SEL0415
Introdução à Organização de Computadores
Lista 08 – Lógica de Seleção
RESOLUÇÃO

1.

a) Pelo mapeamento, nota-se que a faixa completa de endereços vai de 0000h a FFFFh . Cada dígito em Hexa corresponde a 4 bits. Logo, temos um total de 16 bits de endereço para esse microprocessador.

b) Uma forma rápida de fazer o mapeamento é ir somando as capacidades das memórias (em hexa) e obtendo os respectivos endereços iniciais de cada uma.

$16K \rightarrow \boldsymbol{Início} = 0000h$

$8K \rightarrow \boldsymbol{Início} = 0000h + 16k = 0000h + 4000h = 4000h$

$2K \rightarrow \boldsymbol{Início} = 4000h + 8k = 4000h + 2000h = 6000h$

$2K \rightarrow \boldsymbol{Início} = 6000h + 2k = 6000h + 800h = 6800h$

$4K \rightarrow \boldsymbol{Início} = 6800h + 2k = 6800h + 800h = 7000h$

$Vazio \rightarrow \boldsymbol{Início} = 7000h + 4k = 7000h + 1000h = 8000h$

Uma vez tendo o início de cada memória, para encontrar o fim, basta subtrair 1 do início da memória seguinte. Para a última (vazio), o fim é FFFF.

$16K \rightarrow \boldsymbol{Fim} = 4000h - 1 = 3FFFh$

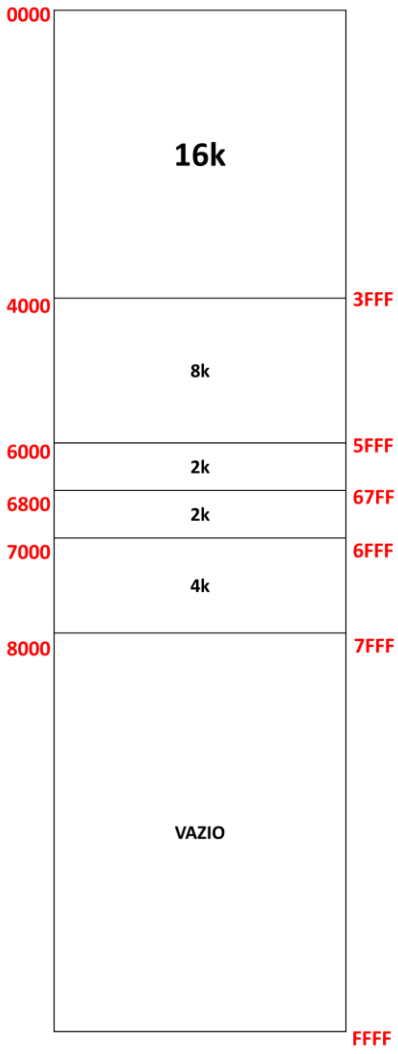
$8K \rightarrow \boldsymbol{Fim} = 6000h - 1 = 5FFFh$

$2K \rightarrow \boldsymbol{Fim} = 6800h - 1 = 67FFh$

$2K \rightarrow \boldsymbol{Fim} = 7000h - 1 = 6FFFh$

$4K \rightarrow \boldsymbol{Fim} = 8000h - 1 = 7FFFh$

$Vazio \rightarrow \boldsymbol{Fim} = FFFFh$



Outra forma é por meio da tabela (abaixo). Note na tabela que as cores diferentes são relacionadas à quantidade de bits de seleção necessários para cada tamanho de memória. Perceba, também, que estes bits nunca mudam de valor para uma mesma memória.

Lógica de Seleção do µP – Linhas de Endereços																	Memória		
Tipo	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Início (H)		Fim (H)
-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000	16k	
	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1			3FFF
-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4000	8k	
	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1			5FFF
-	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	6000	2k	
	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1			67FF
-	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	6800	2k	
	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1			6FFF
-	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	7000	4k	
	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1			7FFF
Vazio	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8000	32k	
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1			FFFF

c) O mapeamento total, com 16 bits, tem o tamanho de:

$$2^{16} = 2^6.2^{10} = 64K$$

Somando-se a capacidade de todas as memórias, temos:

$$16 + 8 + 2 + 2 + 4 = 32K$$

Logo, o espaço vazio tem $64\text{K} - 32\text{K} \rightarrow 32K$

2.

a) Memórias de 4K:

$$4K = 4 * 1024 = 2^2 * 2^{10} = 2^{12}$$

$$\rightarrow 12\ bits\ de\ endereçamento\ (A_{11} - A_0)$$

$$\rightarrow 4\ bits\ de\ seleção\ (A_{15} - A_{12})$$

Memória de 8K:

$$8K = 8 * 1024 = 2^3 * 2^{10} = 2^{13}$$

$$\rightarrow 13\ bits\ de\ endereçamento\ (A_{12} - A_0)$$

$$\rightarrow 3\ bits\ de\ seleção\ (A_{15} - A_{13})$$

Memória de 16K:

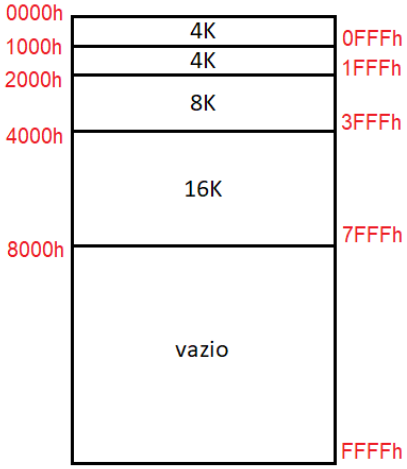
$$16K = 16 * 1024 = 2^4 * 2^{10} = 2^{14}$$

$$\rightarrow 14\ bits\ de\ endereçamento\ (A_{13} - A_0)$$

$$\rightarrow 2\ bits\ de\ seleção\ (A_{15} - A_{14})$$

b)

Memória	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Início	Fim
4K	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000h	
	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1		0FFFh
4K	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1000h	
	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1		1FFFh
8K	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2000h	
	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1		3FFFh
16K	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4000h	
	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		7FFFh
vazio	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8000h	
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		FFFFh



c) Como o pino CS é ativado em nível alto, deve-se usar lógica baseada em portas AND.

A primeira memória de 4K é selecionada quando os bits de seleção $A_{15} - A_{12}$ são ligados ao CS como 0000b, portanto:

$$S = \overline{A_{15}} \cdot \overline{A_{14}} \cdot \overline{A_{13}} \cdot \overline{A_{12}}$$

A segunda memória de 4K é selecionada quando os bits de seleção $A_{15} - A_{12}$ são ligados ao CS como 0001b, portanto:

$$S = \overline{A_{15}} \cdot \overline{A_{14}} \cdot \overline{A_{13}} \cdot A_{12}$$

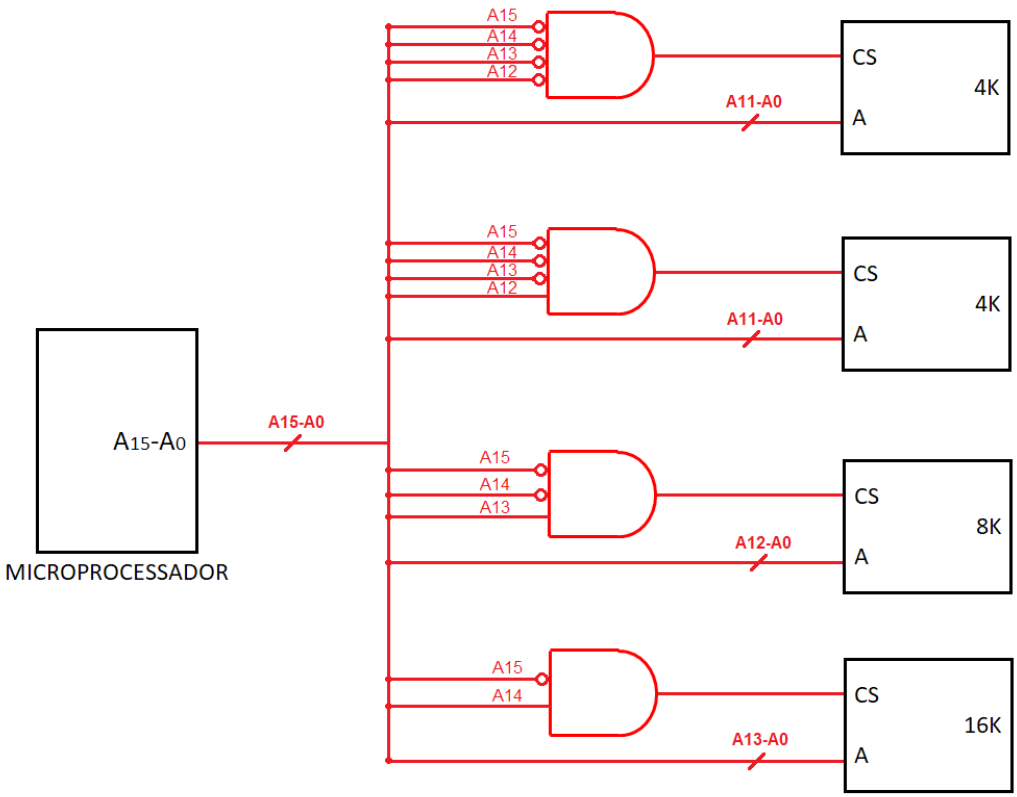
A memória de 8K é selecionada quando os bits de seleção $A_{15} - A_{13}$ são ligados ao CS como 001b, portanto:

$$S = \overline{A_{15}} \cdot \overline{A_{14}} \cdot A_{13}$$

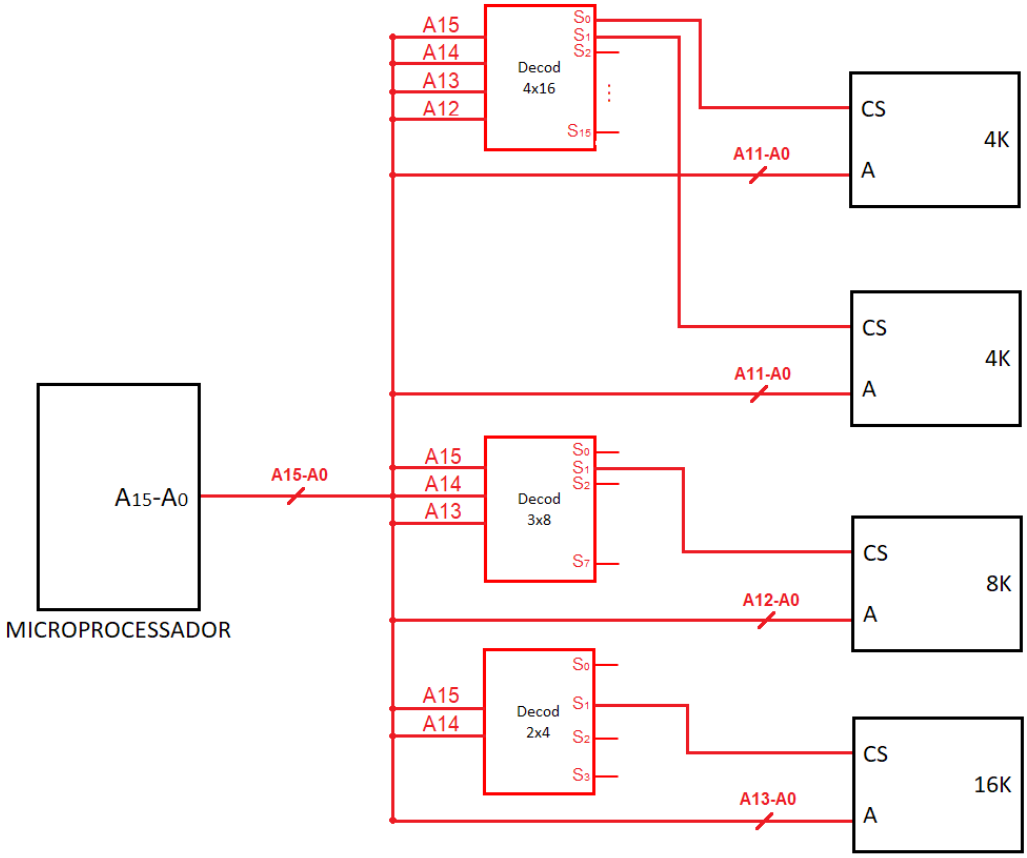
A memória de 16K é selecionada quando os bits de seleção $A_{15} - A_{14}$ são ligados ao CS como 01b, portanto:

$$S = \overline{A_{15}} \cdot A_{14}$$

d)



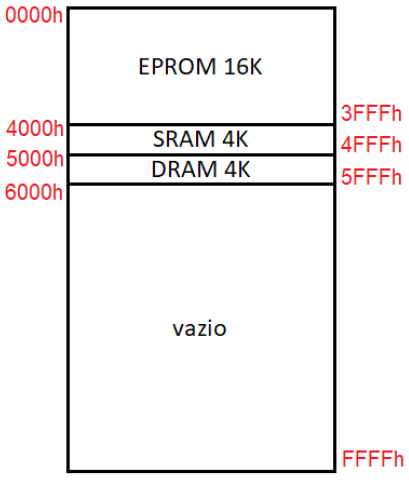
e)



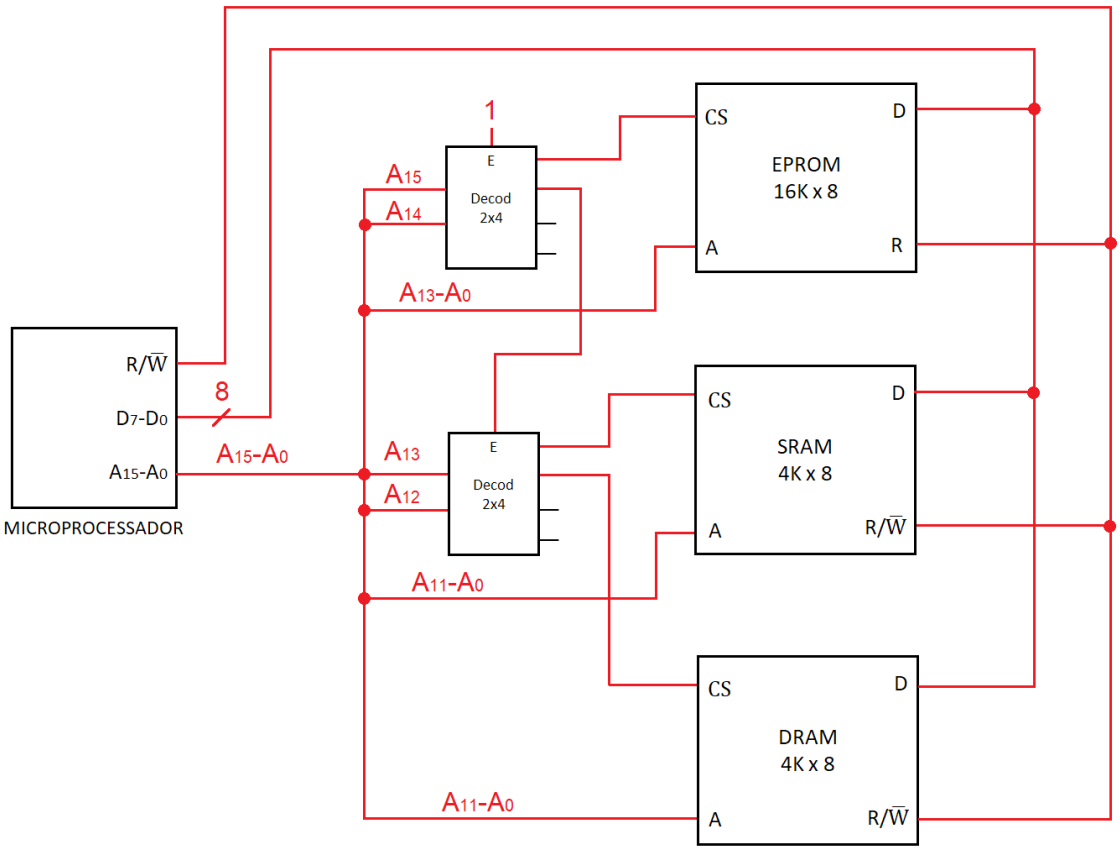
3.

