s \times 20 = 3200 \mu s = 3,2 ms$$

$$b) \left(8 \times \frac{1}{20 \times 10^3}\right) \times 8 = (8 \times 50 \mu s) \times 8 = 3,2 \text{ ms}$$

$$c) \left(8 \times \frac{1}{100 \times 10^3}\right) \times 10 = (8 \times 0,01, 10^{-3}) \times 10 = 8 \times 10 \mu s \times 10 = 80 \mu s \times 10 = 0,08 \text{ ms}$$

$$d) \left(8 \times \frac{1}{50 \times 10^3}\right) \times 30 = (8 \times 0,02 \times 10^{-3}) \times 30 = 160 \mu s \times 30 = 4,8 \text{ ms}.$$

106) Ponto-a-ponto síncrono.

105) Não, o protocolo SPI é adequado para ligações a curta distância apenas (usuário principalmente em embedded systems).

106) A saída SDO do master liga-se à entrada SDI do slave0, a saída SDO do slave0 liga-se à entrada SDI do slave1... a saída SDO do slave2 liga-se à entrada SDI do master. Como o conjunto dos slaves é visto pelo master como um único dispositivo, o conjunto dos 3 slaves têm de ter no max 8 bits (dimensão do master).

- 107)
- 1) Configurar a frequência do clk
  - 2) Especificar qual o flanco do relógio usado para a transmissão (a regras é feita no flanco oposto).

12C

108)

a) 0110101

b) leitura

c) 1 Ack (o primeiro)

d) 1 Ack (o segundo)

e) 1 nAck (penultimo bit) gerado pelo master

f) 00000010\_10011001 = 0x029B

g) Clock stretch → o slave pode fugar o alongamento da transferência mantendo a zero a linha SCL.

Temos 1 situação de clk stretching gerada pelo slave.

$$h) T = \frac{1}{1 \times 10^6} = 1 \mu s \quad T_{total} = 29 \mu s.$$

sem contar o start-bit

109) O endereçamento é feito nos 7 bits mais significativos da trama a enviar, seguido de um bit que sinaliza o tipo da operação. Cada slave vai ler o endereço e se os endereços correspondem, o slave vai ler o bit da operação p/ saber se vai receber um transmitir dados. Por fim, o slave gera um acknowledge e inicia-se a transf. de info.

110) Um barramento I2C é composto p/ 2 linhas físicas, SDA (serial data out) e SCL (serial clock line)

111) Um barramento está livre quando não estão a ser feitas transf., ou seja, após um STOP bit.

112) O master controla a linha SCL.

113) Na gestão de acesso ao barramento é necessário primeiro garantir que os clocks dos masters estão sincronizados recorrendo à técnica bit dominante/recessivo. Depois, por cada bit enviado, quando a linha SCL está a '1' cada master lê a linha SDA e verifica se o seu valor coincide com o que enviou. O processo de arbitragem é permitido por um master quando lê da linha "0" tendo enviado "1".

114) Nível lógico '0' chama-se bit dominante  
Nível lógico '1' chama-se bit recessivo

Na presença do nível lógico '0' no barramento é forçado a '0' independentemente do que lá estava, sendo p/ isso chamado bit dominante.

115) O slave pode forçar o alongamento da transf. mantendo a zero a linha SCL.  
O sinal "wait" condiciona o estado da linha SCL do barramento, enquanto wait = '1'  $\Rightarrow$  SCL = '0' assim slaves mais lentos garantem que a transf. é sempre concluída c/ sucesso.

## RS232

116) Ligação ponto-a-ponto. Na sua forma mais simples requer a utilização de 2 linhas de sinalização e uma de massa.

- b) Comunicação bidirecional, full-duplex.
- c) Comunicação assíncrona c/ relógio implícito.
- d) Bit orientado

117) Erro de paridade: Quando o bit de paridade, por exemplo, devia ser '0' e é '1' (ou como '1').

