PROBLEMAS: La memoria

1. Indique la capacidad en celdas o bits de las siguientes organizaciones de memoria: 1Kbit, 512x2bits, 2Kx4bits, 4KByte, 16KByte, 128x16bits. ¿Diga, para cada caso, el tamaño del bus de direcciones y del bus de datos?

SOLUCIÓN:

La capacidad viene dada por el producto del número de posiciones de memoria de cada pastilla multiplicado por el ancho de palabra. Esto es:

pastilla	(direcciones)×palabra	dir	datos	capacidad
1Kbit	$(1 \times 2^{10}) \times 1$ bits	10	1	2^{10} bits = 1 024 bits
512x2bits	$(512) \times 2$ bits	9	2	2^{10} bits = 1 024 bits
2Kx4bits	$(2 \times 2^{10}) \times 4$ bits	11	4	2^{13} bits = 8 192 bits
4KB	$(4 \times 2^{10}) \times 8$ bits	12	8	2^{15} bits = 32 768 bits
16KB	$(16 \times 2^{10}) \times 8$ bits	14	8	2^{17} bits = 131 072 bits
128x16bits	$(128) \times 16$ bits	7	16	2^{11} bits = 2 048 bits

2. Disponemos de 4 pastillas de memoria SRAM con las siguientes organizaciones: 8KB, 256x16bits, 2Kbit y 4Kx4bits. Indique cuál de ellas dispone de mayor capacidad. Sabiendo que todas ellas tienen el mismo tiempo de ciclo (t_c) indique cuál de ellas tiene la mayor velocidad de transferencia. ¿Qué asociación de memorias ofrecería tanta capacidad como la mayor pastilla con la mayor velocidad de transferencia utilizando el mínimo número de integrados?

SOLUCIÓN:

La capacidad viene dada por el producto del número de posiciones multiplicado por el ancho de palabra. Esto es:

pastilla	(direcciones)×palabra	capacidad
8KB	$(8 \times 2^{10}) \times 8$ bits	2^{16} bits = 65 536 bits
256x16bits	$(256) \times 16$ bits	2^{12} bits = 4 096 bits
2Kbits	$(2 \times 2^{10}) \times 1$ bits	2^{11} bits = 2 048 bits
4Kx4bits	$(4 \times 2^{10}) \times 4$ bits	2^{14} bits = 16 384 bits

La organización de mayor capacidad es la 8KB.

La definición de velocidad de transferencia (V) es la siguiente:

$$V = ancho\_palabra/t_c$$

En consecuencia, dado que todas las pastillas tienen el mismo tiempo de ciclo t_c, la mayor velocidad de transferencia será la correspondiente a la organización de palabra más ancha, es decir, la de la pastilla 256x16bits.

$$V = 16/t_c \text{ bits/s}$$

Para igualar la máxima capacidad (2^{16} bits de la pastilla 8KB) necesitamos la siguiente cantidad de pastillas en cada caso:

pastilla	asociación
8KB	$2^{16}/2^{16} = 1$ pastilla
256x16bits	$2^{16}/2^{12} = 16$ pastillas
2Kbits	$2^{16}/2^{11} = 32$ pastillas
4Kx4bits	$2^{16}/2^{14} = 4$ pastillas

Para obtener la máxima velocidad de transferencia, hemos de utilizar una palabra de 16 bits de ancho y para usar el mínimo número de integrados o pastillas, asociamos 4 pastillas de 4Kx4bits en paralelo.

3. Disponemos de pastillas de memoria cuya organización es 4Kx2bits. Se desea implementar con ellas un mapa de memoria de 16Kx2bits. Diseñe el sistema de memoria sabiendo que el acceso a cada integrado se realiza mediante una señal específica `ME` (*Memory Enable*).

SOLUCIÓN:

Dado que el ancho del bus de datos es idéntico en el mapa de memoria y en las pastillas, solamente hemos de preocuparnos por la extensión del mapa de memoria. El número de pastillas que necesitamos es igual a:

$$direcciones\_mapa\_memoria/direcciones\_pastilla = 16\text{K}/4\text{K} = 4 \text{ pastillas}$$

El número de líneas del bus de direcciones de cada pastilla es $\log_2 4\text{K} = \log_2 2^{12} = 12$ y el del mapa de memoria $\log_2 16\text{K} = \log_2 2^{14} = 14$ de donde las 12 de menor peso atacan en común a todos los integrados y las 2 de mayor peso atacan a un decodificador 2:4 que sirve para habilitar cada integrado individualmente gracias a la señal específica (ME de *Memory Enable*, CE de *Chip Enable* o CS de *Chip Select*).

