

## Descomposiciones posibles

- 2 a 4
- 3 a 8
- 4 a 16

### Ejercicio 1 – Preguntas

- ¿Cuántas direcciones de memoria puede acceder un procesador genérico que tiene 16 líneas de bus de address y 8 de bus de datos? ¿Cuál es el ancho de la palabra?
- ¿Cuántas direcciones de memoria puede acceder un procesador genérico que tiene 16 líneas de bus de address y 16 de bus de datos? ¿Cuál es el ancho de la palabra?
- ¿Cuántas direcciones de memoria ocupa un dispositivo que tiene 10 líneas de bus de address y 8 de bus de datos?
- ¿Cuántas direcciones de memoria ocupa un dispositivo que tiene 10 líneas de bus de address y 16 de bus de datos?
- ¿Si un dispositivo de 1K x 8 debe mapearse a partir de la dirección 03FF, cómo se debe activar su Chip Select?

$$K = 2^{10}$$

a) 16 líneas BA . } 8 líneas BD . }  $\xrightarrow{\text{puede acceder a}}$

$$2^{16} = 2^{10} \cdot 2^6 = 2^6 K = 64 K$$

Ancho de la palabra = 1 byte (8 bits)

$$64 K \text{ direcciones} = 65536 \text{ direcciones}$$

b) 16 líneas BA } 16 líneas BD }  $\xrightarrow{\text{puede acceder a}}$   $2^{16} = 2^6 \cdot 2^{10} = 64 K \text{ direcciones}$   
= 65536 direc.

Ancho palabra = 16 bits . = 2 bytes

c) 10 líneas BA } 8 líneas BD }  $\xrightarrow{\text{Ocupa}} 2^{10} \times 8 = 8192$

Cantidad de direcciones  
" 1024 "

7 anillos de direcciones

Ocupa esa # Dir.

d) 10 l BA ?  
16 l BD ?

$$2^{10} = 1024 \text{ direcciones}$$

$\frac{16}{8} = 2 \text{ bytes} = \text{Ancho de palabra}$

e)  $1K \times 8 \rightarrow 2^{10} =$  ~~lentas direcciones~~  
 $8 = 1 \text{ byte} = \text{Ancho paralelo}$

a partir de  $0x03FF = 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1$

¿Cómo se activa su CS?

Si sumamos uno.

$$0x03FF + \underbrace{1024}_{\leftarrow \text{Hay que sumar 1 menor que lo que ocupa}} = 2046 \quad \text{y} \quad 0x07FF - 1$$

Entonces nos dará:

$$0x03FF - 0x07FF$$

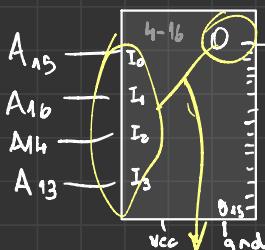
i:	$A_{15} A_{14} A_{13} A_{12}$	$A_9 A_8 A_7 A_6$	$A_5 A_4 A_3 A_2$	$A_1 A_0$
	0 0 0 0	0 0 1 1	1 1 1 1	1 1

f:	0 0 0 0	0 1 1 1	1 1 1 1	1 1 1 1
	0 0 0 0	0 1 1 1	1 1 1 1	1 1 1 1

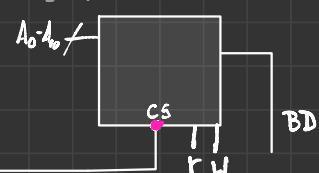
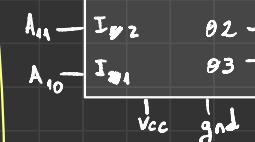
Esto nos da el último bit  
apagado.

Supuestamente este binario si es una coma en dirección a  $0x0400$

En tal caso es fácil:



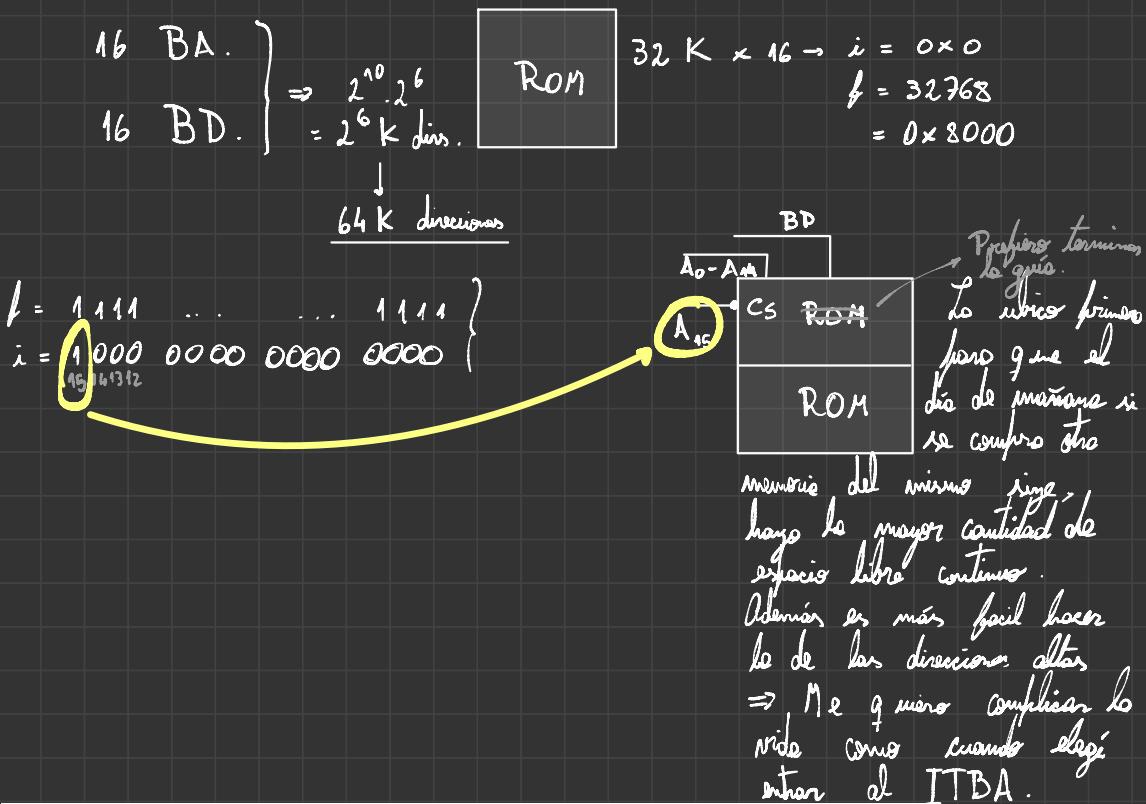
$0\ 0\ 0\ 0$



Modo 0 si están en 0

## Ejercicio 2 – 1 ROM

Diseñar un circuito con un procesador de 16 líneas de bus de address y 16 líneas de bus de datos. Se cuenta con **una** memoria ROM de 32K x 16. Conecte el procesador y el integrado de ROM y explique las decisiones tomadas en cuanto a la ubicación en el mapa de memoria y que sucede con las posiciones de memoria no ocupadas por la ROM.



## Ejercicio 3 – 2 ROMs

Diseñar un circuito con un procesador de 16 líneas de bus de address y 16 líneas de bus de datos. Se cuenta con **dos** integrados de memoria ROM de 16K x 16. Se desea correr en este sistema un programa de 32 KBytes de tamaño que comience en la dirección de memoria 0000h. Tenga presente las compuertas para resolver este ejercicio.

16 BA  $\rightarrow A_0 - A_{15}$  2 ROMs de  $16 \text{ K} \times 16$

16 BD

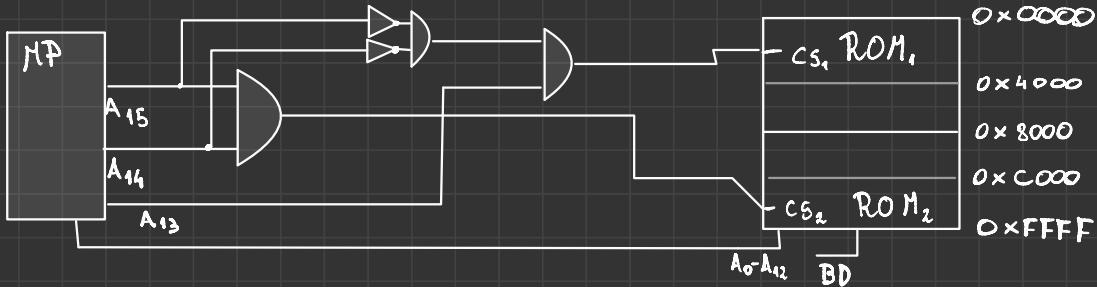
$0 \times 0000 - 0 \times 4000 \rightarrow \text{ROM}_1 \rightarrow \begin{matrix} 0000 & 0000 & 0000 & 0000 \\ 0001 & 1111 & 1111 & 1111 \end{matrix}$

$0 \times C000 - 0 \times FFFF \rightarrow \text{ROM}_2 \rightarrow \begin{matrix} 1100 & 0000 & 0000 & 0000 \\ 1111 & 1111 & 1111 & 1111 \end{matrix}$

