# GABARITO DA AD1 - Organização de Computadores 2019.2

1. (2,5) Considere uma máquina com arquitetura semelhante àquela apresentada em aula. Pode-se endereçar no máximo 32 M células de memória onde cada célula armazena uma palavra e cada instrução tem o tamanho de uma palavra. Todas as instruções desta máquina possuem o mesmo formato: um código de operação, que permite a existência de um valor máximo de 180 códigos, e três operandos, que indicam dois endereços de memória e um registrador. Existem 34 registradores.

   a) Qual o tamanho mínimo do REM ? (0,3)
   *REM = Barramento de endereços, este terá a capacidade de endereçar 32M células = N*
   *N = 32M células => $N = 2^{25}$ => e = 25 bits*
   *REM = barramento de endereços =* ***25 bits***

   b) Qual o tamanho mínimo do CI ? (0,3)
   *CI terá o tamanho necessário para endereçar toda a memória.*
   *Tamanho da memória = N = 32M células*
   *CI = $\log_2$ 32 M células =* ***25*** **bits**

   c) Qual o tamanho do barramento de endereços ? (0,3)
   *REM = barramento de endereços* ***= 25 bits***

   d) Qual o tamanho mínimo do RI ? (0,5)
   *RI (registrador de instruções) deverá ter no mínimo o tamanho de uma instrução*
   *Cada instrução = codOper + Operando 1 + Operando 2 + Operando 3*
   *CodOper deverá permitir 180 códigos diferentes (instruções)*
   *=> CodOper = 8 bits que permite até 256 códigos diferentes*
   *Operando 1 e 2 corresponde a 1 endereço de memória = 25 bits*
   *Operando 3 deverá ter o tamanho capaz de endereçar até 34 registradores, necessário, então, ter 6 bits*
   *Instrução = CodOper + Operando 1 + Operando 2 + Operando 3*
   *= 8 + 25 + 25 + 6 = 64 bits*
   *O tamanho mínimo para RI deverá ser* ***64 bits***

   e) Qual a capacidade máxima da memória em bits ? (0,5)
   *N = 32M células*
   *O tamanho da célula (M) deverá armazenar uma palavra, esta terá o tamanho da instrução*
   *M = tamanho de uma palavra = tamanho de uma instrução = 64 bits/célula*
   *T = M x N => T = 64 bits/célula x 32K células => T = 2048 K bits ( 256 K bytes )*

   f) Se a largura do barramento de dados desta máquina for igual à metade do tamanho de uma instrução, como funcionará o ciclo de busca ? (0.6)
   *Serão necessários 2 ciclos de busca para obter uma instrução*

2. (2,0) Considere a máquina apresentada na aula 4. Descreva detalhadamente (do mesmo modo que é apresentado na aula 4) como é realizada a execução das seguintes instruções:

   a) JP 520
   *1. RI ← (CI), ou seja, RI recebe a instrução lida*
   *2. CI ← CI + 1*
   *3. Decodificação do código de operação*
   *- recebe os bits do código de operação*
   *- produz sinais para a execução da operação de salto condicional.*
   *4. Busca do operando*
   *- UC emite sinal para transferir conteúdo acumulador para UAL (UAL ← ACC )*
   *5. Executa operação de comparação*
   *- Se o resultado for verdadeiro, ou seja, ACC > 0*
   *CI ← Operando, ou seja, CI ← 520*
   *6. Inicia o procedimento de leitura da instrução contida no endereço que consta em CI*

b) STR 30

1. *RI ← (CI), ou seja, RI recebe a instrução lida*
2. *CI ← CI + 1*
3. *Decodificação do código de operação*
   - *Recebe os bits do código de operação*
   - *Produz sinais para a execução da operação de escrita.*
4. *Execução da operação*
   - *A UC emite sinais para que o valor do campo operando = 30 seja transferido para o REM*
   - *Conteúdo do Acumulador (ACC) é transferido para o RDM (RDM ← ACC)*
   - *A UC ativa a linha WRITE do barramento de controle*
   - *O REM passa o conteúdo para o barramento de endereços*
   - *O RDM passa o conteúdo para o barramento de dados*
5. *A memória grava o dado recebido pelo barramento de dados no endereço enviado através do barramento de endereços*

3. (1,5) Escreva um programa que utilize as instruções de linguagem de montagem apresentadas na aula 4 para executar o seguinte procedimento. O conteúdo da memória cujo endereço é 100 é lido e verifica-se se o seu valor é 0. Caso seu valor seja 0, o conteúdo de memória cujo endereço é 250 é somado ao conteúdo de memória cujo endereço é 350 e o resultado é armazenado no endereço 500. Caso contrário, o conteúdo de memória cujo endereço é 250 é subtraído do conteúdo de memória cujo endereço é 350 e o resultado é armazenado no endereço 500. Além de apresentar seu programa escrito em linguagem de montagem, apresente também o programa traduzido para linguagem de máquina.