4. Disponemos de pastillas de memoria cuya organización es 4Kx2bits. Se desea implementar con ellas un mapa de memoria de 4Kx8bits. Diseñe el sistema de memoria sabiendo que el acceso a cada integrado se realiza mediante una señal específica ME (*Memory Enable*).

SOLUCIÓN:

Ya que el mapa de memoria pedido y el de cada pastilla coinciden, solamente hemos de hacer una extensión del bus de datos disponiendo las pastillas de memoria necesarias en paralelo de forma que el bus de direcciones seleccione la misma posición en todas y cada una de ellas y todas vuelquen su dato simultáneamente al bus de datos. Necesitaremos las siguientes pastillas:

$$datos\_mapa\_memoria/datos\_pastilla = 8/2 = 4 \text{ pastillas}$$

Las 12 líneas del bus de direcciones atacan a todos los integrados a la vez y también la señal específica (ME de *Memory Enable*, CE de *Chip Enable* o CS de *Chip Select*) se activa a la vez. Luego, las salidas de datos se disponen en paralelo sobre el bus de datos.

5. Disponemos de pastillas de memoria cuya organización es 4Kx2bits. Se desea implementar con ellas un mapa de memoria de 16Kx8bits. Diseñe el sistema de memoria sabiendo que el acceso a cada integrado se realiza mediante una señal específica ME (*Memory Enable*).

SOLUCIÓN:

En este caso hemos de hacer tanto una extensión de direcciones como de datos. Necesitaremos en total 16 pastillas:

$$\frac{direcciones\_mapa\_memoria \times datos\_mapa\_memoria}{direcciones\_pastilla \times datos\_pastilla} = \frac{16\text{K} \times 8}{4\text{K} \times 2} = \frac{2^{17}}{2^{13}} = 16 \text{ pastillas}$$

De las 14 líneas de direcciones, 12 atacan a los 16 integrados simultáneamente. Las otras dos atacan un decodificador 2:4 que selecciona 4 pastillas a activar en parelalo a través de la señal específica (ME de *Memory Enable*, CE de *Chip Enable* o CS de *Chip Select*).

6. Se desea diseñar un mapa de memoria con 512K de RAM y 128K de ROM al que se acceda por bytes. Disponemos para la tarea de pastillas RAM estáticas de 64Kx8bits y ROM de 32Kx8bits. Determine la distribución de pastillas indicando sus rangos de direcciones si la memoria RAM se situa en posiciones altas y las ROM en bajas.

SOLUCIÓN:

La organización de las pastillas en palabras de tamaño byte coincide con la del mapa de memoria por lo que no hay más que distribuir las pastillas según su tipo empezando por la ROM en las posiciones bajas.

¿Cuántas pastillas de ROM se necesitan? Será el tamaño total de ROM entre el tamaño de la pastilla ROM, es decir, $2^{17}/2^{15} = 2^2 = 4$ pastillas.

¿Cuántas pastillas de RAM se necesitan? Será el tamaño total de RaM entre el tamaño de la pastilla RAM, es decir, $2^{19}/2^{16} = 2^3 = 8$ pastillas.

El mapa de memoria total tiene un tamaño de $512K + 128K = 640K$ y le corresponde un bus de direcciones de $\log_2 640K = 20$ bits. Las pastillas de ROM se direccionan con 15 bits y las de RAM con 16 bits. El mapa de memoria será:

A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	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							ROM 0								
0	0	0	0	1							ROM 1								
0	0	0	1	0							ROM 2								
0	0	0	1	1							ROM 3								
0	0	1	0	0							libre								
0	1	1	1								libre								
1	0	0	0								RAM 0								
1	0	0	1								RAM 1								
1	0	1	0								RAM 2								
1	0	1	1								RAM 3								
1	1	0	0								RAM 4								
1	1	0	1								RAM 5								
1	1	1	0								RAM 6								
1	1	1	1								RAM 7								

Vemos que el rango libre es $2^{19} - 2^{17} = 384K$.

7. Sea una memoria con una capacidad total de 24Mbits organizada en bytes y distribuida en un número par de pastillas. ¿Cuántos bits de direccionamiento son necesarios para referenciarla? Supongamos que los accesos se realizan alineados a palabra de 16 bits, es decir, cada referencia a memoria accede al byte par y al impar consecutivo. ¿Qué mecanismo *hardware* asegura el alineamiento? ¿Cuántos bits de direccionamiento son necesarios ahora? ¿Influye el acceso alineado en la velocidad de transferencia?

SOLUCIÓN:

El tamaño de la memoria es $24Mbits = 3 \times 2^3 \times 2^{20} = 3 \times 2^{23} bits$. Como está organizada en palabras de tamaño byte, el número de palabras es $3 \times 2^{23}/2^3 = 3 \times 2^{20} palabras$ y, por tanto, necesitamos $\lceil \log_2 3 \times 2^{20} \rceil = 24$ líneas en el bus de direcciones.