ROM <sub>1</sub>	0x0000
ROM <sub>2</sub>	0x4000
	0x8000
	0xC000
	0xFFFF

Entonc  $\lceil \frac{32 \text{ K bytes}}{1 \text{ ROM}} \rceil \text{ progr.} \{ 16 \text{ BA}, 16 \text{ BD} \rightarrow 2^{16} \times 2 = 131072 \text{ bytes}$

Menor

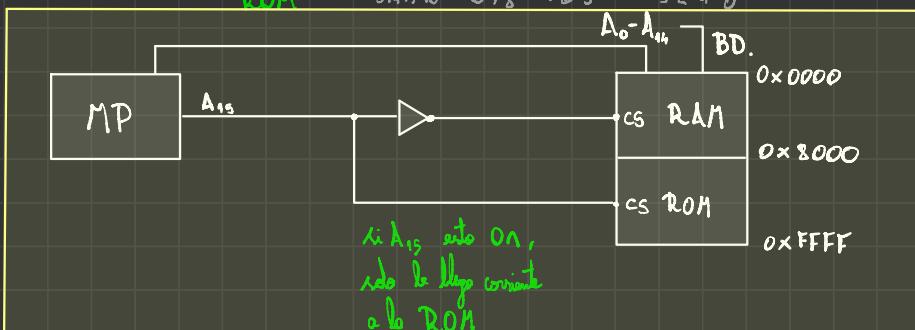
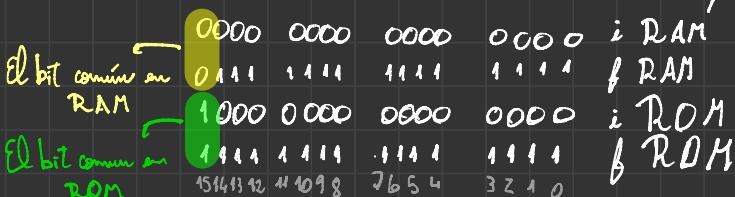


### Ejercicio 4 – RAM + ROM

Diseñar un circuito de una computadora, con un procesador de 16 líneas de bus de address y 16 líneas de bus de datos. Se cuenta con dos tipos de integrados para las memorias RAM y ROM. Los chips son de 32K x 16 para ambas. Armar el sistema de tal manera que el sistema cuente con 32K x 16 de RAM y 32K x 16 de ROM.

$$\begin{aligned}
 & 16 \text{ BA} \longrightarrow A_0 - A_{15} \\
 & 16 \text{ BD} \\
 & - \text{ De RAM: } 32K = 32 \cdot 2^{10} \times 16 \\
 & - \text{ De ROM: } 32K = 32 \cdot 2^{10} \times 16
 \end{aligned}
 \quad \left. \begin{array}{l} = 2^6 \cdot K \times 16 \\ = 64 \cdot K \times 16 \end{array} \right\} \rightarrow \begin{array}{c} \text{RAM} \\ \text{ROM} \end{array} \quad \begin{array}{l} 0x0000 \\ 0x8000 \\ 0xFFFF \end{array}$$

La mitad para RAM y la otra mitad para ROM.



→ No está mal el razonamiento pero la idea es usar decod.

¿No pasa nada si no llenas todas las entradas de un decod?

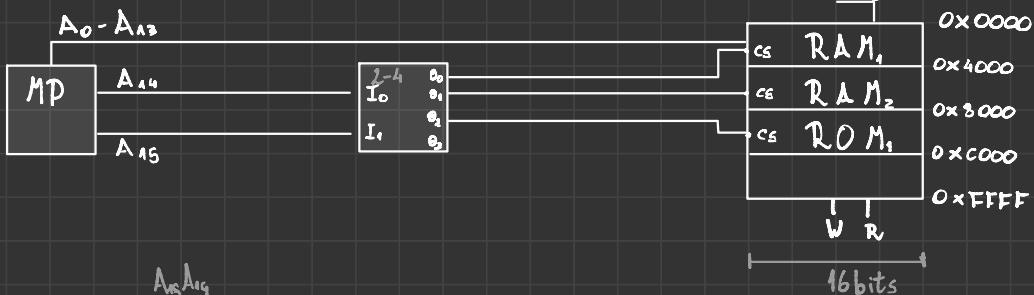
Entender que se puede no llenar todas las entradas.

Si se fijan las entradas de decod. para tener que se conecte a tierra porque si no hay "nodo" en la entrada.

## Ejercicio 5 – RAM + ROM (V2)

Igual que el ejercicio anterior, pero ahora las memorias son todas de 16K x 16, y se quieren mapear 32K x 16 de RAM y 16K x 16 de ROM.