Erro de framing: Quando é detectado o nível lógico '0' no instante em que era esperado um stop bit (nível lógico '1')

118) Para diminuir o erro de fase é utilizado uma técnica que consiste em gerar no receptor um relógio com uma freq. N vezes superior ao relógio do transmissor e sincronizar a receção a partir desse relógio. Por ex. se  $N=16$  o erro de fase max vai ser  $T/16$ , em que  $T$  é o período do relógio do transmissor. O erro de fase mantém-se até ao final da receção da trama corrente, mas os instantes de validação estão bem definidos (num ex. "Start bit" válido ao fim de 8 ciclos de relógios e os restantes bits validados a cada 16 ciclos de relógio).

119) Baute rate 57600  
trama é composta p/ 8 bits c/ paridade par  
2 stop bits  
fator de sobre amostragem é 16.

a)

120) N = 64 ; 38400 bps ; 7 bits sem paridade ; 1 stop bit  
a)

121)

a) 19200 bps  
7 bits de dados  
1 bit de paridade  
1 stop bit  
(1 start bit)  
10 bits total

$$\frac{19200 \times 7}{10} = 13440 \text{ bps.}$$

b)  $\frac{115200 \times 8}{11} = 83781,82 \approx 83782 \text{ bps}$

c)  $\frac{9600 \times 8}{12} = 6400 \text{ bps}$

d)  $\frac{1200 \times 7}{9} = 933,33 = 933 \text{ bps}$

122) 100000 bps

8 data bits  
paridade ímpar  
1 stop bit

0x5A

01011010



123) 7 bits de dados  
paridade par  
2 stop bits

8 data bits  
sem paridade  
1 stopbit  
Rx

Não vai ser detectado nenhum erro no receptor.

0x2D,

124) 8 data bits  
ímpar  
1 stopbit  
Tx

7 data bits  
s/ paridade  
1 stop bit  
Rx

Vai ser detectado um erro de framing, e é detectado um '0' onde deveria ser detectado um stop bit ('1').

125) 7 data bit  
par  
2 stop bits  
Tx

## ~~QUESTION~~

### Device Drivers

126)

a) O device driver é um programa que permite a outro programa/app/so interagir com um dispositivo hardware. Para isso implementa uma camada de abstração que lida com as particularidades do dispositivo controlado.

b) A função do kernel Linux é traduzir as chamadas da aplicação ou SO em ações específicas do dispositivo. Posto isto, cria-se uma interface entre o hardware e a aplicação/SO.

127) O device driver tem de lidar com aspectos específicos da implementação do dispositivo por isso o seu fornecimento é garantido pelo fabricante. A biblioteca de classes para comunicação tem de ser compatível para que não haja necessidade de alterar a aplicação.

128)

a) Um buffer circular é um array dinâmico c/ variáveis de acesso definidas, que são incrementadas ou decrementadas em cada iteração. Um FIFO pode ser implementado c/ um buffer circular.

b) Head, tail e count. A primeira aponta p/ o caractere que está no buffer à mais tempo, pronto a ser transmitido. A segunda aponta p/ a primeira posição livre do buffer, onde vai ser escrita a informação nova. A variável count expressa o número de caracteres presente no buffer.

c) A variável count é considerada um recurso partilhado pois ao contrário das variáveis head/tail que têm de ser atualizadas na rotina ou na função, dependendo do sentido da transferência, a variável count tem de ser atualizada a cada iteração para que aquela rotina quer a função tenham a info. do estado do buffer atualizado. Para que o incremento desta variável ocorra dentro da normalidade, esta operação tem de estar numa seção crítica do código.

129) As seções críticas servem p/ garantir que o incremento da variável count (que ocorre na função) é uma operação atómica. Desta forma procede-se à desativação das interrupções, para que quando ocorrer este incremento a operação não seja interrompida para executar outra zona de código que também possa alterar o valor dessa variável. Após ocorrer o incremento, ativa-se de novo as interrupções e finaliza a seção crítica.

## ~~QUESTION~~

### CAN

130)

a) Barramento multi-master. Muitas linhas físicas limitadas pelo número possível de identificadores diferentes.

b) Bidirecional half-duplex

c) Assíncrona com relogio implícito

d) Bit oriented

131) A técnica de bit-stuffing é utilizada p/ garantir um tempo máximo entre transições da linha de dados, assegurando que os relógios locais estão sincronizados.