Para alinear los accesos a palabras de 16 bits basta con emitir direcciones pares, es decir, acceder a memoria de 2 en 2 bytes. De esta manera basta con emitir direcciones de 19 lineas puesto que el LSB siempre es 0.

Las pastillas se disponen como si de una extensión del bus de datos se tratara y cada vez que emitimos una dirección ataca a dos pastillas, una suministra el byte de la dirección par y la otra el

de la impar. El mecanismo multiplica por 2 la velocidad de transferencia pues suministra acceso a 2 bytes por cada direccionamiento.

El problema no establece los tamaños de las pastillas pero sí nos asegura de que son pares con lo que la distibusión del mapa de memoria es compatible con el mecanismo de alineamiento.

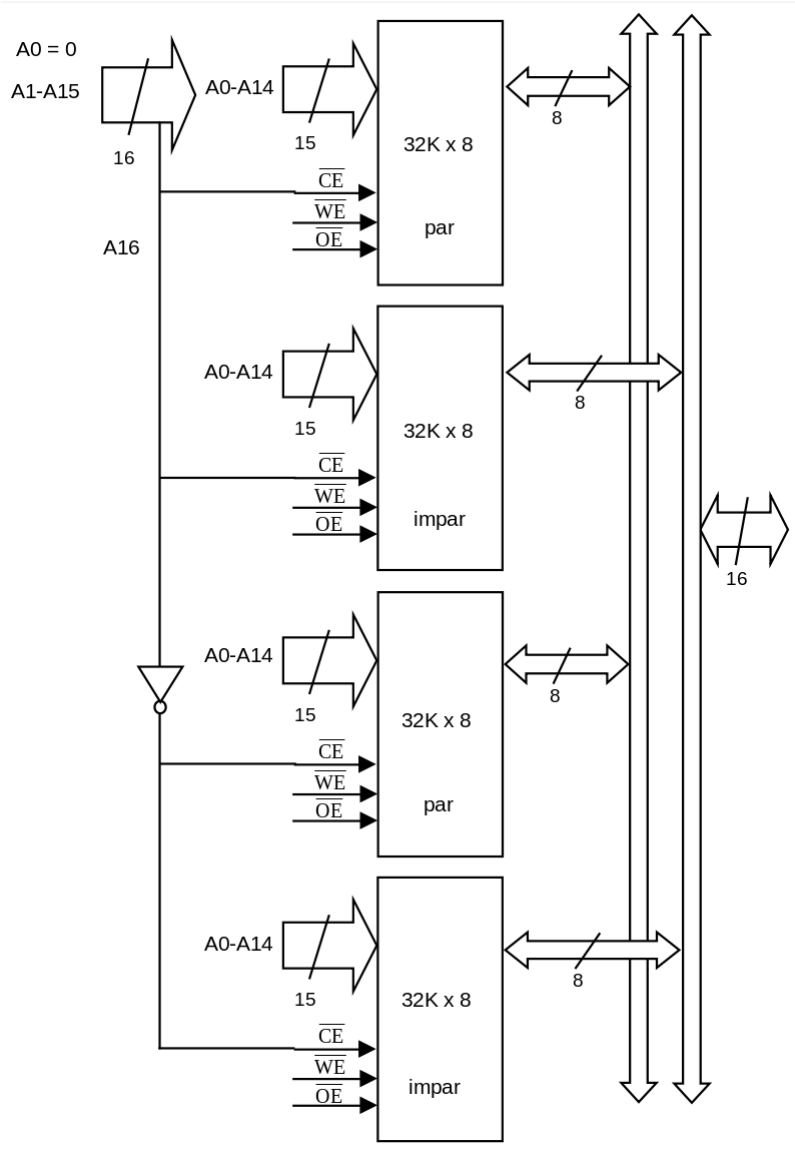
8. Diseñe un sistema de memoria de 128Kbytes empleando pastillas de 32Kbytes que disponen de las señales de control `CE` (*Chip Enable*), `WE` (*Write Enable*) y `OE` (*Output Enable*), con un bus de direcciones de 16 líneas. La memoria está organizada en bytes pero existe un alineamiento *hardware* a palabra de 16 bits.

SOLUCIÓN:

Las pastillas de memoria de 32Kbytes tienen 15 líneas de direcciones. Dado que existe un alineamiento *hardware* a palabra de 16 bits, en cada acceso a memoria se deben leer o escribir 16 bits, el byte par y el byte impar. Asociamos las pastillas de dos en dos para transferir los 2 bytes.

El mapa de memoria tiene 128Kbytes lo que requiere $\log_2 128K$ líneas de direcciones, es decir, 17 líneas. Puesto que el alineamiento es a palabra de 2 bytes, el LSB es siempre 0 ($A_0 = 0$) y, por tanto, solamente se requieren 16 bits (los de mayor peso $A_1 - A_{15}$).