a)

Memória	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Início	Fim
16K	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000h	
	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1		3FFFh
4K	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4000h	
	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1		4FFFh
4K	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	5000h	
	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1		5FFFh
vazio	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	6000h	
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		FFFFh

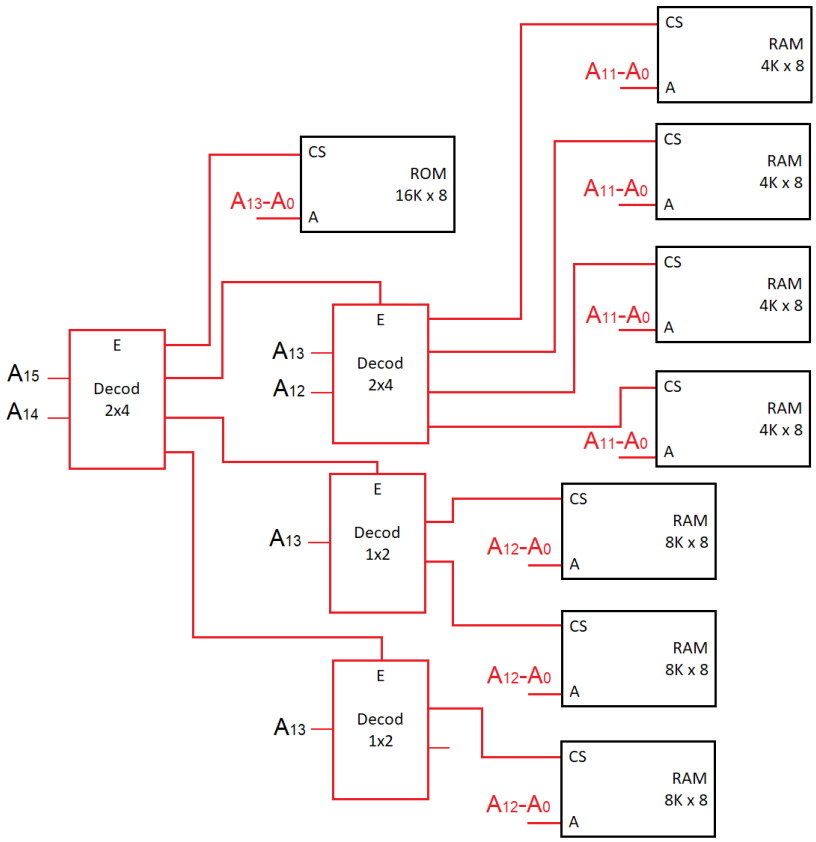


b)

