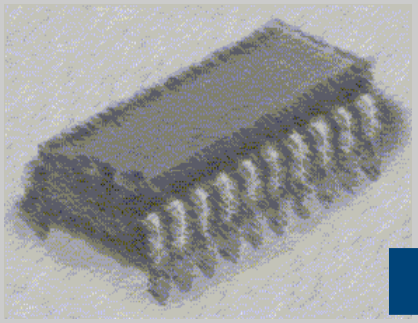
Enunciados de problemas

Tema 5. El sistema de memoria



Arquitectura de Computadores

I. T. Informática de Gestión

Curso 2009-2010

Base teórica

El sistema de memoria es el lugar donde residen los programas y datos ya que según la arquitectura von Neumann un programa debe estar almacenado en memoria para poder ser ejecutado

El sistema de memoria se organiza de manera jerárquica de forma que se tenga un sistema de alta capacidad y velocidad próxima a la de los dispositivos más rápidos y un coste cercano al de los dispositivos más lentos y baratos, tal y como muestra la figura siguiente.

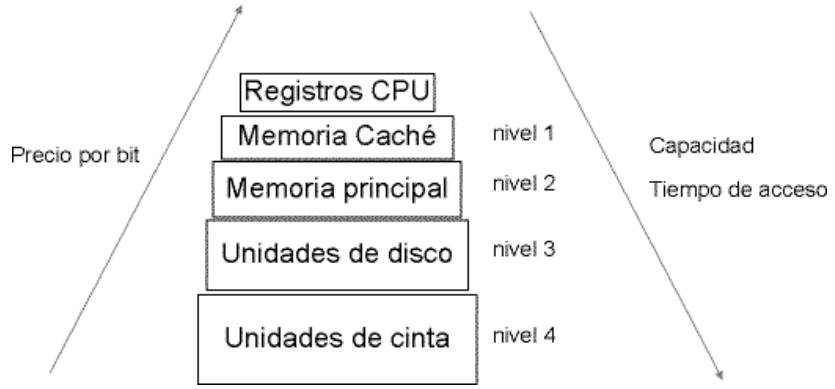


Figura 1. Jerarquía del sistema de memoria

Los principios de localidad espacial y temporal hacen que el sistema de memoria pueda ser eficiente y determina que la relación entre los diferentes niveles de la jerarquía sea a nivel de bloque de palabras y no de una única palabra

Principio de	Comentarios
Localidad temporal	Si se referencia a un elemento, éste tendera a ser referenciado en un corto espacio de tiempo ▪ Los datos generados en operaciones siguientes ▪ Las mismas instrucciones dentro de un bucle
Localidad espacial	Si se referencia a un elemento, los elementos cercanos a él tenderán a ser referenciados también ▪ Los datos pertenecientes a un vector ▪ La estructura secuencial de un programa

El hecho de encontrar (acierto) o no encontrar (fallo) un elemento en un determinado nivel de la jerarquía determinará la bondad y el rendimiento del sistema completo

Mejora de la memoria principal

Para aumentar el rendimiento de la memoria principal se debe aumentar el ancho de banda de la misma

Mecanismos para aumentar el ancho de banda de MP (figura 2)

- Reducción de su tiempo de acceso
- Aumento del tamaño de palabra
- Permitir el acceso concurrente a varios módulos de memoria, organización entrelazada.