El mapa de memoria me cambia pero tenemos más secciones



RAM<sub>1</sub>: i: 0000 0000 0000 0000  
f: 0011 1111 1111 1111

RAM<sub>2</sub>: i: 0100 0000 0000 0000  
f: 0111 1111 1111 1111

ROM<sub>1</sub>: i: 1000 0000 0000 0000  
f: 1011

ROM<sub>2</sub>: i: 1100 0000 0000 0000  
f: 1111 1111 1111 1111

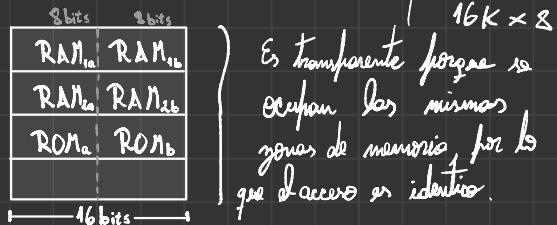
## Ejercicio 6 – RAM + ROM (V3)

Igual que el ejercicio anterior, pero ahora las memorias son de 16K x 8. ¿Cómo conectaría las memorias para que para el procesador sea transparente? ???

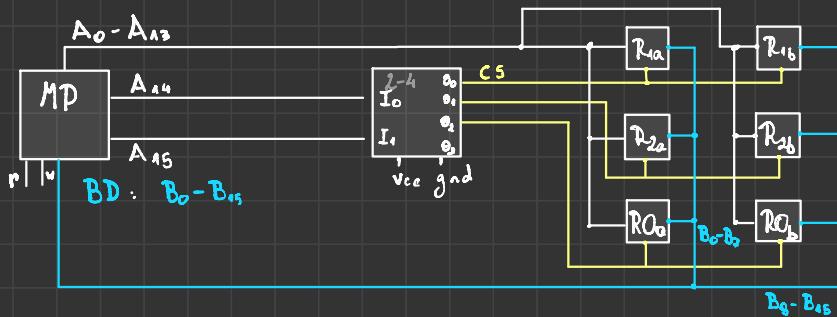
Entiendo que se quiere mantener la misma implementación que en el ej. Anterior, pues la transparencia implica independencia respecto del procesador.

Entonces se podrían formar 2 RAM 16K x 16 con 2 RM<sub>s</sub>

De la siguiente forma:



Es transparente porque se ocupan las mismas zonas de memoria, por lo que el acceso es idéntico.



### Ejercicio 4 – RAM + ROM

Diseñar un circuito de una computadora, con un procesador de 16 líneas de bus de address y 16 líneas de bus de datos. Se cuenta con dos tipos de integrados para las memorias RAM y ROM. Los chips son de 32K x 16 para ambas. Armar el sistema de tal manera que el sistema cuente con 32K x 16 de RAM y 32K x 16 de ROM.

### Ejercicio 7

Al circuito del Ejercicio 4 se le desea adicionar un periférico que esté mapeado en la memoria. Posee 16 registros de 16 bits cada uno. ¿Qué es lo único que interpreta el procesador?

Solo interpreta que tiene más memoria en lo que leer y escribir en el mapa de I/O.

16 l BA

16 l BD

RAM 32K x 16 : 0x0000 - 0x7FFF

ROM 32K x 16 : 0x8000 - 0xFFFF

Perif. 16 registros x 16 bits  
 $\Rightarrow$  16 direcciones con 16 de ancho

RAM: i: 0000 0000 0000 0000

f: 0111 1111 1111 1111

$0x0000 - 0x000F$   
 (con ancho de 16b)