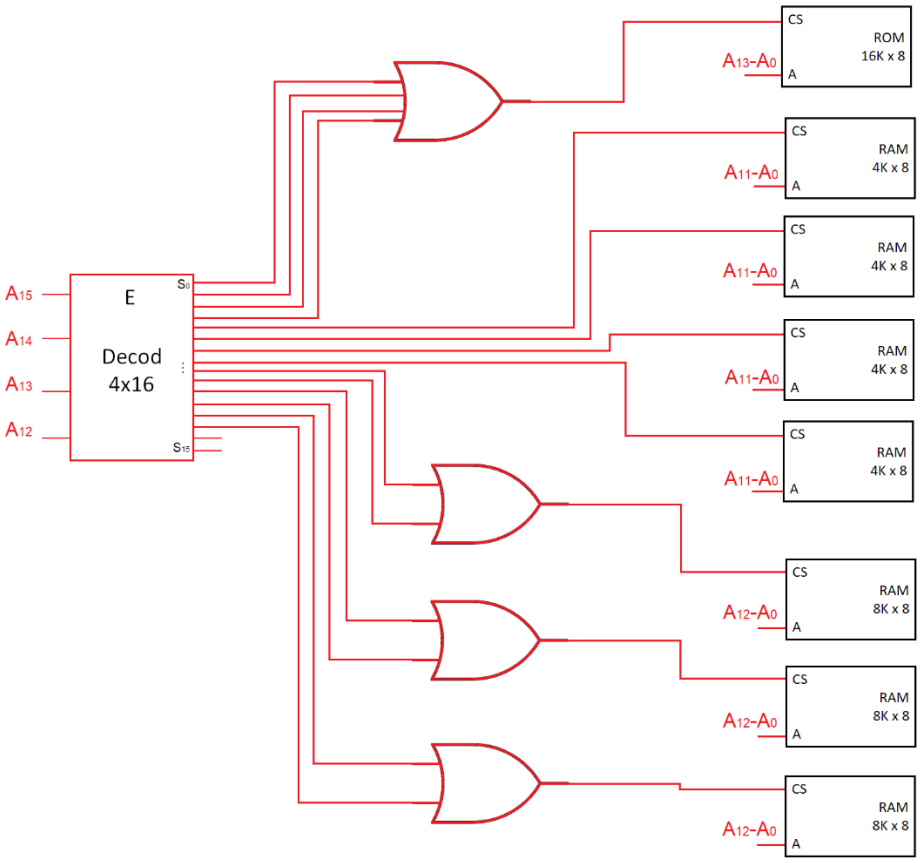

1
E
Decod
2x4
A15
A14
A13-A0
CS
D
EPROM
16K x 8
A
R
R/W
D7-D0
A15-A0
8
MICROPROCESSADOR
A13
A12
A11-A0
SRAM
4K x 8
R/W
DRAM
4K x 8

4.

a)


CS
ROM
16K x 8
A13-A0
A
RAM
4K x 8
A11-A0
E
Decod
2x4
A15
A14
A13
A12
Decod
1x2
RAM
8K x 8
A12-A0

b)



5.

a) Percebe-se, pelo esquemático, que A_{19} é o bit mais significativo. Logo, esse microcontrolador possui 20 bits de endereço.

b) Pensando somente nos endereços, temos:
Microprocessador → $2^{20} = 1M$

$$M1$$

$$Dois\ bits\ de\ seleção\ (A_{19}\ e\ A_{18})\ -\ Decod\ 2\ x\ 4$$

$$Logo, estamos\ dividindo\ o\ total\ por\ 4$$

$$Sendo\ assim \rightarrow M1 = \frac{2^{20}}{2^2} = 2^{18} = 2^8.2^{10} = 256K$$

$$M2\ ,M3, M5\ e\ M6$$

$$Após\ a\ divisão\ no\ primeiro\ decodificador, o\ tamanho\ resultante\ é\ 256K$$

$$Posteriormente, ocorre\ uma\ nova\ divisão\ por\ 4, tendo\ em\ vista\ o\ decod\ 2\ x\ 4.$$
$$Logo, para\ cada\ uma\ delas, o\ tamanho\ é:$$

$$M2 = M3 = M5 = M6 = \frac{256K}{4} = 64K$$

$M4$

$Para\ M4, o\ tamanho\ de\ 64K\ obtido\ anteriormente\ é\ novamente\ dividido\ por\ 2,$
$tendo\ em\ vista\ o\ decodificador\ 1\ x\ 2. Logo:$

$$M4 = \frac{64K}{2} = 32K$$

c) Seguindo o "caminho", temos que:

$$Primeiro\ decodificador \rightarrow Saída\ S_1 \rightarrow A_{19}A_{18} = 01$$
$$Segundo\ decodificador \rightarrow Saída\ S_0 \rightarrow A_{17}A_{16} = 00$$
$$Terceiro\ decodificador \rightarrow Saída\ S_0 \rightarrow A_{15} = 0$$

Com isso, temos:

Lógica de Seleção do µP – Linhas de Endereços																				Memória		
	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Início (H)		Fim (H)
M4	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	40000	32k	
	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1			47FFF

6. Explique a diferença entre decodificação absoluta e não-absoluta

Decodificação absoluta ou não-absoluta se refere à forma como os bits de seleção são ligados às memórias.
Na decodificação absoluta, todos os bits do barramento de endereços do microprocessador são utilizados na lógica de seleção.
Já na decodificação não-absoluta, alguns desses bits não são usados e são deixados soltos.
Assim, como alguns bits são deixados soltos, a variação de sinal 0 ou 1 neles não é relevante para a seleção das memórias. Desta forma, um mesmo local de memória pode ser acessado via dois ou mais endereços diferentes.

7.

(F) Na memória M2 da questão 5, 4 bits são usados para seleção e 15 bits para endereçamento

São usados 4 bits para seleção e 16 bits para endereçamento

(F) Na questão 5, ao escrever um dado no endereço D40FFh, ele será escrito num espaço vazio

O primeiro dígito hexadecimal D representa que:

$$A_{19} = 1, A_{18} = 1, A_{17} = 0, A_{16} = 1$$

Assim, estão habilitadas a saída S_3 do primeiro decodificador 2x4 e a saída S_1 do decodificador 2x4 ligado em cascata, o que significa que será selecionada a memória M6.

(V) Na questão 5, caso fosse desejado utilizar apenas 1 decodificador, ele deveria ser, pelo menos, um decodificador 5x32

(F) No exercício 4b, as memórias possuem 16 bits de endereçamento

As memórias de 4K possuem 4 bits de seleção e 12 bits de endereçamento;

As memórias de 8K possuem 3 bits de seleção e 13 bits de endereçamento;

As memórias de 16K possuem 2 bits de seleção e 14 bits de endereçamento.

(F) A utilização de decodificação não-absoluta deve ser evitada pois ela reduz pela metade o espaço de armazenamento disponível para as memórias, já que um mesmo dado é escrito em dois endereços diferentes.

A decodificação não-absoluta não afeta em nada a capacidade de armazenamento das memórias. A decodificação não-absoluta apenas "multiplica" ou "espelha" outros endereços para um mesmo espaço físico de memória. A capacidade de armazenamento de cada memória não é afetada.