Obs 1: neste gabarito os endereços mencionados no título da questão (100, 250, 350, 500) foram considerados em hexadecimal, entretanto, soluções onde os mesmos endereços tenham sido entendidos como sendo na base 10, também serão considerados quando na correção da AD.

Obs 2: Nas instruções utilizadas na aula 5 o operando possuía 8 bits. Essa quantidade de bits não é suficiente para representar os endereços apontados no título da questão, assim, é necessário adotar um operando com uma maior quantidade de bits. Para esta solução foi adotada uma instrução com 12 bits, sendo 4 bits para o código de operação e 12 bits para o operando

| ***Endereço (hexa)*** | ***Instrução*** | ***Descrição*** | ***Linguagem Máquina (bin / hexa)*** |
|---|---|---|---|
| *00* | *LDA 100* | *ACC ← (100)* | *( 0001 0001 0000 0000 / 1100 )* |
| *01* | *JZ 06* | *se ACC = 0, CI ← 06* | *( 0111 0000 0000 0110 / 7006 )* |
| *02* | *LDA 350* | *ACC ← (350)* | *( 0001 0011 0101 0000 / 1350 )* |
| *03* | *SUB 250* | *ACC ← ACC + (250)* | *( 0100 0010 0101 0000 / 4250 )* |
| *04* | *STR 500* | *(500) ← ACC* | *( 0010 0101 0000 0000 / 2500 )* |
| *05* | *HLT* | *Encerra Procedimento* | *( 0000 0000 0000 0000 / 0000 )* |
| *06* | *LDA 250* | *ACC ← (250)* | *( 0001 0010 0101 0000 / 1250 )* |
| *07* | *ADD 350* | *ACC ← ACC + (350)* | *( 0011 0011 0101 0000 / 3350 )* |
| *08* | *STR 500* | *(500) ← ACC* | *( 0010 0101 0000 0000 / 2500 )* |
| *09* | *HLT* | *Encerra Procedimento* | *( 0000 0000 0000 0000 / 0000 )* |

4. (0,5) Descreva passo a passo as operações de leitura da memória e de escrita na memória, indicando como os registradores RDM e REM são utilizados e como a unidade de controle gera os sinais necessários.

*Passos de uma operação de leitura*
*1) (REM) ← (outro registrador da UCP)*
*1.1) O endereço é colocado no barramento de endereços*
*2) Sinal de leitura é colocado no barramento de controle*
*2.1) Decodificação do endereço e localização da célula na memória*
*3) (RDM) ← (MP(REM)) pelo barramento de dados*
*4) (outro registrador da UCP) ← (RDM)*

*Passos de uma operação de escrita*
*1) (REM) ← (outro registrador)*
*1.1) O endereço é colocado no barramento de endereços*
*2) (RDM) ← (outro registrador)*
*2.1) O dado é colocado no barramento de dados*
*3) Sinal de escrita é colocado no barramento de controle*
*4) (MP(REM)) ← (RDM) O conteúdo do barramento de dados, proveniente do RDM, é armazenado na MP na posição contida no barramento de endereços, este último proveniente do REM,*

5. (1,0) Escreva um programa que utilize as instruções de linguagem de montagem apresentadas na aula 4 para executar o seguinte procedimento. O conteúdo da memória cujo endereço é 60 é lido e verifica-se se o seu valor é menor que 0. Caso seu valor seja menor que 0, o conteúdo de memória cujo endereço é 80 é adicionado ao conteúdo de memória cujo endereço é 60 e o resultado é armazenado no endereço 60. Caso contrário, o conteúdo de memória cujo endereço é 50 é multiplicado por 3 e o resultado é armazenado no endereço 80. Além de apresentar seu programa escrito em linguagem de montagem, apresente também o programa traduzido para linguagem de máquina.