Consiste em introduzir um bit de polaridade oposta, a cada 5 bits iguais. Este bit é ignorado pelo receptor.

132) "Identifier" é responsável por identificar a mensagem transmitida. Este campo é também utilizado p/ fazer a arbitragem do acesso ao barramento, sendo o id mais baixo o de maior prioridade.

133) Qualquer nó pode desempenhar o papel de master mas não podem ser enviadas 2 mensagens em simultâneo, para evitar estas situações há o processo de arbitragem que garante que o barramento é entregue a um master a cada vez.

134) A arbitragem é feita analisando o campo identifier bit-a-bit, o identificador de menor valor ganha o processo.

135) Todos, porque a comunicação é do tipo broadcast.

136) CRC error:  $\text{CRC calculado} \neq \text{CRC recebido}$ .

Acknowledge error: o produtor não recebe um bit dominante no campo ACK, o que significa que a mensagem não foi recebida p/ nenhum nó da rede.

Form error: verificação que analisa campos que devem estar sempre a '1'.

Bit error: Cada bit transmitido é analisado pelo produtor da mensagem; se o produtor lê um valor que é o oposto do que escreveu gera um erro.

Stuffing error: Se após 5 bits iguais não surgir um bit polaridade oposta, é gerado um erro.

137) máscara = 0111 1111 1010  
filtro = 0101 1100 0000

138)

139) Data frame: usado no envio de dados de um produtor para os receptores.

RTR Frame: Enviado a '1' por um nó consumidor a solicitar uma transmissão de uma trama de dados específica.

Error frame: Usado para reportar um erro detectado.

Overflow frame: usada para atrair o envio da prox. trama em caso de sobreexigência.

VET

140)

- |    | Largura de banda  | garantia entrega (?) |
|----|---|----------------------|
| a) | Não periódica, sem garantia de taxa de transmissão ou latência. Info. a transmitir de pequenos dimensões. Utilizado pelo host-controller p/ envio de mensagens e comando e configuração e consulta de estado. |                      |
| b) | Não periódica, sem garantia de taxa de transmissão e/ou latência ou largura de banda. Garante a entrega c/ mecanismo de retransmissão em caso de detecção de erros. Utilizado em discos/hd-arras...           |                      |
| c) | Periodicas com limite máximo p/ latência, largura de banda garantida. Erros não são corrigidos, não há controle. Utilizado em aplicações de áudio/vídeo.  |                      |
| d) | Periódica c/ limite max. p/ latência. Retransmissão em caso de erro e é utilizada p/ transmitir baixas quantidades de informação. Ratos, teclados...  |                      |

141) Ligação física em árvore com um máximo de 7 níveis e de 127 dispositivos, partindo de um sistema central p/ vários dispositivos.

142)

a) Master-Slave

b) USB 1.0 e 2.0 c/ half-duplex  
" 3.0 full-duplex

c) Codificação: NRZI

Os dados são transmitidos em modo diferencial em paralelo.

d) Comunicação síncrona orientada ao bit.

143) Tanto pode ser um hub e disponibilizar acessos para outros dispositivos, ou então, pode ser um extremo, ou seja, o dispositivo encarado como um simples periférico.

144) Tarefa permanente que permite ao host-controller reconhecer e inicializar um dispositivo (Atribuição de um endereço, leitura de info., alocação de recursos, atribuição e carregamento do Device-Drive)

145) Endereço do dispositivo, identificador do dispositivo e pode ir de 1 a 127.

146) Número de endpoint, corresponde ao terminal lógico associado e pode ir de 0 a 15.

Dirigido, in ou out.

146) Pipes, estabelecem a comunicação entre o host controller e um endpoint. Permite comunicação bidirecional de forma a conectar o host c/ o dispositivo.