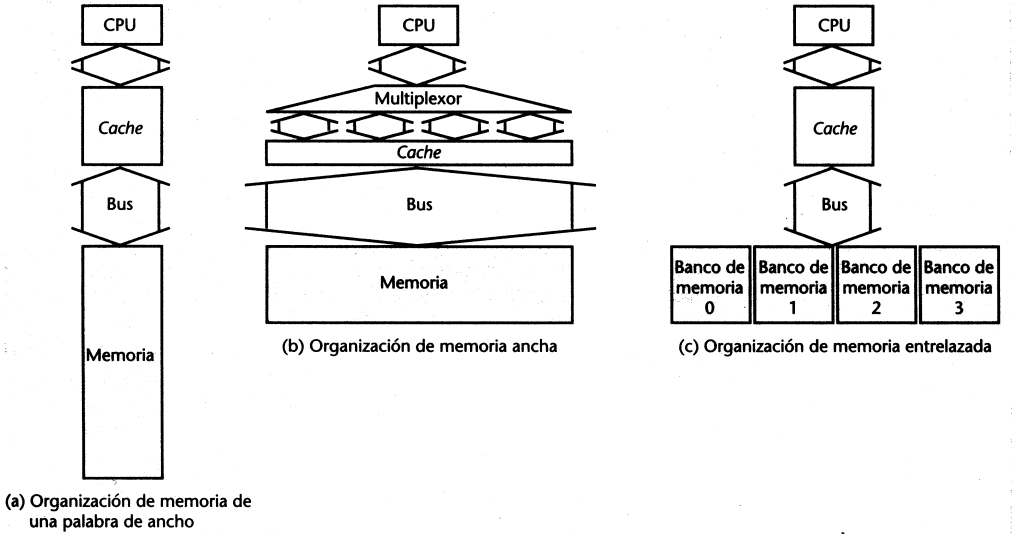


Figura 2. Aumento del rendimiento de la memoria

Mejora de la velocidad del sistema. La memoria cache

El objetivo de la memoria cache es dar la impresión de que las referencias memoria se sirven a una velocidad muy cercana a la del procesador

En el diseño de la MCaché: se debe tener en cuenta la optimización de los siguientes parámetros:

- Probabilidad de acierto
- Tiempo de acceso efectivo
- Retardos debidos a fallos (actualizar de la MP en caso de escritura)
- Establecer la correspondencia entre bloques de MP y MCa

Para ello se deben definir adecuadamente y tener en cuenta las políticas y parámetros siguientes:

- **Política de ubicación:** correspondencia entre bloques de MP y MCa
- **Política de extracción:** qué y cuándo se envían datos de MP a MCa
- **Política de reemplazo:** qué bloque abandona MCa para dejar espacio a otro de MP si está llena el bloque o el conjunto de la cache
- **Política de actualización:** cuándo se escribe en la MP
- **Unicidad y homogeneidad de la Mca:** una o varias MCa
- Tamaño más adecuado de la Mca y de sus bloques
- Minimización del tiempo de espera en caso de fallo en MCa

Políticas de ubicación

Determina la correspondencia entre bloques de MP y MCa tal y como se muestra en la figura 3 dichas políticas son:

Correspondencia directa

Consiste en hacer corresponder a todo bloque i de MP el bloque (i mod k) de MCa, donde k es el número total de bloques de la MCa

Una dirección en MCa consta de:

- Etiqueta

- Nº de bloque de MCa
- Posición en el bloque (palabra)

Totalmente asociativa

Cualquier bloque de MP puede ubicarse en cualquier bloque de la Mca

Una dirección en MCa consta de:

- Etiqueta
- Posición en el bloque (palabra)

Asociativa por conjuntos

Consiste en dividir la MCa en C conjuntos de B bloques cada uno. Se aplica correspondencia directa a nivel de conjunto y correspondencia asociativa a nivel de bloque.

Un bloque i de MP puede ubicarse en cualquier bloque del conjunto (i mod C) de MCa

Una dirección en MCa consta de:

- Etiqueta
- Conjunto
- Posición en el bloque

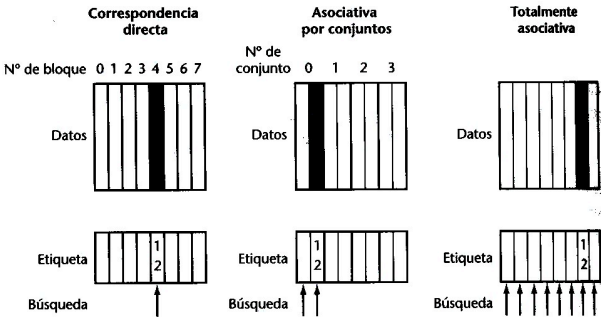


Figura 3. Políticas de ubicación

Políticas de extracción

Indican cuándo y qué información se envía de MP a MCa. Las más utilizadas son:

- **Extracción por demanda**: traer a MCa el bloque en el momento en que se necesita, es decir, cuando se produce un fallo
- **Extracción con anticipación:**
    - Anticiparse siempre: traer a MCa el bloque i+1 al referenciar el i
    - Anticiparse si se produce fallo: traer a MCa el bloque i+1 de MP solamente cuando se produce un fallo al refernciar el bloque i
    - Anticipación marcada: traer el bloque i+1 de MP al producirse el fallo al referenciar el i, marcarlo como traído y al referenciar el i+1 traer el i+2
- **Extracción selectiva:** marca algún tipo de información para que nunca pueda ser enviada a MCa

Políticas de reemplazo

Determina cuándo y qué bloque se sustituye en MCa porque al traer un bloque a cache cuando todos los marcos se encuentran ocupados

Únicamente se da en las caches asociativas y asociativas por conjuntos ya que se puede elegir que bloque sale. En las de correspondencia directa un bloque de MP siempre va a un bloque de MCa

Las más utilizadas son:

- **Aleatoria:** consiste en elegir el bloque a reemplazar de forma aleatoria
- **LRU Least-Recently Used:** consiste en reemplazar aquel bloque de MCa que no ha sido utilizado durante el mayor periodo de tiempo
- **FIFO - First In First Out:** consiste en reemplazar el bloque que ha permanecido en MCa el mayor periodo de tiempo

Políticas de actualización

Determina cuándo se actualiza la información en MP al haberse producido una escritura en MCa para evitar los problemas de falta de coherencia

- **Escritura inmediata:** consiste en escribir en MCa y MP a la vez
- **Escritura aplazada:** consiste en escribir en MCa y únicamente se escribe en MP si el bloque a reemplazar ha sido modificado

Las maneras más empleadas de actuar ante un fallo de escritura son:

- **Con ubicación:** consiste en llevar el bloque que produce el fallo de MP a MCa y a continuación realizar la escritura en MCa. Suele ir unida a la escritura aplazada
- **Sin ubicación:** consiste en escribir únicamente en MP. Suele ir asociada a la escritura inmediata

Minimización del tiempo de penalización por fallo

Las maneras más empleadas de actuar ante un **fallo de lectura** son:

- **Early start:** consiste en enviar al procesador la palabra que produjo el fallo tan pronto como llegue a cache y sin esperar a que se complete la transferencia del bloque
- **Out of order fetch:** consiste en enviar la palabra que produjo el fallo a MP y a MCa y luego terminar de transferir el bloque

La manera más empleada de actuar ante un **fallo de escritura** es el empleo de un buffer de escritura, de tal manera que en el caso de escritura sin ubicación se escribe en el buffer y éste luego escribirá la información en MP

Otros parámetros. Tamaño y unicidad de cache

Tamaño de la memoria cache y sus bloques

Aumentar el tamaño del bloque hace que aumenta la tasa de aciertos, por la localidad espacial y que disminuya el número de bloques en MCa por la localidad temporal

Disminuir el tamaño del bloque hace que disminuya la cantidad de tráfico de información entre MCa y MP

Unicidad y homogeneidad de la memoria cache

Las instrucciones presentan localidad temporal y los datos espacial el tener dos caches separadas una para datos y otra para instrucciones permite el acceso simultaneo a instrucciones y datos (aumenta la velocidad) y definir las políticas más adecuadas para cada una de ellas

Cache de uno y dos niveles

Existe una mejora de las prestaciones con una MCa de dos niveles. Normalmente el primer nivel de MCa se encuentra integrado en el procesador lo que elimina el acceso al bus y aumentan los tiempos de ejecución

Mejora de la capacidad del sistema. La memoria virtual

El objetivo es dar la impresión de que el computador dispone de una memoria ilimitada. Para ellos combina dos dispositivos de almacenamiento de diferente capacidad, velocidad y coste

Separa los conceptos de direcciones lógicas (capacidad de direccionamiento) y direcciones físicas (la memoria instalada) – figura 4 -

La memoria virtual se divide en bloques y el intercambio entre memoria secundaria y MP se hace por bloques. La memoria virtual implementa un mecanismo por el que únicamente los bloques de memoria que se están usando se encuentren en memoria principal y el resto en memoria secundaria. Además, la memoria virtual la suele gestionar el sistema operativo.