Obs: neste gabarito os endereços mencionados no título da questão (50, 60, 80) foram considerados em hexadecimal, entretanto, soluções onde os mesmos endereços tenham sido entendidos como sendo na base 10, também serão considerados quando na correção da AD.

| ***Endereço (hexa)*** | ***Instrução*** | ***Descrição*** | ***Linguagem Máquina (bin / hexa)*** |
|---|---|---|---|
| *00* | *LDA 60* | *ACC ← (60)* | *( 0001 0110 0000 / 160 )* |
| *01* | *JN 06* | *se ACC < 0, CI ← 07* | *( 0101 0000 0111 / 507 )* |
| *02* | *LDA 50* | *ACC ← (50)* | *( 0001 0101 0000 / 150 )* |
| *03* | *ADD 50* | *ACC ← ACC + (50)* | *( 0011 0101 0000 / 350 )* |
| *04* | *ADD 50* | *ACC ← ACC + (50)* | *( 0011 0101 0000 / 350 )* |
| *05* | *STR 80* | *(80) ←ACC* | *( 0010 1000 0000 / 280 )* |
| *06* | *HLT* | *Encerra Procedimento* | *( 0000 0000 0000 / 000 )* |
| *07* | *ADD 80* | *ACC ← ACC + (80)* | *( 0011 1000 0000 / 380 )* |
| *08* | *STR 60* | *(60) ← ACC* | *( 0010 0110 0000 / 260 )* |
| *09* | *HLT* | *Encerra Procedimento* | *( 0000 0000 0000 / 000 )* |

6. (1,5) Considere uma máquina que possa endereçar 512 Mbytes de memória física, utilizando endereço referenciando byte, e que tenha a sua memória organizada em blocos de 32 bytes. Ela possui uma memória cache que pode armazenar 8K blocos, sendo um bloco por linha. Mostre o formato da memória cache, indicando os campos necessários (tag, bloco) e o número de bits para cada campo, e o formato de um endereço da memória principal, indicando os bits que referenciam os campos da cache, para os seguintes mapeamentos:

a) Mapeamento direto.

***Memória Principal***

⇒ *Tamanho da memória (em bytes) = 512 M bytes, como cada célula é referenciada a 1 byte, temos N = 512 M células*

⇒ *Será organizada em blocos de 32 bytes. Como 1 célula = 1 byte, cada bloco = 32 células, K = 32*

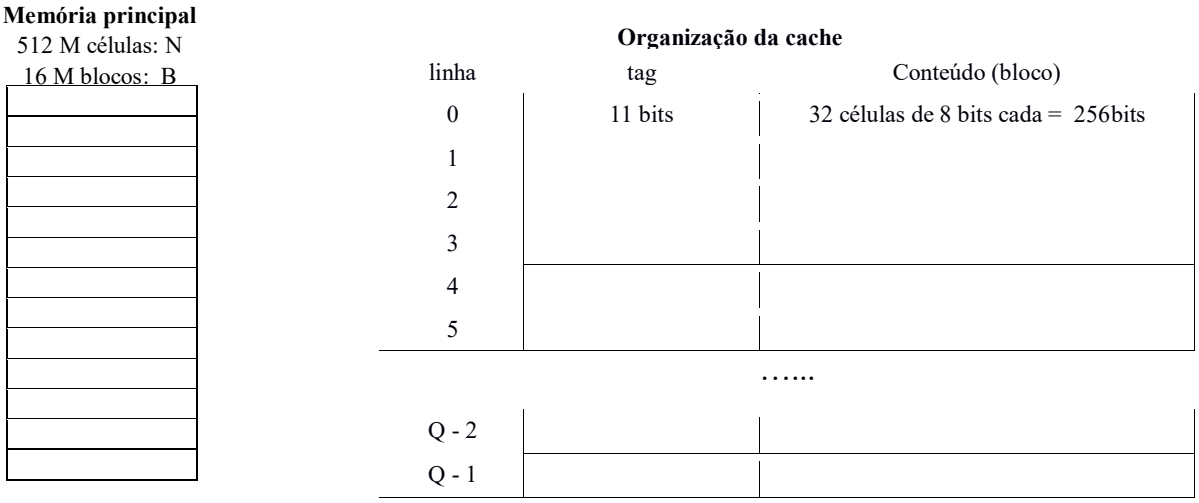
⇒ *Sendo N a quantidade de células da memória e K a quantidade de células por blocos temos:*
*N = 512 M células e K = 32 células / blocos , o total de blocos da MP ( B ) será:*
*Total de blocos: B = N / K => B = 512Mcélulas / 32 células/bloco => B = 16M blocos*