### Tecnologia, Organização e Funcionamento da RAID

147) SRAM:

Pelo menos uma ordem de grandeza mais rápido

Não necessita de refreshamento constante

Custo por bit elevado

Maior dissipação de potência

DRAM:

Mais lenta

Necessita de refreshamento

Custo p/ bit menor

→ Menor dissipação de potência.

148) A organização em matriz aumenta o número de descodificadores mas reduz o número total de portas lógicas. Cada palavra pode ser vista como um elemento da matriz e não como uma linha ou memória como na organização linear.

149) Tempo total da operação de leitura.

150)

- a) Tempo necessário para os dados ficarem disponíveis no barramento de dados.
- b) Taxa a que os dados podem ser transferidos de/para memória.

151)

a)  $\frac{128K}{32K} = 4 \times \frac{8}{1} = 32$  circuitos

b)  $\frac{128K}{32K} = 4 \times \frac{8}{4} = 8$  circuitos

c)  $\frac{128K}{16K} = 8 \times \frac{8}{8} = 8$  circuitos

d)  $\frac{128K}{64K} = 2 \times \frac{8}{8} = 2$  circuitos

e)  $\frac{128K}{128K} = 1 \times \frac{8}{1} = 8$  circuitos

152)  $256K = 2^18$

$2^9$  linhas  $\times$   $2^9$  colunas =  $256K$  células

512 linhas, 512 colunas, 8 matrizes

153)

a)  $16\pi \times 4$

b)  $8\pi \times 8$

c)  $2\pi \times 32$

d)  $2\pi \times 32$

e)  $8\pi \times 8$

154)  $512\pi = 2^{29}$  (Total)  $2^9 - 1 = 15$  bits p/ colunas  
 $16K = 2^{14}$  (linhas)

$\max(14, 15) = 15$  bits,

b)  $256\pi = 2^{28}$  (Total)  $2^8 - 1 = 14$  bits (colunas)  
 $16K = 2^{14}$  (linhas)  $\Rightarrow$

c)  $4G = 2^{32} \Rightarrow 16$  bits p/ colunas e linhas

d)  $1G = 2^{30}$  (Total)  
 $32K = 2^{15}$  (linhas)  $\Rightarrow 15$  bits p/ colunas

e)  $2G = 2^{31}$  (Total)  
 $64K = 2^{16}$  (linhas)  $\Rightarrow 15$  bits p/ colunas

barramento =  $\max(15, 16) = 16$  bits

$$f) 256\pi = 2^{28} \\ 8\text{K linhas} = 2^{13} \Rightarrow 15\text{ bits p/ colunas} \quad \text{Barramento} = \min(13, 15) = 13\text{ bits}$$

155)  $27 = 2^7$  (total)  
 $2K = 2^10$  (colunas)  
 linhas =  $2^{14} = 16K$

a)  $1024 \times (50+25) = 76,8\text{ ms}$

b)  $1024 \times (40+15) = 1K \times 55\text{ ns} = 56,32\text{ ns}$

c)  $1024 \times (65+30) = 1K \times 95 = 97,28\text{ ms}$

156)

a)  $8\pi = 2^{23}$ ;  $1024 = 2^{10} \Rightarrow \text{linhas} = 2^{13} = 8K$

$8192 \times (50+25) = 8K \times 75\text{ ms}$

b) 16K linhas

$16K \times (50+25)$

c)  $4\pi = 2^{22}$ ;  $4096 = 2^{12} \Rightarrow \text{linhas} = 2^{10} = 1024$

$1K \times (50+25)$

d)  $8\pi = 2^{23}$ ;  $1024 = 2^{10} \Rightarrow 2^{13} \text{ linhas} = 8K$

$8K \times (50+25)$

157) 1- Pré-carregar a linha "bit" a  $VDD/2$

2- Ativar a linha seletor

3- Valor lógico detectado pela diferença de tensão na linha bit, relativamente a  $VDD/2$

4- Restaurar o valor do condensador.

158) (Verificar sinal "WE")

Litura em page mode

Memória cache.