El esquema hardware sería el siguiente:



9. Calcule el número de accesos que pueden realizarse a un dispositivo DRAM durante un intervalo de tiempo de 1 segundo sabiendo que el tiempo de acceso es $t_a = 80$ns y el tiempo de restauración de la información $t_{res} = 20$ns, y que acceso y restauración se alternan.

SOLUCIÓN:

El tiempo de ciclo (t_c) es la suma del tiempo de acceso (t_a) más el tiempo de restauración (t_{res}):

$$t_c = t_a + t_{res} = 80 + 20 = 100\text{ns}$$

El número de accesos por segundo será:

$$\frac{\Delta t}{t_c} = \frac{1}{100 \cdot 10^9} = 10^7 \text{accesos}$$

Es decir, la DRAM propuesta puede servir 10 millones de accesos por segundo.

10. La velocidad de transferencia V de una memoria DRAM es de 2,4Mbits/s. Supongamos que logramos disminuir el tiempo empleado en el refresco o restauración de la misma de forma que el tiempo de ciclo t_c disminuya un 5%. Bajo estas condiciones determine la nueva velocidad de transferencia.

SOLUCIÓN:

El nuevo tiempo de ciclo será:

$$t_{c_{\text{NUEVO}}} = t_c \cdot 0,95$$

De donde la nueva velocidad de transferencia será:

$$V_{\text{NUEVO}} = \frac{ancho\_palabra}{t_c \cdot 0,95} = \frac{V}{0,95} = \frac{2,4}{0,95} = 2,52 \text{ Mbits/s}$$

11. La velocidad de transferencia V de una memoria DRAM es de 2,4Mbits/s. Sabemos que el mapa de memoria tiene un bus de datos de tamaño *byte* y que el tiempo de restauración t_{res} supone el 15% del tiempo de ciclo t_c. Determine el tiempo de acceso t_a.

SOLUCIÓN:

Sabemos que la velocidad de transferencia es:

$$V = ancho\_palabra/t_c$$

De donde:

$$t_c = ancho\_palabra/V = 8/(2,4 \cdot 10^6) = 3.33 \cdot 10^{-6} s$$

El tiempo de acceso será:

$$t_a = 0,85 \cdot t_c = 0,85 \cdot 3.33 \cdot 10^{-6} = 2,833 \cdot 10^{-6} s = 2,833 \mu s$$

12. Disponemos de una memoria EDO (*Extended Data Output*) cuyos bloques son de 4 direcciones. El tiempo de acceso al bloque (direccionar fila) es de 30ns y el tiempo en acceder a datos del bloque es de 40ns por dato (direccionar columna). Calcule la mejora en tiempo frente a una DRAM convencional de tiempo de acceso 60ns. Desprecie en ambos casos el tiempo de refresco. Suponga que se accede a todas las direcciones del bloque consecutivamente. Repetir si sólo se accede a dos por bloque y si únicamente se accede a una de cada bloque.

SOLUCIÓN:

El tiempo de acceso a 4 posiciones de la DRAM es igual a $60 \times 4 = 240$ ns.
El tiempo de acceso a 4 posiciones consecutivas de la EDO es igual a $30 + 40 \times 4 = 190$ ns.

$$S = \frac{t_{\text{sin}}}{t_{\text{con}}} = \frac{240}{190} = 1.26$$

El tiempo de acceso a 4 posiciones dispersas de 2 en 2 es igual a $(30 + 40 \times 2) \times 2 = 220$ ns.

$$S = \frac{t_{\text{sin}}}{t_{\text{con}}} = \frac{240}{220} = 1.09$$

El tiempo de acceso a 4 posiciones dispersas de 1 en 1 es igual a $(30 + 40 \times 1) \times 4 = 280$ ns.

$$S = \frac{t_{\text{sin}}}{t_{\text{con}}} = \frac{240}{280} = 0.85$$

13. Disponemos de una memoria EDO (*Extended Data Output*) organizada en bloques de 4 direcciones. El tiempo de acceso al bloque (direccionar fila) es de 40ns y el tiempo de acceso a cada dato del bloque (direccionar columna) es de 25ns. Calcule el *speed-up* frente a una DRAM convencional de tiempo de acceso 60ns despreciando en ambos casos el tiempo de refresco. Suponga que la huella de memoria es tal que en el 50% de los accesos referenciamos todas las direcciones del bloque consecutivamente y en el 50% restante solamente una dirección.