***Memória Cache***

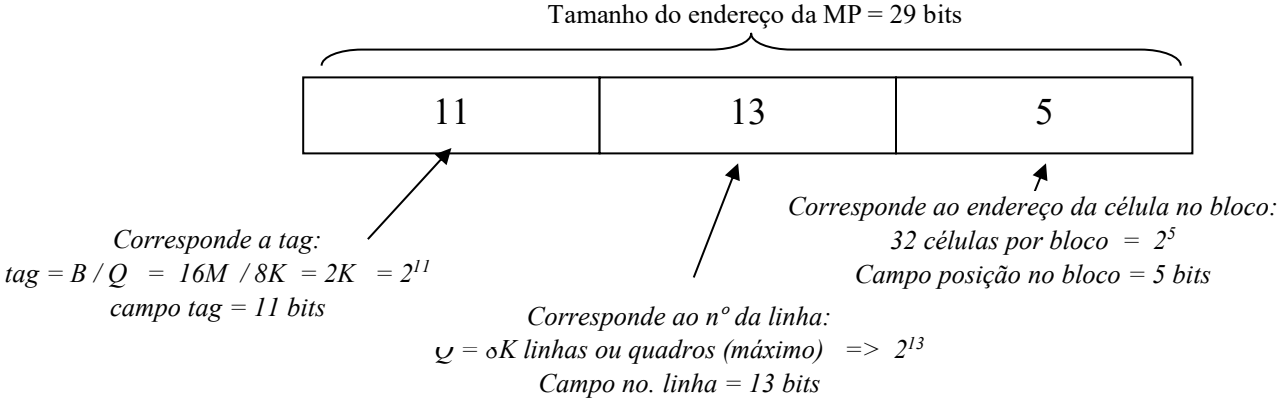
*OBS: O K (quantidade de células/linha) é igual ao do MP.*

⇒ *Tamanho da memória cache em blocos = 8K linhas que podem armazenar 8K blocos*

⇒ *Tamanho da memória cache em células = 8K blocos x 32 células/bloco = 256K células*




*Para endereçarmos toda a MP precisamos de E bits*
*sendo $N = 2^E$ => N = 512M células => $N = 2^{29}$ => E = 29 bits*

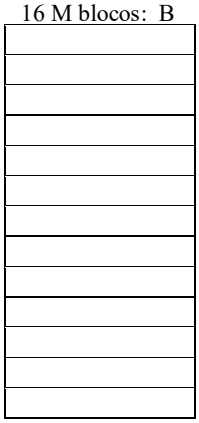


b) Mapeamento totalmente associativo.

***Memória Principal***
*=> N = 512 M células*
*=> K = 32 células por bloco*
*=> B = 16 M blocos*

***Memória Cache***
*OBS: O K (quantidade de células/bloco) tem de ser igual a MP.*
*=> Q =8K linhas ou 8K blocos*
*=> Tamanho da memória cache = 8K células*

**Memória principal**
512 M células: N
16 M blocos: B

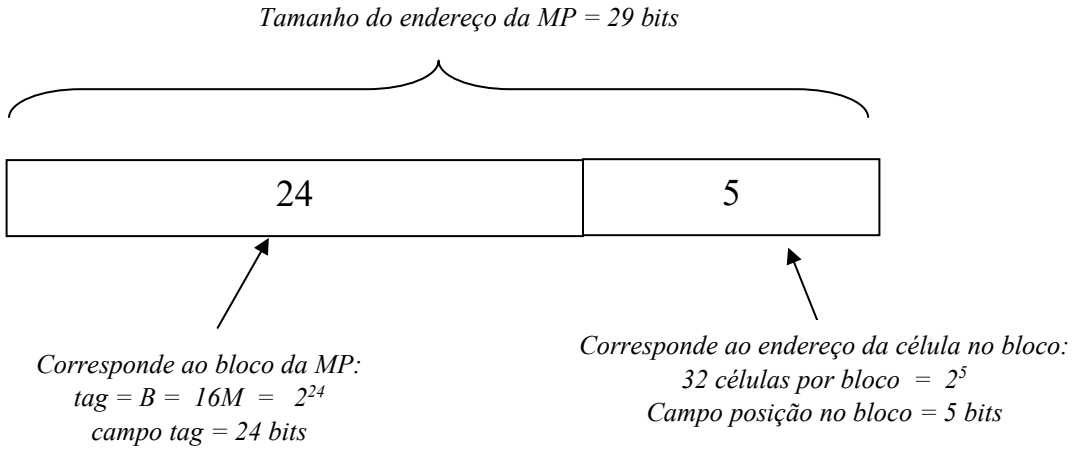


**Organização da cache**

| linha | tag | Conteúdo (bloco) |
|---|---|---|
| 0 | 24 bits | 32 células de 8 bits cada = 256bits |
| 1 | | |
| 2 | | |
| 3 | | |
| 4 | | |
| 5 | | |
| | ..... | |
| Q - 2 | | |
| Q - 1 | | |

*Para endereçarmos toda a MP precisamos da seguinte quantidade de bits: E = 29 bits*

*Como o bloco pode ser alocado em qualquer posição da memória cacha a tag indicará qual dos blocos da MP está alocado naquela posição da memória cachê*



c) Mapeamento associativo por conjunto, onde cada conjunto possui quatro linhas, cada uma de um bloco.