ROM: i: 1000 0000 0000 0000

f: 1111 1111 1111 1111

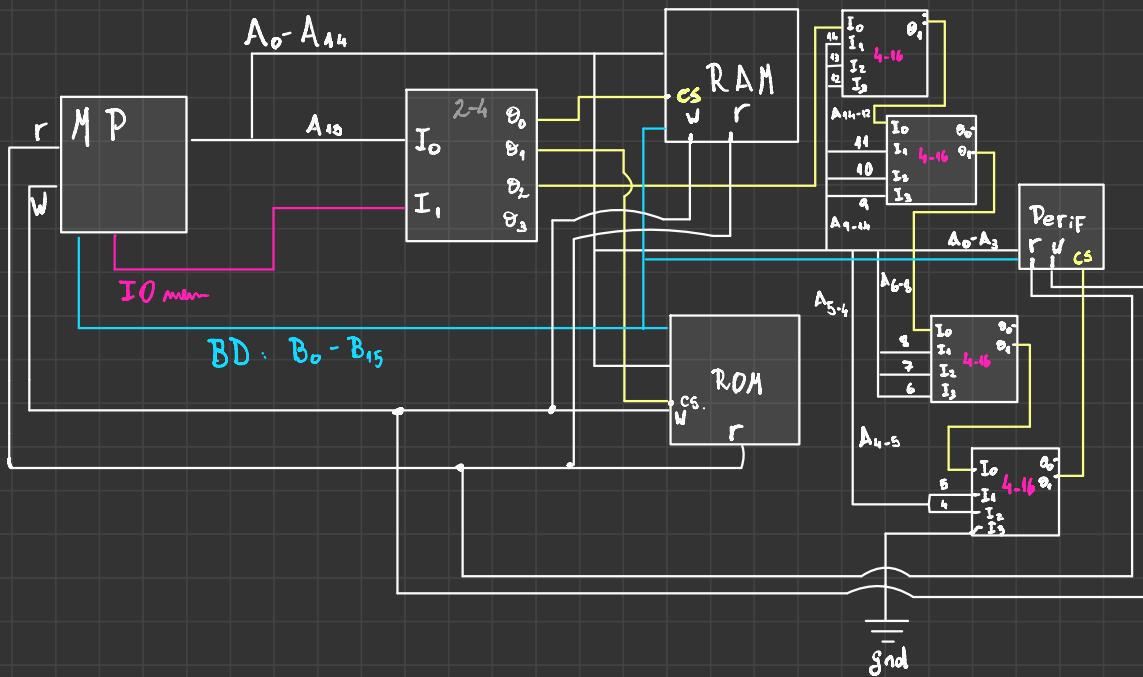
Perif. 16x16: i: 0000 0000 0000 0000

f: 0000 0000 0000 1111

↑ 14 13 12 11 10 9 8 7 6 5 4 3 2 ↑  
 ↓ 15

En el mapa de I/O

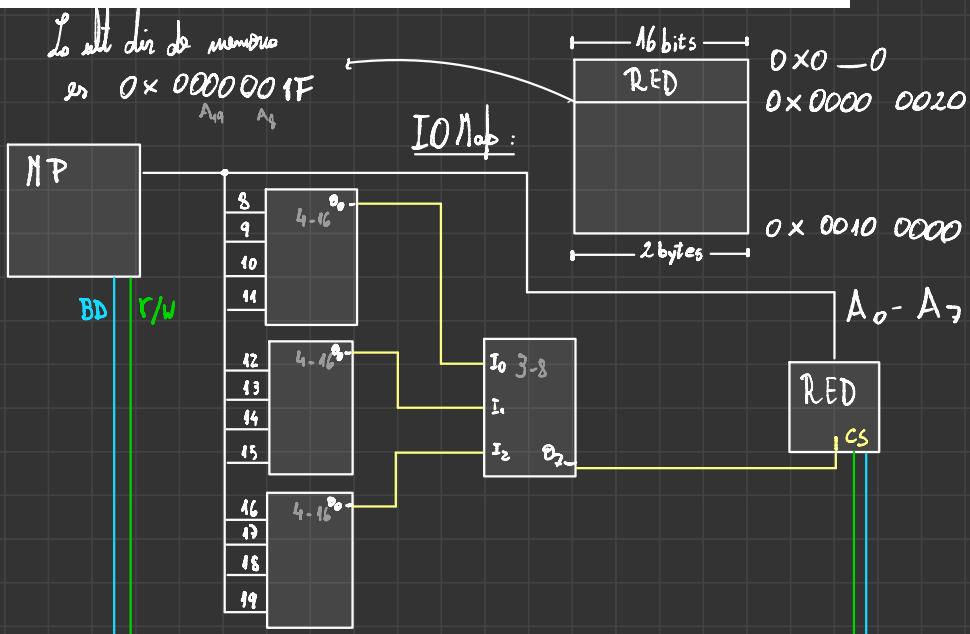
I/O mapeo.	0x0000
Perif.	0x000F
	0xFFFF



## **Ejercicio 9 – Placa red (V2)**

Explique de qué manera se debe modificar el sistema del ejercicio 8, si en lugar de tener un registro de envío y otro para recepción, la placa tuviese un buffer de 64 bytes mapeado en memoria

Realice la decodificación completa de este nuevo sistema y muestre el mapa de memoria.



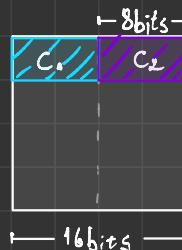
## Ejercicio 10 – Config. memorias

Únicamente se cuenta con Chips de 8K x 8, ¿Cómo se logran las siguientes configuraciones de memoria?