159)

a) (mapeamento direto) 1 comparador

b) - 1 comparador

c) parcialmente associativa (assoc. 4) 4 comparadores

d) - - (assoc. 8) 4 comparadores

e) (totalmente associativa) 1 comparador p/ linha

$$\text{n linhas} = \frac{\log_2(256K)}{\log_2(256)} = \frac{2^{18}}{2^8} = 2^10 = 1024 \text{ comparadores}$$

160) bus add = 32 bits

a) bloco 64 bytes  $\Rightarrow$  byte =  $\log_2(64) = 6$  bits  
 $256$  linhas  $\Rightarrow$  group =  $\log_2(256) = 8$  bits

$$\text{Tag} = 32 - 8 - 6 = 18 \text{ bits}$$

b) blocos 128bytes  $\Rightarrow$  byte = 7 bits Tag =  $32 - 8 - 7 = 17$  bits  
256 linhas  $\Rightarrow$  group = 8 bits

c) blocos 64 bytes  $\Rightarrow$  byte = 6 bits Tag =  $32 - 8 - 6 = 18$  bits  
 $\log_2(64k) = \frac{2^{10}}{4} = 256$  linhas  $\Rightarrow$  group = 8 bits  
 $\log_2(64)$

c) blocos 64 bytes  $\Rightarrow$  byte = 6 bits Tag = 18 bits  
 $\frac{\log_2(128k)}{\log_2(64)} = \frac{2^{11}}{2^6} = 2^5/8 = 256$  linhas  $\Rightarrow$  group = 8 bits

e) blocos de 256 bytes  $\Rightarrow$  byte = 8 bits Tag =  $32 - 8 = 24$  bits

161) Na cache totalmente associativa, qualquer bloco da mem. principal pode residir em qualquer posição da cache, enquanto que numa cache é mapeamento direto apenas um bloco da um bloco grupo pode residir na cache.

Por outro lado a cache é mapeamento direto de muitas menos comparações, sendo a implementação mais simples e barata.

162) FIFO  $\rightarrow$  e' subs. o bloco que está na cache há mais tempo

LRU  $\rightarrow$  e' subs. o bloco que está há mais tempo sem ser referenciado

LFU  $\rightarrow$  e' subs. o bloco menos utilizado.

Ranum  $\rightarrow$  subst. aleatória.

163) Write-Through: Todas as escritas são realizadas simult. na cache e na memória principal.

Write-Back: Valor escrito apenas na cache, só é feita a escrita na memória quando o bloco da cache foi substituído.

164) O dirty bit sinaliza se um bloco da mem. foi ou não modificado. Assim quanto um bloco da memória é subst., o dirty bit é verificado para determinar se é necessário ou não escrever o bloco na mem. principal antes da sua subst. ou se pode ser ignorado.

165)

a) linhas =  $\frac{\log_2(512k)}{\log_2(32)} = \frac{2^{10}}{2^5} = 2^5/4$  linhas =  $256 = 2^8$ ; group = 8 bits.

b) linhas =  $\frac{2^{10}}{2^6} = 2^4/8 = 128 = 2^7$ ; group = 7 bits

c) linhas =  $\frac{2^{11}}{2^6} = 2^5/4 = 2048 = 2^{11}$ ; group = 11 bits.

d) linhas =  $\frac{2^{10}}{2^7} = 2^3/4 = 8 = 2^3$ ; group = 3 bits

166) O valid bit indica se a linha correspondente tem conteúdo válido ou se não está ocupada. Para uma cache com N linhas são necessários N valid bits.

167) O princípio da localidade consiste no acesso à memória da forma não aleatória, ou seja, os programas usam endereços que se situam vizinhos.

Localidade temporal: Endereços aceitáveis, têm mais probabilidade de serem novamente aceitáveis num futuro próximo.

Localidade espacial: Endereços contíguos ou aceitáveis também têm mais probabilidade de serem aceitáveis.