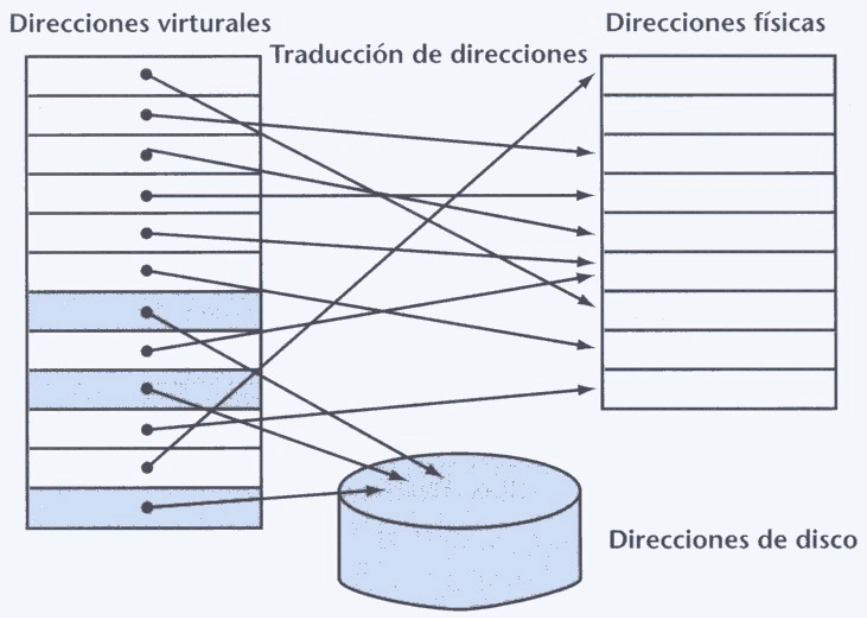


Figura 4. Esquema de traducción de direcciones de MV a MP

La traducción es necesaria debido a que los programas generan direcciones lógicas pero para la ejecución se necesitan direcciones físicas

Se trata de definir una función F tal que:

- F(x) = y si la información contenida en la dirección lógica x se encuentra en memoria principal en la dirección física y
- F(x) = 0 si la información no se encuentra en memoria principal. Al producirse el fallo el sistema deberá:
    - Interrumpir la ejecución del proceso que provoca el fallo
    - Traer la información referenciada de la memora auxiliar a una dirección física de memoria principal
    - Modificar la función de traducción para reflejar la nueva correspondencia
- Una vez completada la transferencia, continuar con la ejecución del proceso

Alternativas para la implementación de la función de traducción:

- **Paginación:** se dividen los espacios lógicos y físicos en bloques del mismo tamaño denominados paginas

- **Segmentación:** se dividen los espacios lógicos y físicos en bloques de tamaño variable denominados segmentos
- **Segmentación paginada:** mecanismo híbrido de los anteriores

1. Sea un ordenador con un procesador de 32 bits, con un sistema de memoria de las siguientes características:

   - Memoria principal con un tiempo de acceso de 32 ns.
   - Dos módulos de memoria caché independientes (instrucciones y datos) estructuradas en bloques de 16 bytes, a los que se accede en 4 ns.
   - Ambas memorias, tienen la política de ubicación directa y política de actualización con escritura inmediata Además, la memoria caché de datos tiene un tamaño de 64 Kbytes.