(V) Na ausência de decodificadores muito específicos, podem ser utilizados decodificadores comerciais, em que algumas entradas podem ter seus níveis forçados para funcionarem como desejado

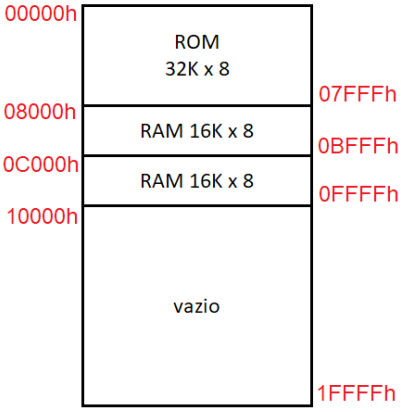
8.

a) Mapa 1:

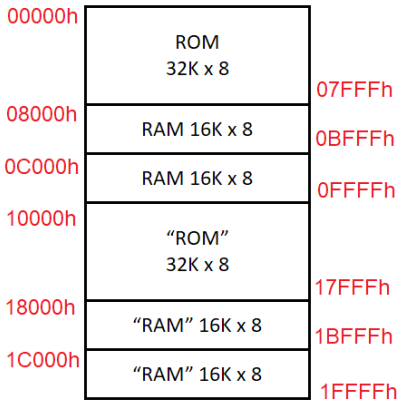
Memória	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Início	Fim
32K	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00000h	
	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		07FFFh
16K	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	08000h	
	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1		0BFFFh
16K	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0C000h	
	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		0FFFFh
vazio	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000h	
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		1FFFFh

Mapa 2:

Memória	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Início	Fim
32K	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00000h	
	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		07FFFh
16K	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	08000h	
	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1		0BFFFh
16K	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0C000h	
	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		0FFFFh
"32K"	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10000h	
	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		17FFFh
"16K"	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18000h	
	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1		1BFFFh
"16K"	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1C000h	
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		1FFFFh



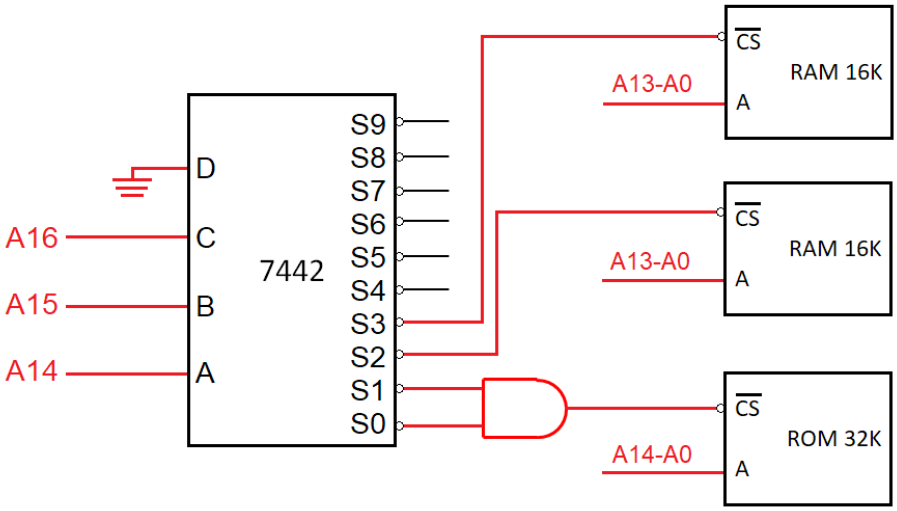
Mapa 1



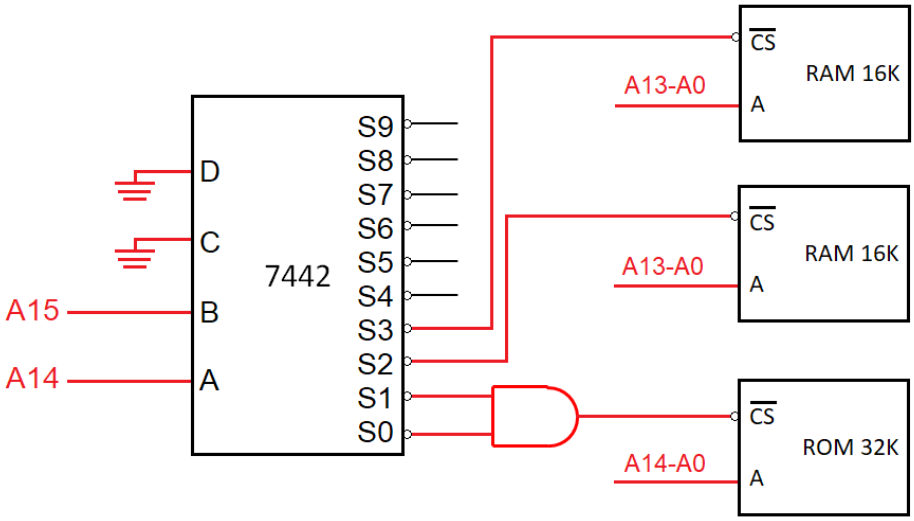
Mapa 2

b)

Mapa 1:



Mapa 2:



Note que no mapa 2, o bit de seleção A_{16} não foi ligado, portanto, isso é o que gera a decodificação não absoluta e faz com que as memórias sejam "espelhadas" entre os endereços em que $A_{16} = 0$ e $A_{16} = 1$

c)

No caso do mapa 1 (decodificação absoluta), nada acontecerá, o dado não será escrito em lugar nenhum, pois será jogado em um espaço vazio.
Já no caso do mapa 2 (decodificação não absoluta), o dado será escrito na segunda memória RAM de 16K, já que o endereço 1DFFFh é um espelho do endereço 0DFFFh, o qual corresponde a um endereço da segunda memória RAM de 16K.
Caso já haja algum dado no endereço 0DFFFh, ele será sobrescrito. Por isso, deve-se ter cuidado para não perder dados importantes quando se utiliza decodificação não absoluta.