SOLUCIÓN:

El tiempo de acceso a 4 posiciones de la DRAM es igual a $60 \times 4 = 240$ ns.
El tiempo de acceso a 4 posiciones consecutivas de la EDO es igual a $40 + 25 \times 4 = 140$ ns.
El tiempo de acceso a 4 posiciones dispersas de la EDO es igual a $(40 + 25) \times 4 = 260$ ns.
La media ponderada de accesos a la EDO es igual a $140 \cdot 0.5 + 260 \cdot 0.5 = 200$ ns.
De donde el *speed-up* es:

$$S = \frac{t_{\text{sin}}}{t_{\text{con}}} = \frac{240}{200} = 1.2$$

14. El sistema de memoria de un computador cuenta con una memoria principal de 4MBytes y con una caché de 256KBytes. La memoria caché está organizada de forma totalmente asociativa y cada marco de bloque o línea contiene 4 bytes. Conteste a las siguientes preguntas:
    a) número total de bloques en memoria principal.
    b) número total de líneas o marcos de bloque en caché.
    c) campos y tamaño de los mismos en los que se divide la dirección física de memoria.
    d) tamaño (en filas x columnas) de la matriz de búsqueda de la memoria asociativa que tabla las correspondencias entre bloques y líneas o marcos de bloque.
    e) tamaño de la matriz de datos de la memoria asociativa.

    Realice una comparación del tamaño total dedicado a la implementación de las tablas de correspondencias frente al área de datos de la caché.

SOLUCIÓN:

Tamaño de memoria principal: $4MB = 2^{22} bytes$
Tamaño de memoria caché: $256KB = 2^{18} bytes$

a) número de bloques: $\frac{tamano\_mem\_ppal}{tamano\_bloque} = \frac{2^{22}}{2^2} = 2^{20}$ bloques

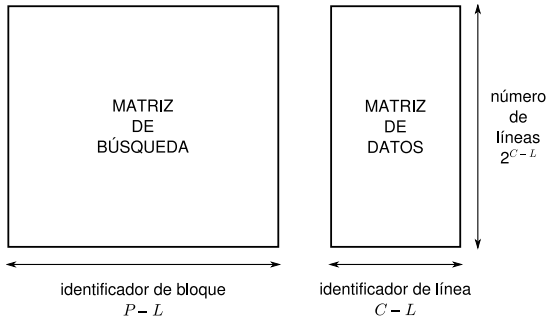
b) número de líneas: $\frac{tamano\_mem\_cache}{tamano\_bloque} = \frac{2^{18}}{2^2} = 2^{16}$ líneas

c) la dirección física se divide en dos campos cuando la organización es totalmente asociativa: campo *etiqueta* o *marca* y campo *palabra*. El campo *palabra* ocupa 2 bits ya que cada bloque tiene 4 bytes. El resto es el campo *etiqueta* cuyo tamaño coincide con la potencia del número de bloques.

21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
etiqueta																				palabra	

d) matriz de búsqueda: $numero\_lineas \times tamano\_etiqueta = 2^{16} \times 20$ celdas

e) matriz de datos: $numero\_lineas \times \log_2 numero\_lineas = 2^{16} \times 16$ celdas



La memoria asociativa que implementa la tabla de correspondencias ocupa en total:

$$2^{16} \times 20 + 2^{16} \times 16 = 2^{16} \times (20 + 16) = 2^{16} \times 36 = 2\,304Kceldas = 288KB$$

Es decir, la tabla de correspondencias (288KB) ocupa más que el área de datos de la propia caché (256KB).

15. El sistema de memoria de un computador cuenta con una memoria principal de 4MBytes y con una caché de 256KBytes. La memoria caché está organizada de forma asociativa por conjuntos de 4 vías y cada marco de bloque o línea contiene 4 bytes. Conteste a las siguientes preguntas:
    a) número total de bloques en memoria principal.
    b) número total de líneas o marcos de bloque en caché y número de conjuntos.
    c) campos y tamaño de los mismos en los que se divide la dirección física de memoria.
    d) tamaño (en filas x columnas) de la matriz de búsqueda de la memoria asociativa que tabla las correspondencias entre bloques y líneas o marcos de bloque.
    e) tamaño de la matriz de datos de la memoria asociativa.

    Realice una comparación del tamaño total dedicado a la implementación de las tablas de correspondencias frente al área de datos de la caché.