- 16K x 8
- 8K x 16
- 64K x 16
- 128K x 32
- 4K x 8
- 4K x 16

} → Entiendo que uno no puede

- 8K x 16 : 8K de direcciones, los 16 bits de ancho, es fácil ver que podríamos juntar dos chips y que el primero represente la parte baja mientras que el segundo la parte alta:



- 16K x 8 :

$$\begin{array}{|c|c|} \hline C_1 & + 8K \text{ direc.} \times 8 \\ \hline C_2 & + 8K \text{ direc.} \times 8 \\ \hline \end{array} \quad \left. \begin{array}{l} \\ \end{array} \right\} = 16K \text{ direc.} \times 8 \text{ bits}$$

- 64K x 16 :

$$\sum_{1}^{16} 8K \times 16 = 64K \times 16$$

$$\left. \begin{array}{|c|c|} \hline C_1 & C_2 \\ \hline C_3 & C_4 \\ \hline C_5 & C_6 \\ \hline C_7 & C_8 \\ \hline C_9 & C_{10} \\ \hline C_{11} & C_{12} \\ \hline C_{13} & C_{14} \\ \hline C_{15} & C_{16} \\ \hline \end{array} \right\} + 8K \times 8 \times 2 = 8K \times 16$$

$+ 8K \times 16$   
 $+ 8K \times 16$   
 $+ 8K \times 16$

•  $128 \text{ K} \times 16$  con chips de  $8 \text{ K} \times 8$

$$128/8 = \text{Cant rangos} = 16 \}$$

$$16/8 = \text{Cant columnas} = 2 \}$$

32 chips  
juntos en el mismo  
formato que  $64 \text{ K} \times 16$

### Ejercicio 8 – Placa de red

Se dispone de un microprocesador con 20 líneas de bus de address y 16 líneas de bus de datos.

Se desea utilizarlo para enviar y recibir datos a través de una red. Para esto se cuenta con una placa de RED de red con las siguientes características:

- 1 registro ST0, cuyo valor es 1 cuando hay datos esperando ser leídos.
- 1 registro ST1, del cual se pueden obtener los datos recibidos.
- 1 registro CR0, que se setea en 1 para enviar datos.
- 1 registro CR1, en el cual se colocan los datos a enviar.

Realice la decodificación completa de éste sistema. Mapee la placa de Red en el Mapa de Entrada y Salida.

### IO Map:

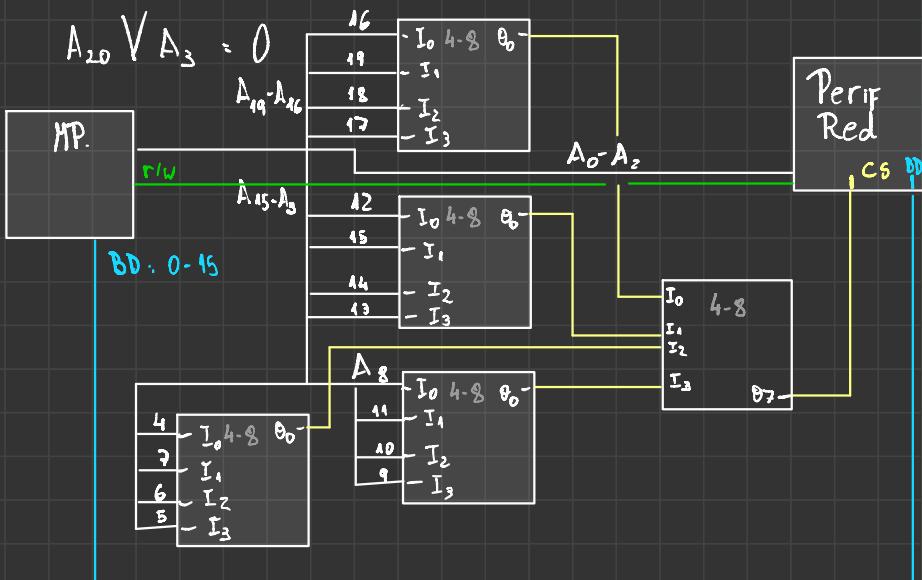
ST0	0x 0000 0000
ST1	0x 0000 0001
CR0	0x 0000 0002
CR1	0x 0000 0003
	0x 0000 0004

↓ 16 bits ↓

0x 0010 0000

$$\text{MP} \rightarrow \left\{ \begin{array}{l} 20 \text{ l BA.} \\ 16 \text{ l BD.} \end{array} \right\} \rightarrow 2^{20} = 2^{10} \times 2^{10} = 1 \text{ K}^2$$

16 bits = 2 bytes = Ancho paralelo.