Questões 9 e 10: Microprocessador de 16 bits de endereço e 8 bits de dados.

9.

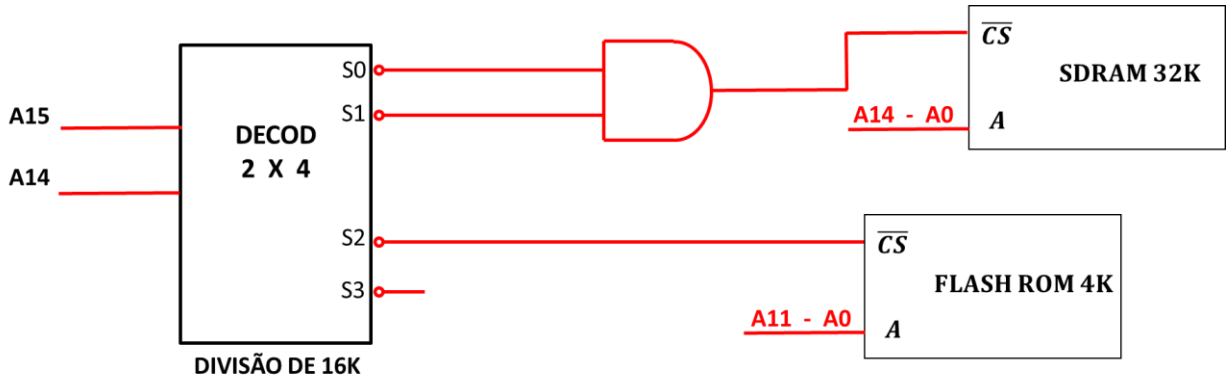
O decodificador irá dividir o total (64K) em quatro, logo teremos 4 espaços de 16k.

Dessa forma, pode-se utilizar as duas primeiras saídas do decodificador, por meio de uma porta AND, para a SDRAM de 32k.

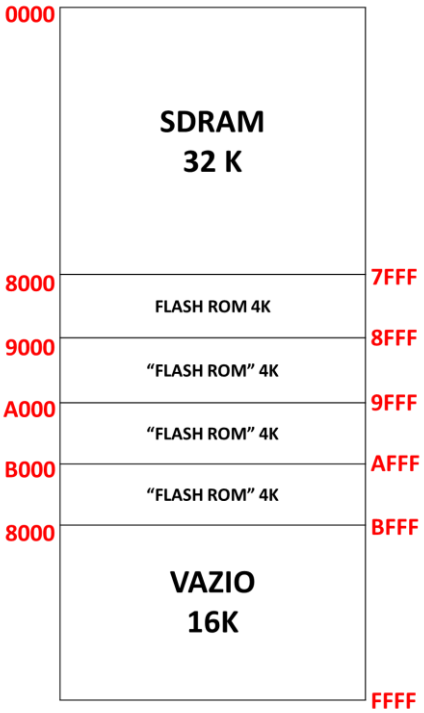
A terceira saída utilizamos para a Flash ROM de 4k. Com isso, como a memória é de tamanho menor ao da saída de seleção do decodificador, teremos a decodificação não-absoluta.

A última saída do decodificador não é ligada a nenhum dispositivo, sendo um espaço vazio de tamanho 16k.

O esboço do circuito para essa seleção fica da seguinte forma:

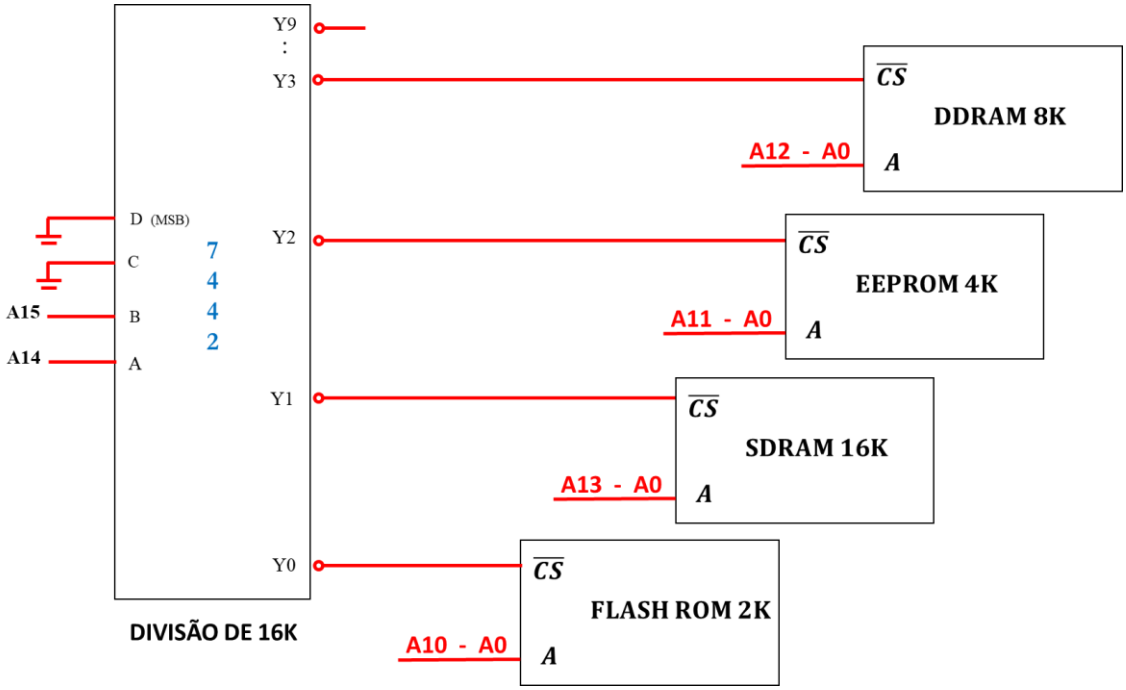


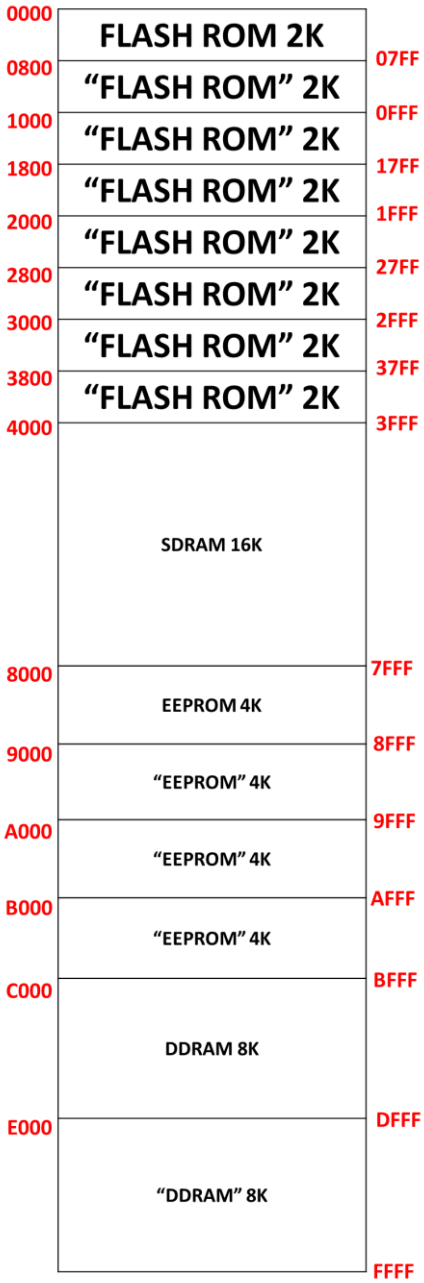
Sendo assim, o mapeamento fica da seguinte maneira:



10.

a) Por se tratar de decodificação não-absoluta, uma forma simples de montar o hardware é utilizando o decodificador 7442 de tal forma que ocorra uma divisão com espaços de 16k, ou seja, um decodificador 2x4. Isso porque temos 4 memórias, em que a maior delas é de 16k.
Cada memória será selecionada por uma saída do decodificador. Com isso, para as memórias que são de tamanho menor que 16k, teremos espaços fantasmas.


Y9
:
Y3
Y2
Y1
Y0
D (MSB)
C
B
A
A15
A14
7
4
4
2
DIVISÃO DE 16K
$\overline{CS}$
DDRAM 8K
A
A12 - A0
$\overline{CS}$
EEPROM 4K
A
A11 - A0
$\overline{CS}$
SDRAM 16K
A
A13 - A0
$\overline{CS}$
FLASH ROM 2K
A
A10 - A0


0000
FLASH ROM 2K
07FF
0800
“FLASH ROM” 2K
0FFF
1000
“FLASH ROM” 2K
17FF
1800
“FLASH ROM” 2K
1FFF
2000
“FLASH ROM” 2K
27FF
2800
“FLASH ROM” 2K
2FFF
3000
“FLASH ROM” 2K
37FF
3800
“FLASH ROM” 2K
3FFF
4000
SDRAM 16K
7FFF
8000
EEPROM 4K
8FFF
9000
“EEPROM” 4K
9FFF
A000
“EEPROM” 4K
AFFF
B000
“EEPROM” 4K
BFFF
C000
DDRAM 8K
DFFF
E000
“DDRAM” 8K
FFFF

b) Por se tratar de decodificação absoluta, temos que conseguir uma divisão de 2k (tamanho da menor memória). Para isso, utilizamos o 7442 primeiramente, com um único bit de seleção (A15), obtendo divisão de 32k. Posteriormente, pegamos a primeira saída (Y0) e conectamos ao pino $\overline{G1}$ do 74154. Por fim, utilizamos os 4 bits (A14-A11) nesse decodificador, de forma que o espaço de 32k seja dividido em 16 partes de 2k cada uma.

Com isso, para as memórias maiores que 2k, basta utilizar combinações com portas AND, de forma a obter o tamanho desejado.