SOLUCIÓN:

Tamaño de memoria principal: $4MB = 2^{22}bytes$
Tamaño de memoria caché: $256KB = 2^{18}bytes$

a) número de bloques: $\dfrac{tamano\_mem\_ppal}{tamano\_bloque} = \dfrac{2^{22}}{2^2} = 2^{20}$ bloques

b) número de líneas: $\dfrac{tamano\_mem\_cache}{tamano\_bloque} = \dfrac{2^{18}}{2^2} = 2^{16}$ líneas

número de conjuntos: $\dfrac{numero\_lineas}{numero\_vias} = \dfrac{2^{16}}{2^2} = 2^{14}$ conjuntos

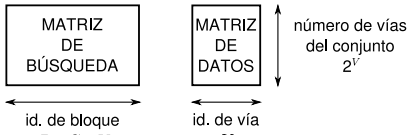
c) la dirección física se divide en tres campos cuando la organización es asociativa por conjuntos: campo *etiqueta* o *marca*, campo *conjunto* y campo *palabra*. El campo *palabra* ocupa 2 bits ya que cada bloque tiene 4 bytes. El campo *conjunto* ocupa 14 bits que coincide con la potencia

del número de conjuntos. El resto es el campo *etiqueta* cuyo tamaño coincide con la potencia de la cantidad de bloques que puede recibir cada conjunto $2^{20}/2^{14} = 2^6$.

21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
etiqueta						conjunto														palabra	

d) matriz de búsqueda de cada conjunto: $numero\_vias \times tamano\_etiqueta = 4 \times 6$ celdas

e) matriz de datos de cada conjunto: $numero\_vias \times \log_2 numero\_vias = 4 \times 2$ celdas



La memoria asociativa que implementa la tabla de correspondencias ocupa en total:

$$(4 \times 6 + 4 \times 2) \times numero\_conjuntos = 32 \times 2^{14} = 512Kceldas = 64KB$$

Es decir, la tabla de correspondencias (64KB) ocupa la cuarta parte del área de datos de la propia caché (256KB).

16. Sea un computador con una memoria caché organizada en correspondencia directa con 256 líneas de 16 bytes cada una. La política de escritura es directa con no ubicación en fallo de escritura. Determine la traza de accesos a memoria principal, el bloque de memoria principal, la línea de caché y si se produce acierto o fallo en la ejecución de la secuencia de instrucciones siguiente sabiendo que inicialmente `r1` vale `02h` y `r2` vale `008020h` y que la memoria caché está inicialmente vacía.

dirección	**instrucción**	**operación**
`0A2000h`	`st [r2],0h`	`[r2]` ← `0`
`0A2004h`	`add r2,r2,4`	`r2` ← `r2+4`
`0A2008h`	`dec r1`	`r1` ← `r1-1`
`0A200Ch`	`bnz r1,A2000h`	si `r1`≠`0` `PC = 0A2000h`
		si `r1`=`0` `PC = 0A2010h`

SOLUCIÓN:

La tabla siguiente muestra la dirección accedida, el bloque al que pertenece, el tipo de flujo, el tipo de acceso, la línea referenciada y si hay acierto o fallo. El bloque se calcula como el cociente de la división entera:

$$\texttt{bloque} = \left\lfloor \frac{\texttt{dirección}}{tamaño\_bloque} \right\rfloor$$

Y la línea como el resto de la división entera (función módulo):

$$\texttt{línea} = \texttt{bloque} \bmod número\_líneas$$

Dado que el tamaño del bloque es de 16 bytes, la división entera que nos da el número del bloque consiste en el desplazamiento a la derecha de un carácter hexadecimal y dado que el número de líneas es 256 (`100h`), la función módulo que nos da el número de línea consiste en tomar los dos dígitos hexadecimales de menor peso.

dirección	`0A2000h`	`008020h`	`0A2004h`	`0A2008h`	`0A200Ch`
bloque	`0A200h`	`00802h`	`0A200h`	`0A200h`	`0A200h`
flujo	i	d	i	i	i
acceso	l	e	l	l	l
línea	`00h`	`02h`	`00h`	`00h`	`00h`
A/F	F	F[1]	A	A	A

dirección	0A2000h	008020h	0A2004h	0A2008h	0A200Ch
bloque	0A200h	00802h	0A200h	0A200h	0A200h
flujo	i	d	i	i	i
acceso	l	e	l	l	l
línea	00h	02h	00h	00h	00h
A/F	A	F[1]	A	A	A

F[1] es un fallo en escritura sin ubicación

17. Sean dos computadores A y B cuyo procesador y memoria principal tienen las mismas características. La memoria principal tiene un tiempo de acceso de 5 ciclos de reloj por byte. La memoria caché tiene, en ambos computadores, el mismo tamaño pero distinta organización tal y como se muestra en la tabla siguiente:

	computador A	**computador B**
organización	correspondencia directa	asociativa por conjuntos
número de líneas	128	128
número de vías	—	2
tamaño de la línea	16 bytes	16 bytes
política de escritura	postescritura	directa
política de fallo en escritura	ubicación	no ubicación
política de reemplazo	—	LRU
tiempo acierto lectura	1 ciclo	1 ciclo
tiempo fallo lectura	80 ciclos	80 ciclos
tiempo acierto escritura	1 ciclo	5 ciclos
tiempo fallo escritura	85 ciclos	5 ciclos
tiempo escritura de línea	80 ciclos	—

En dichas máquinas se ejecuta un programa del que se muestra a continuación una pequeña secuencia de instrucciones:

dirección	**instrucción**
00	`ld r1,256`
04	`st 0,[r1]`
08	`add r1,4`
12	`cmp r1,272`
16	`bnz #04`

donde las direcciones y los operandos están expresados en decimal.