## Ejercicio 12 – Sistema encriptar

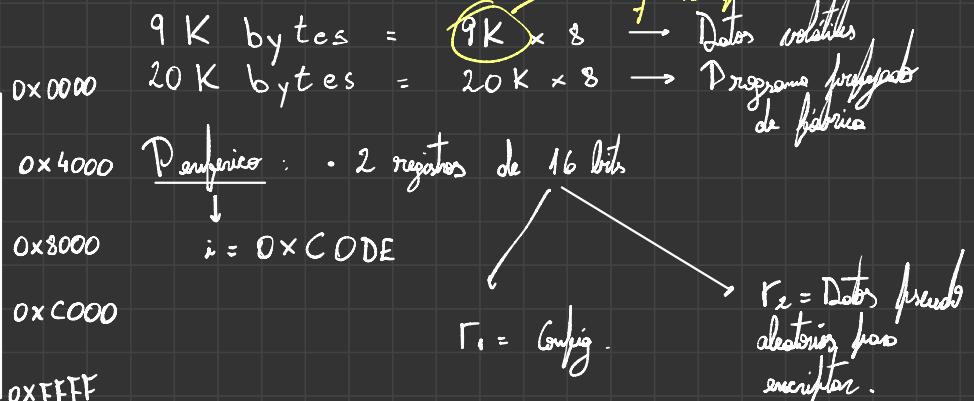
Se está desarrollando un pequeño sistema para encriptar mensajes. El sistema debe soportar 9 Kbytes para datos volátiles y 20 Kbytes para el programa prefijado de fábrica. Además se desea mapear como periférico un integrado que tiene 2 registros de 16 bits en la dirección 0xC0DE. En el primer registro se encuentra la configuración y en la segunda se encuentran los datos pseudoaleatorios para encriptar.

Se cuenta con un **microprocesador genérico** de 16 bits de bus de datos y 16 de bus de address, RAM de 16K x 16 y ROM 8K x 8

- Dibuja el Mapa de Memoria y de Entrada/Salida
- Haga esquema de todo el circuito con todas las memorias y periféricos
- Haga esquema del circuito Decodificador

¿Qué sería esto?

Mem Map:



### Ejercicio 13 – Proyecto

Se dispone de un microprocesador del tipo Intel (tiene salida IO/M) de 8 bits de bus de datos de 32 bits de bus de direcciones. El dueño del proyecto nos brinda 2 (dos) módulos de memoria RAM, cada uno de ellos de 1G x 8 y nos pide que los ubicemos a partir de la dirección 0h en forma contigua para lograr 2 Gbytes de RAM.

Además tenemos que decodificar un modulo de ROM de 256M x 8 para el código que corre al encender el sistema, pero el único dato que tenemos es que el registro EIP al iniciarse el procesador toma el valor 0xFFFFFFFF. **El dueño del proyecto nos avisa que su código ocupa 200 Mbytes.** → ~~es imposible qg al EIP inicie a 16s del final~~

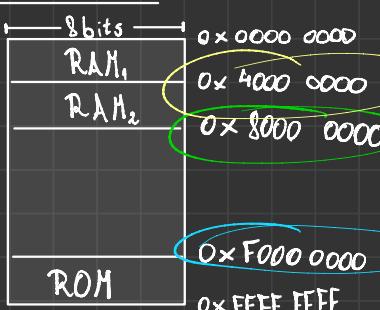
Por una cuestión de espacio y economía usted solo puede usar hasta 3 decodificadores como máximo para resolver la decodificación completa de RAM y ROM. Pueden ser decodificadores de 2 a 4, 3 a 8 y 4 a 16. No dispone de compuertas para la decodificación.

Se pide en este orden que:

- Dibuje el mapa de memoria.
- Dibuje un diagrama esquemático de los integrados utilizados para la RAM y ROM (líneas de datos, direcciones, etc.) Debe colocar nombres a todas las líneas de circuito utilizadas.
- Resuelva la decodificación con estas condiciones y dibuje el circuito obtenido.
- ¿Qué pedido le tiene que hacer usted al programador del código que irá en la ROM para que todo esto funcione correctamente al iniciarse el sistema.