168) a)  $0,9 \times 6\text{ns} + 0,1 \times 66\text{ns} = 12\text{ns}$

b)  $0,95 \times 4\text{ns} + 0,05 \times 39\text{ns} = 5,75\text{ns}$

c)  $0,85 \times 5\text{ns} + 0,15 \times 55\text{ns} = 12,5\text{ ns}$

169)

a)  $0,9 \times 2 + 0,1 \times 102 = 12$  cicles

b)  $0,95 \times 2 + 0,05 \times 82 = 6$  cicles

c)  $0,85 \times 4 + 0,15 \times 125 = 22$  cicles

170) Parcialmente associativa 8KB

associat. ↴

blocos 32 bytes  $\Rightarrow$  5 bits (byte)

a)  $0x12B8 = 0001\underset{\text{byte.}}{\overbrace{00101011}}\underset{\text{1}}{\overbrace{1000}}$

$$\frac{2^8}{2^5} = \frac{2^8}{2^4} = 2^4 \Rightarrow 6 \text{ bits (sbt)} \quad 010101_2 = 15_{10} = 1 \times 16 + 5 \times 16 = 21_{10}$$

b)  $0x355F = 0011\underset{\text{1}}{\overbrace{01010101}}\underset{\text{1}}{\overbrace{1111}}$

$$\hookrightarrow 101010_2 = 2A_{10} = 2 \times 16 + A \times 16 = 42_{10}$$

c)  $0x2760 = 0010\underset{\text{1}}{\overbrace{01110110}}\underset{\text{1}}{\overbrace{0000}}$

$$\hookrightarrow 110110_2 = 3B_{10} = 3 \times 16 + 11 \times 16 = 48 + 11 = 59_{10}$$

171) bloco 32 bytes  $\Rightarrow$  5 bits (byte)

$$\frac{2^{16}}{2^5} = \frac{2^9}{2^3} = 2^6 \Rightarrow 6 \text{ bits (sbt)}$$

a)  $0x56B9 = 0101\underset{\text{1}}{\overbrace{01010101}}\underset{\text{1}}{\overbrace{1001_2}}$

$$\hookrightarrow 110101_2 = 35_{10} = 3 \times 16 + 5 \times 16 = 48 + 5 = 53_{10}$$

b)  $0x7041 = 0111\underset{\text{1}}{\overbrace{00000101}}\underset{\text{1}}{\overbrace{0001_2}}$

$$\hookrightarrow 02_{10} = 0 \times 16 + 2 \times 16 = 2_{10}$$

c)  $0x23F2 = 0010\underset{\text{1}}{\overbrace{00111111}}\underset{\text{1}}{\overbrace{0010_2}}$

$$\hookrightarrow 1F_{10} = 1 \times 16 + 15 \times 16 = 31_{10}$$

172) Campo Tag

173)  $\log_2(\text{linhas})$ .

174) Add Bus = 16 bits

a) bloco 16 byte  $\Rightarrow$  4 bits (byte)

$$\text{linhas} = \frac{2^{16}}{2^4} = \frac{2^{12}}{2^2} = 2^{10} = 1024 \text{ linhas} \Rightarrow \text{group} = 10 \text{ bits}$$

$16 - 10 - 4 = 2$  bits (tag).

b) byte = 8 bits  
 group:  $\frac{2^7}{2^3} = \frac{2^7}{8} = 256 \Rightarrow 8 \text{ bits}$

$$\text{Tag} = 16 - 8 - 6 = 2 \text{ bits}$$

c) byte = 5 bits  
 group:  $\frac{2^{10}}{2^5} = \frac{2^{10}}{32} = 2^5 = 32 \text{ bits}$

$$\text{Tag} = 16 - 11 - 5 = 0$$

d) byte = 5 bits  
 group:  $\frac{2^{20}}{2^5} = \frac{2^{20}}{32} = 2^5 = 32 \text{ bits}$

$$\text{Tag} = 16 - 11 - 5 = 0$$

175) Add bus = 16 bits

a) n: 16 colunas  $\Rightarrow 4$  bits      | dimensão:  $2^4$  (colunas)  $\times 2^4$  (linhas) = 1K,  
 k:  $16 - 4 = 12$  bits

b)  $0x3785 = \underbrace{0011\ 0111}_{\hookrightarrow 5}, \underline{1000}, 0101 \Rightarrow \text{hit (19)}$