   En este computador se ejecuta el siguiente código:

```
; Realizar el cálculo del total de una compra almacenando el
; resultado en la posición correspondiente a la variable Total
; mediante el producto de las listas precio y cantidad
WHILE (i ≤ nprod) DO
BEGIN
        Total := Total + precio[i] * cantidad[i];
        i:= i + 1;
END;
```

Si inicialmente, la memoria caché de datos se encuentra vacía:

a.- Indicar la correspondencia entre una dirección de memoria principal y una de memoria cache

b.- Indicar la traza de los 4 primeros accesos, a los datos, realizada por este fragmento de código suponiendo que:

- Las variables ***i*** y ***Total*** se almacenan en dos de los registros del procesador, inicializadas a 0,
- Las listas ***precio*** y ***cantidad*** se almacenan a partir de las direcciones de memoria principal 3001Ah y 9001Ah respectivamente.

c.- Indicar en cuáles de las anteriores referencias a la memoria principal producen fallos en la memoria caché, considerando las 4 iteraciones que realiza el bucle (***nprod = 4***)

d.- Calcular la tasa de aciertos de la memoria caché de datos para el anterior fragmento de programa

---

2. Resolver el ejercicio anterior con la diferencia de que la política de ubicación de las memorias caches de datos e instrucciones son asociativas por conjuntos de 2 bloques cada uno y la política de reemplazo es FIFO

---

3. Se dispone de un ordenador con buses de datos de 16 bits y direcciones de 24 bits. El sistema de memoria se organiza de la siguiente forma:
   - Memoria principal entrelazada simple de 16 módulos con un tiempo de acceso de 100 ns.
   - Memoria caché unificada (común para instrucciones y para datos), con las siguientes características:
     - Capacidad: 128 K palabras
     - Tiempo de acceso: 20 ns.
     - Política de ubicación: asociativa por conjuntos de 2 bloques, cada uno de ellos con 16 palabras
     - Política de reemplazo: LIFO
     - Política de actualizacón: inmediata sin ubicación
     - En caso de fallo de lectura se emplea la política out of order fetch

En este computador se ejecuta el siguiente programa:

Dir. en MP de la instrucción	Dir. en MP del dato	Pseudocódigo	
011110 h		Llamar al procedimiento PedirDatos.	Programa principal
011112 h		Llamar al procedimiento CodificarDatos.	
031110 h		Sacar mensaje solicitando un dato	Procedimiento PedirDatos
031112 h	001119 h	Leer el dato desde teclado y almacenarlo en la var. Datoleído	
031114 h		Retorno al programa principal	
041110 h	001119 h	Realizar un XOR de Datoleído con 70h	Procedimiento CodificarDatos
041112 h	001119 h	Invertir el bit más significativo de Datoleído	
041114 h		Retorno al programa principal.	

Se sabe que las operaciones

- Datoleido := Datoleido XOR 70h;
- Datoleido := Datoleido XOR 8000h;

implican la lectura de la variable Datoleído y su posterior escritura

Suponiendo que inicialmente la memoria caché de instrucciones está vacía y la variable ***Datoleído*** se encuentra almacenada en la dirección de memora principal 001119 h:

a.- Establecer la correspondencia entre una dirección de memoria principal y de memoria caché.

b.- Indicar cuales de las anteriores referencias a memoria principal producen fallos en memoria caché, considerando la ejecución completa del programa

c.- Calcular el tiempo de ejecución del programa y la tasa de aciertos

---

4. Resolver el ejercicio anterior con la diferencia de que la política de actualización es inmediata con ubicación

---

5. Sea un ordenador con buses de datos y de direcciones de 32 bits. El sistema de memoria se organiza según:
   - Memoria principal entrelazada simple de 16 módulos con un tiempo de acceso de 20 ns.
   - Memoria caché unificada (común para instrucciones y para datos), con las siguientes características:
     - Tamaño 128 Kpalabras
     - Tiempo de acceso: 4 ns
     - La política de ubicación: asociativa por conjuntos de 2 bloques, cada uno de ellos con 16 palabras
     - Política de reemplazo: FIFO
     - Política de escritura: escritura aplazada sin ubicación