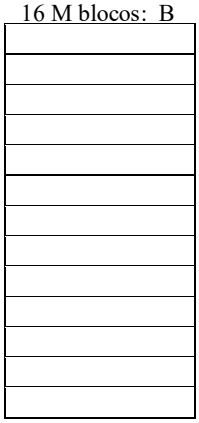
***Memória Principal***

*=> N = 512 M células*
*=> K = 32 células por bloco*
*=> B = 16 M blocos*

***Memória Cache***

*OBS: O K (quantidade de células/bloco) tem de ser igual a MP.*
*=> Q =8 K linhas ou 8K blocos*
*=> Tamanho da memória cache = 8K células*
*=> 1 conjunto = 4 linhas (ou quadros) =>*
*Total de conjuntos => C = 8K células / 4 =>* ***C = 2K conjuntos***

**Memória principal**
512 M células: N
16 M blocos: B

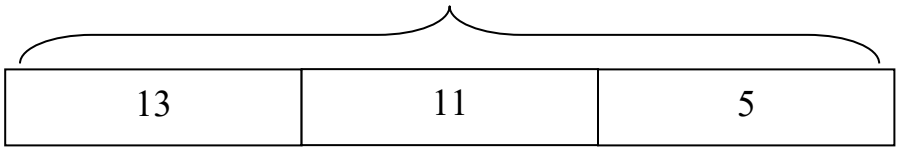


**Organização da cache**

| conjunto | linha | tag | Conteúdo (bloco) |
|---|---|---|---|
| 0 | 0 | 13 bits | 32 células de 8 bits cada = 256bits |
| | 1 | | |
| | 2 | | |
| | 3 | | |
| 1 | 4 | | |
| | 5 | | |
| | | ..... | |
| C - 1 | Q - 2 | | |
| | Q - 1 | | |

*Para endereçarmos toda a MP, necessitamos de 32 bits*

*Tamanho do endereço da MP = 32 bits*




*Corresponde a tag:*
*$B / C = 16 M / 2 K = 8K = 2^{13}$*
*campo tag = 13 bits*

*Corresponde ao nº do conjunto da memória cache:*
*C = 2K conjuntos (máximo) => $2^{11}$*
*Campo no. conjunto = 11 bits*

*Corresponde ao endereço da célula no bloco:*
*32 células por bloco = $2^5$*
*Campo posição no bloco = 5 bits*

7. (1,0) Explique em detalhes a organização hierárquica do subsistema de memória nos computadores atuais

*O subsistema de memória é interligado de forma bem estruturada seguindo uma organização hierárquica. Podemos representar essa organização, conforme as características das memórias, na forma de uma pirâmide cujos níveis são descritos a seguir.*

*No topo da pirâmide temos os registradores, que são pequenas unidades de memória que armazenam dados dentro do núcleo da UCP. Os registradores são dispositivos de maior velocidade com tempo de acesso em torno de 1 ciclo de memória, possui a menor capacidade de armazenamento e também o menor tempo de armazenamento.*

*Em um nível abaixo temos a memória cache, cuja função é acelerar a velocidade de transferência das informações entre UCP e a memória principal (MP), aumentando, assim, o desempenho do sistema. A UCP procura informações primeiro na cache e caso não as encontre, estas serão transferidas da MP para a cache. A cache possui tempo de acesso menor que a da MP, embora seja superior aos dos registradores. A capacidade da cache é bem inferior à da MP, mas possui um tamanho capaz de armazenar uma apreciável quantidade de informações. O tempo de permanência do dado é menor do que o tempo de duração do programa a que pertence.*

*Abaixo da memória cache temos a memória básica de um sistema de computação, que é a memória principal (MP). A MP armazena e disponibiliza o programa (e seus dados) à UCP para que este, durante a execução, busque instrução a instrução. A MP é mais lenta que a cache e mais rápida que a memória secundária, possui capacidade bem superior à da cache e os dados ou instruções permanecem na MP enquanto durar a execução do programa.*

*Finalmente, na base da pirâmide temos a memória secundária (MS), memória auxiliar ou memória de massa, que fornece garantia de armazenamento permanente aso dados e programas dos usuários, mesmo com a perda de energia. Alguns dispositivos são ligados ao barramento de caráter permanente, como o disco rígido, e outros são conectados conforme a necessidade do usuário, como os cartões de memória, CD-ROMs e pen-drives. A MS é a memória mais lenta quando comparada as dos demais níveis, mas possuem a maior capacidade de armazenamento.*