32 bits BA  $\Rightarrow$  4 bytes

1) Mapa de memoria:  $2^{32}$  direcciones



$$M = 2^{20}, G_1 = 2^{30}, T = 2^{40}, P = 2^{50}, E = 2^{60}$$

• 32 bits BA

• 8 bits BD

• 2 RAMs  $\rightarrow 1G \times 8$

↳ A partir de 0h de forma continua

• ROM  $\rightarrow 256M \times 8$

↳ Para el código de inicio

• EIP  $i = 0 \times FFFFFFFF0$

• El código de inicio ocupa 200MB

{ El código era al final bytes

Esto es qg es un EIP  
 $\Rightarrow$  Ocupa 16 bits y justamente de 0x FFFF FFFF al final, hay 16 direcciones hasta el final

$2^{32} - 2^{28} = \# \text{Dir} - (\text{size ROM})$   
= Donde debería estar la ROM

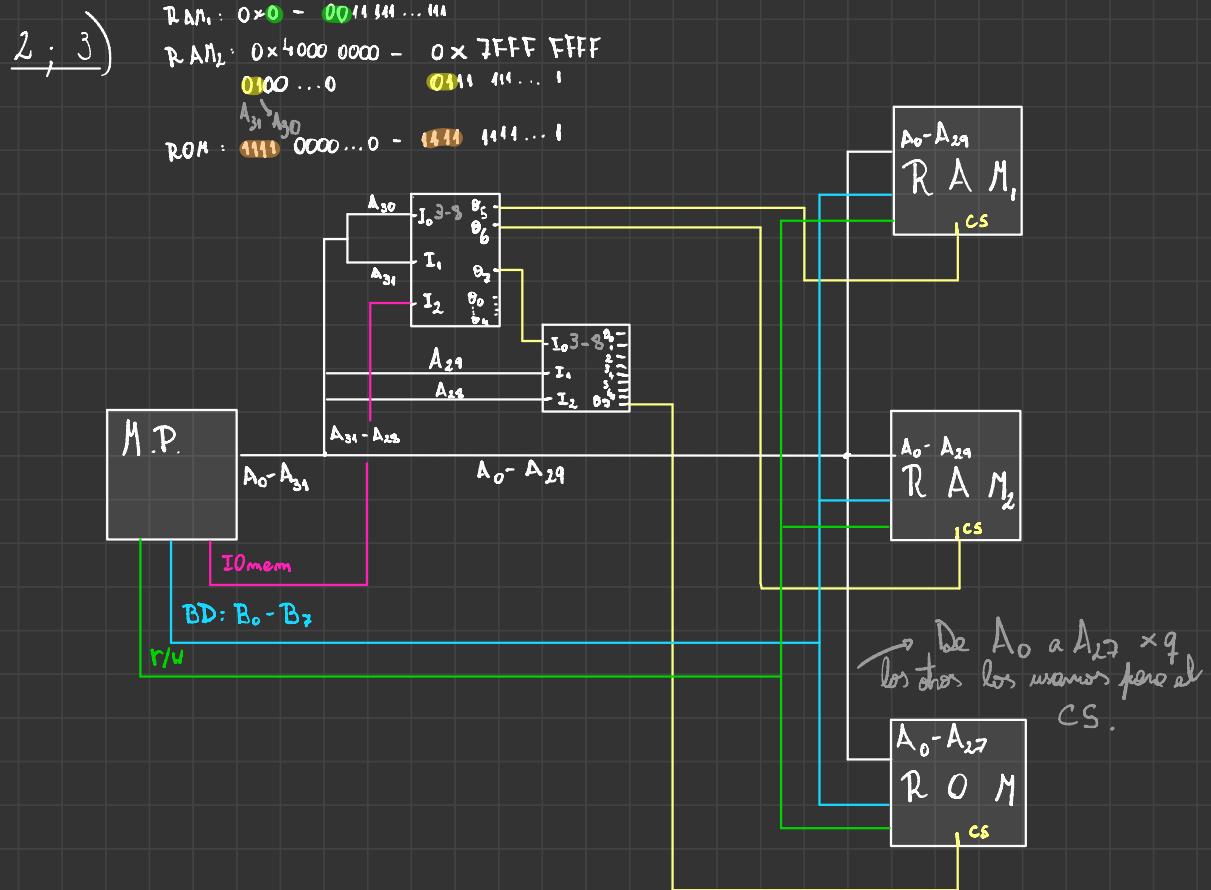
$$1G_i = 2^{30} \Rightarrow 2G = 2 \cdot 2^{30} \Rightarrow 0 \times 03FF FFFF$$

$$= 0000\ 0111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111 \rightarrow \text{Donde } 0 \times 0 \text{ hasta}$$

$$256M \times 8 \Rightarrow 256 \times 2^{20} = 2^8 \times 2^{20} = 2^{28}$$

$$2^{32} - 2^{28} = \boxed{\text{Direcciones}}$$

Donde empieza la ROM pero que entra al final.



4) Que el EIP inicia en  $0 \times FFFF FFF0$  por lo que la primera instrucción de start tiene que estar ahí, pero si realmente el código en instrucciones ocupa más de 16 bytes, lo cual es muy probable, no a tener que escribir en direcciones menores y hacer un  $Jmp$  porque sino no va a entrar el código.