En este computador se ejecuta el siguiente programa:

Dir. en MP de la instrucción	Dir. en MP del dato	Pseudocódigo	
0001 0004 h		Llamar al procedimiento PedirDatos	Programa principal
0001 0008 h		Llamar al procedimiento CodificarDatos	
0008 0002 h		Sacar mensaje solicitando un dato	Procedimiento PedirDatos
0008 0006 h	0000 0009 h	Leer el dato desde teclado y almacenarlo en la variable DatoLeído	
0008000A h		Retorno al programa principal	
0006 0000 h	0000 0009 h	Realizar una XOR de DatoLeído con 70 h	Procedimiento CodificarDatos
0006 0004 h	0000 0009 h	Invertir el bit más significativo de DatoLeído	
0006 0008 h		Retorno al programa principal	

Se sabe que las operaciones

- Datoleido := Datoleido XOR 70h;
- Datoleido := Datoleido XOR 8000h;

implican la lectura de la variable Datoleído y su posterior escritura

Suponiendo que inicialmente, la memoria caché de instrucciones está vacía y la variable ***DatoLeído*** se encuentra almacenada en la dirección de memoria principal 0000 0009 h:

a.- Indicar la traza de la ejecución realizada por este programa

b.- Indicar cuál es la correspondencia entre una dirección de memoria principal y de memoria caché

c.- Indicar cuáles de las anteriores referencias a la memoria principal producen fallos en la memoria caché, considerando la ejecución completa del programa

d.- Calcular el tiempo que tarda en ejecutarse el programa en función de los accesos a memoria

---

6. Resolver el ejercicio anterior con la diferencia de que la política de actualización es aplazada con ubicación

---

7. Sea un computador con el siguiente sistema de memoria:
   - Memoria principal:
     - Capacidad: 4 Mbytes
     - Tiempo de acceso: 64 ns
   - Memoria caché:
     - Unificada para datos e instrucciones
     - Capacidad de 16 Kbytes
     - Tamaño de bloque de 16 bytes
     - Tiempo de acceso: 9 ns
     - Política de ubicación: asociativa por conjuntos de 4 bloques cada uno
     - Política de actualización: escritura inmediata sin ubicación
     - Política de reemplazo: FIFO
     - Como mecanismo para minimizar el tiempo de espera ante un fallo de lectura se emplea la técnica del "out of order fetch"

En este computador se ejecuta el siguiente programa:

Dirección de memoria	Instrucción	
5010h	Llamar Procedimiento 1	
5014h	Llamar Procedimiento 2	Programa principal
5018h	Llamar Procedimiento 3	
6014h	Leer dato por teclado	Procedimiento 1
6018h	Almacenar en DatoLeido	
601Ch	Retorno de procedimiento	
7010h	Restar 30h a DatoLeido	Procedimiento 2
7014h	Retorno de procedimiento	
8018h	Sumar a Suma Datoleido	Procedimiento 3
801Ch	Retorno de procedimiento	

Se sabe que las operaciones

- Datoleido := Datoleido - 30h;
- Suma := Suma + Datoleido;

implican la lectura de las variables ***Datoleído*** y ***Suma*** y su posterior escritura

Se supone que las variables ***Suma*** y ***Datoleido*** se encuentran en las posiciones de memoria 2001Ch y 30014h.

a.- Establecer la correspondencia entre una dirección de memoria principal y una de memoria caché

b.- Indicar cuáles de las anteriores referencias a la memoria principal producen fallos en la memoria caché, considerando la ejecución completa del programa

c.- Calcular el tiempo de ejecución del siguiente programa (considerando solamente los accesos a memoria)

8. Resolver el ejercicio anterior con la diferencia de que la política de actualización es aplazada con ubicación

9. Sea un ordenador con un sistema de memoria de las siguientes características:

   - Memoria principal:
     - Capacidad 4 GB.
     - Entrelazada simple de orden inferior con 16 módulos.
     - Tiempo de acceso de 32 ns.
   - Memoria cache:
     - Dos módulos de memoria caché independientes (instrucciones y datos)
     - Capacidad de ambas caches de 256 Kbytes.
     - Tiempo de acceso de 4 ns.
     - Bloques de 16 bytes.
     - **Política de ubicación**: asociativa por conjuntos de cuatro bloques.
     - **Política de actualización**: escritura inmediata sin ubicación.
     - **Política de reemplazo**: FIFO.