Considerando que las memorias caché están inicialmente vacías, se pide calcular la traza de referencias generada por la ejecución de la secuencia del programa dado. Dicha traza debe indicar el bloque de memoria principal al que pertenece cada una de las referencias, la línea y vía en su caso, y si se produce acierto o fallo en el acceso a caché. Determine el tiempo de ejecución del programa en ambos computadores, indicando la velocidad relativa de un computador frente al otro. Se tomarán únicamente los tiempos de acceso a memoria despreciando los de ejecución. Si asumimos que todos los tiempos de ejecución son de un ciclo, ¿cuál es el porcentaje de tiempo consumido en accesos a memoria frente al total?

SOLUCIÓN:

Para obtener la traza generada por la ejecución de la secuencia de programa, se deben tener en cuenta no sólo las direcciones de cada una de las instrucciones ejecutadas sino también las direcciones de los datos referenciados en ellas, si los hubiere.

Para determinar el bloque de memoria principal al que pertenece cada referencia se hace la división entera de la dirección entre el tamaño del bloque, que en este caso es de 16 bytes:

$$\texttt{bloque} = \left\lfloor \frac{\texttt{dirección}}{tamaño\_bloque} \right\rfloor = \left\lfloor \frac{\texttt{dirección}}{16} \right\rfloor$$

En cada referencia se indica si pertenece al flujo de instrucciones (i) o de datos (d) y si el acceso es en lectura (l) o escritura (e). Finalmente, el cálculo de la línea se realiza mediante la función módulo con el número de líneas, en este caso 128:

$$\texttt{línea} = \texttt{bloque} \bmod \textit{número\_líneas} = \texttt{bloque} \bmod 128$$

Y el conjunto como función módulo con el número de conjuntos, en este caso 64 (128 líneas en conjuntos de 2 en 2):

$$\texttt{línea} = \texttt{bloque} \bmod \textit{número\_líneas} = \texttt{bloque} \bmod 64$$

La traza para cada máquina será:

dirección		00	04	256	08	12	16	04	260	08	12	16
bloque		0	0	16	0	0	1	0	16	0	0	1
flujo		i	i	d	i	i	i	i	d	i	i	i
acceso		l	l	e	l	l	l	l	e	l	l	l
A	línea	0	0	16	0	0	1	0	16	0	0	1
	A/F	F	A	F	A	A	F	A	A	A	A	A
B	conj.	0	0	16	0	0	1	0	16	0	0	1
	A/F	F	A	F^1	A	A	F	A	F^1	A	A	A

dirección			04	264	08	12	16	04	268	08	12	16
bloque			0	16	0	0	1	0	16	0	0	1
flujo			i	d	i	i	i	i	d	i	i	i
acceso			l	e	l	l	l	l	e	l	l	l
A	línea		0	16	0	0	1	0	16	0	0	1
	A/F		A	A	A	A	A	A	A	A	A	A
B	conj.		0	16	0	0	1	0	16	0	0	1
	A/F		A	F^1	A	A	A	A	F^1	A	A	A

F^1 es un fallo en escritura sin ubicación

Para el computador A, los primeros accesos a los bloques provocan fallos pero luego ya son todo aciertos puesto que todos los bloques se han copiado en las líneas de caché. En flujo de datos, las escrituras también producen ubicación ya que la política de fallo en escritura es con ubicación.

Para el computador B, todos los accesos en escritura debidos al flujo de datos se cuentan por fallos ya que la política de fallo en escritura es con no ubicación y el bloque 16 no llega a cargarse nunca en el conjunto de caché. En la tabla, se ha especificado el conjunto pero no la línea ya que nunca se cargan dos sobre el mismo conjunto.

Veamos el recuento de ciclos:

computador A			computador B		
2	fallos en lectura	$2 \times 80 = 160$ ciclos	2	fallos en lectura	$2 \times 80 = 160$ ciclos
15	aciertos en lectura	$15 \times 1 = 15$ ciclos	15	aciertos en lectura	$15 \times 1 = 15$ ciclos
1	fallos en escritura	$1 \times 85 = 85$ ciclos	4	fallos en escritura	$4 \times 5 = 20$ ciclos
3	aciertos en escritura	$3 \times 1 = 3$ ciclos			
	TOTAL	263 ciclos		TOTAL	195 ciclos

El tiempo total consumido por el computador A en los accesos a memoria es de 263 ciclos de reloj mientras que el B consume 195 ciclos. El comportamiento de la caché asociativa por conjuntos es mejor siendo la de correspondencia directa un 30% más lenta aproximadamente. En la máquina A, el tiempo de escritura de línea se suma al tiempo de fallo (en lectura o escritura) si la línea a reemplazar fue escrita (está sucia). En este problema no se da nunca esa circunstancia.