$0xF0A3 \Rightarrow \text{hit (C1)}$

$0xFB7 \Rightarrow \text{miss}$

$0x1932 \Rightarrow \text{miss}$

$0x59E5 \Rightarrow \text{miss}$

$0x6D51 \Rightarrow \text{hit (9E)}$

$0x04CF \Rightarrow \text{hit (5D)}$

Memória Virtual

176) Endereçamento Virtual

a)

dimensão pag Virtual:  $8KB = 2^{13} \Rightarrow VPC = 13$  bits  
 entradas de c/processo:  $256K = 2^{18} \Rightarrow VPN = 18$  bits

$$\text{add bus Virtual} = 13 + 18 = 31 \text{ bits}$$

b)  $4KB = 2^{12} \Rightarrow VPC = 12$  bits  
 $1M = 2^{20} \Rightarrow VPN = 20$  bits

$$\text{add bus virtual} = 32 \text{ bits}$$

c)  $4KB = 2^{12} \Rightarrow VPC = 12$  bits  
 $256K = 2^{18} \Rightarrow VPN = 18$  bits  
 $\underline{30 \text{ bits}}$

d)  $16KB = 2^{14} \Rightarrow VPC = 14$  bits  
 $128K = 2^{17} \Rightarrow VPN = 17$  bits  
 $\underline{31 \text{ bits}}$

177) ATLB é uma memória rápida, semelhante a uma cache, onde se encontram as entradas da tabela mais recentemente utilizada.

Implementada como uma memória associativa com procura paralela.

178)

- a) impossível (não pode dar miss na page table)
- b) possível (A página está na memória, dados não disponíveis no cache).
- c) impossível
- d) impossível
- e) possível (pag. disponível na TLB)

179)  $2^{VPN}$

180) Processo no qual o endereço virtual gerado pelo CPU é traduzido num endereço físico de mem. principal. ( $VPN \rightarrow PPN + offset$ )

181)

a) entradas:  $512K = 2^{19} \rightarrow 0(faz)$

dimensão da pag. =  $32 - 19 = 13$  bits

b)  $2^{17} = 2^{20}$

dim pag. =  $48 - 20 = 28$  bits

c)  $128K = 2^{18}$

dim da pag. =  $30 - 18 = 12$  bits

182)  $2^{VPN}$

183) O valid bit de uma entrada da page table sinaliza que a página pretendida está na memória e que a page table fornece o número da página física.

184) A memória é associativa c/ procura paralela, ou seja, a TLB vai simultaneamente, comparar o VPN com todas as tags das entradas.

185)

185/186) Permite a utilização eficiente da mem. física, numericamente.

187) Quando ocorre um page fault

188) LRU → É subs. a página que está há mais tempo sem ser referenciada. As páginas candidatas têm o referençado bit a 0, e são copiadas para o bloco anterior da subs.

189) Write-back → consiste na escrita em disco página a página, e só quando a página necessita de ser retirada da memória física. O dirty bit indica quanto e que a página foi alterada e necessita de ser escrita em disco aquando da sua substituição.

190) endereçamento virtual: 26 bits

pag. 512 bytes

Page table → endereços multiplicados por 2  
→ 16 bits de dimensão.

a) VPU  $\log_2(512) = 9$  bits

$$VPN = 26 - 9 = 17 \text{ bits}$$

191)

188) LRU → F subs. a página que está há mais tempo sem ser referenciada. As páginas candidatas têm o referençado bit a 0, e são copiadas para o disco antes da substituição.

189) Write-back → consiste na escrita em disco página a página, e só quando a página necessita de ser retirada da memória física. O dirty bit indica quando e se a página foi alterada e necessita de ser escrita em disco quando da sua substituição.

190) endereçamento virtual: 26 bits

pag. 512 bytes

Page table → endereços multiplicados por 2  
→ 16 bits de dimensão.

a) VPC  $\log_2(512) = 9$  bits

$$VPN = 26 - 9 = 17 \text{ bits}$$

191)