En este computador se ejecuta el siguiente código:

```
i := 1;
  WHILE (i < 3) DO
  BEGIN
        A  := Vector1(i);
        B:= Vector2(i);
        IF A > B
        THEN BEGIN
                Mayor(i) := A;
                Menor(i) := B;
        END
        ELSE BEGIN
                Mayor(i) := B;
                Menor(i) := A;
        END;
        Suma(i) := Suma(i) + Mayor(i);
        i:= i + 1;
  END;
```

Se sabe que la instrucción *Suma(i) := Suma(i) + Mayor(i)* implica tres accesos a memoria: dos de lectura *suma(i)* y *mayor(i)* y otro de escritura del resultado en *suma(i)*

También se sabe que la variable *i* se encuentra ubicada en un registro, inicializada a 1.

Finalmente las direcciones de memoria de las variables son:

Mayor	3C89 2365	Vector1	0039 2364
Menor	3D89 236A	Vector2	0029 2368
Suma	0019 2367		

Si inicialmente, la memoria caché de datos se encuentra vacía:

a) Indicar la correspondencia entre una dirección de memoria principal y una de memoria cache.

b) Indicar la traza de los accesos, a los datos, realizada por este fragmento de código.

c) Indicar en cuáles de las anteriores referencias a la memoria principal producen fallos en la memoria caché y que acción se realiza.

d) Calcular la tasa de aciertos de la memoria caché de datos para el anterior fragmento de programa.

e) Calcular el tiempo de ejecución del fragmento del programa anterior debido únicamente a los accesos a memoria.

---

10. Resolver el ejercicio anterior con la diferencia de que la política de actualización es aplazada con ubicación

---

11. Sea un ordenador con un procesador de 8 bits, con un sistema de memoria de las siguientes características:

- Memoria principal:
  - Capacidad 4 GB
  - Entrelazada simple de orden inferior con 16 módulos
  - Tiempo de acceso de 32 ns.
- Memoria cache:
  - Dos módulos de memoria caché independientes (instrucciones y datos)
  - Bloques de 16 bytes.
  - Tiempo de acceso de 4 ns.
  - **Política de ubicación**: asociativa por conjuntos de dos bloques.

- **Política de actualización**: escritura inmediata sin ubicación.
- **Política de reemplazo**: FIFO.
- Capacidad de ambas caches de 64 Kbytes.

En este computador se ejecuta el siguiente código:

```
; Realizar el cálculo del total de una compra almacenando el
; resultado en la posición correspondiente a la variable Total
; mediante el producto de las listas precio y cantidad
WHILE (i ≤ nprod) DO
BEGIN
      Total := Total + precio[i] * cantidad[i];
      i:= i + 1;
END;
```

Si inicialmente, la memoria caché de datos se encuentra vacía:

a) Indicar la correspondencia entre una dirección de memoria principal y una de memoria cache.

b) Indicar la traza de los 3 primeros accesos, a los datos, realizada por este fragmento de código suponiendo que:
   - Las variables i, nprod y Total se almacenan en dos de los registros del procesador, inicializadas a 0,
   - Las listas precio y cantidad se almacenan a partir de las direcciones de memoria principal 80018h y 70018h respectivamente.

c) c.- Indicar en cuáles de las anteriores referencias a la memoria principal producen fallos en la memoria caché, considerando las 3 iteraciones que realiza el bucle.

d) d.- Calcular la tasa de aciertos de la memoria caché de datos para el anterior fragmento de programa.

e) Calcular el tiempo de ejecución del fragmento del programa anterior debido únicamente a los accesos a memoria.