Si asumimos que el tiempo de ejecución es de 1 ciclo, tenemos un total de 17 ciclos consumidos en ejecución frente a varios centenares en accesos a memoria dependiendo de la máquina. Este hecho demuestra que el impacto de la memoria sobre el rendimiento es mucho más fuerte que el propio del CPI de ejecución.

18. Un computador dispone de una memoria caché con un tiempo de acceso de 20ns por palabra y una memoria principal con un tiempo de acceso de 100ns por palabra. La política de escritura de la memoria caché es escritura directa. Mediante la ejecución de una serie de programas de prueba se ha observado que el 53% de las referencias a memoria son a instrucciones y que de cada 10 referencias a datos, 8 son de lectura y 2 de escritura. La tasa de aciertos con esta configuración es del 98%. Calcule el tiempo medio que consume un acceso a memoria suponiendo que el tamaño del bloque es de una palabra.

SOLUCIÓN:

El cálculo del tiempo medio de acceso debe distinguir entre lecturas y escrturas para tener en cuenta la diferencia debida a la política de escritura:

$$\bar{t}_a = p_l \cdot \bar{t}_l + p_e \cdot \bar{t}_e$$

Donde p_l y p_e son las probabilidades de acceso en lectura y escritura respectivamente y $\bar{t}_l$ y $\bar{t}_e$ los tiempos medios de acceso en lectura y escritura respectivamente. Para la probabilidad de lectura tenemos:

$$p_l = 0,53 + 0,47 \cdot 0,8 = 0,906$$

Y para la de escritura:

$$p_e = 0,47 \cdot 0,2 = 0,094$$

Para calcular los tiempos medios de acceso acudimos a la siguiente expresión:

$$\bar{t} = t_{\text{nivel 1}} + (1 - h) \cdot t_{\text{nivel 2}}$$

En el caso de las lecturas nos queda:

$$\bar{t}_l = 20 + (1 - 0,98) \cdot 100 = 22ns$$

En el caso de las escrituras no podemos aplicar esta fórmula ya que la política de escritura es directa y, por tanto, el tiempo de acceso es el de la memoria principal, esto es:

$$\bar{t}_e = 100ns$$

En definitiva, el tiempo medio de acceso es:

$$\bar{t}_a = 0,906 \cdot 22 + 0,094 \cdot 100 = 29,332ns$$

Vemos como pasamos de un tiempo medio de acceso de $100ns$ a un poco más de $30ns$.

19. Un computador dispone de una memoria caché con un tiempo de acceso de 20ns por palabra y una memoria principal con un tiempo de acceso de 100ns por palabra. La política de escritura de la memoria caché es escritura directa. Mediante la ejecución de una serie de programas de prueba se ha observado que el 53% de las referencias a memoria son a instrucciones y que de cada 10 referencias a datos, 8 son de lectura y 2 de escritura. Mediante simulaciones se ha comprobado que si se usan cachés separadas en instrucciones y datos, la tasa de aciertos en instrucciones llega al 99% mientras que en datos alcanza el 98%. Calcule el tiempo medio que consume un acceso a memoria suponiendo que el tamaño del bloque es de una palabra.

SOLUCIÓN:

Ahora hemos de desagregar tanto en accesos de lectura y escritura como en flujo de instrucciones y de datos, puesto que la caché es separada para ambos flujos.

$$\bar{t}_a = \bar{t}_{a_i} + \bar{t}_{a_d} = (p_l \cdot \bar{t}_l)_i + (p_l \cdot \bar{t}_l + p_e \cdot \bar{t}_e)_d$$

Como vemos, en el caso del flujo de instrucciones solo hay lecturas ya que las instrucciones no se escriben.

Tenemos:

$$\bar{t}_{l_i} = 20 + (1 - 0,99) \cdot 100 = 21ns$$

$$\bar{t}_{l_d} = 20 + (1 - 0,98) \cdot 100 = 22ns$$

$$\bar{t}_{e_d} = 100ns$$

De donde:

$$\bar{t}_a = 0,53 \cdot 21 + 0,47 \cdot 0,8 \cdot 22 + 0,47 \cdot 0,2 \cdot 100 = 28,8ns$$

Observamos que las cachés separadas proporcionan una mejora del 1,8% respecto al uso de caché unificada.