Electronica digital 2

caso de estudio de cache

Ferney Alberto Beltrán Molina

Agosto 2019

Contacto

Nombre:	Ferney Alberto Beltrán Molina, Ing, MSc, PhD(c)
Email:	fabeltranm@unal.edu.co
oficina:	

Contenido

Jerarquía de Memoria

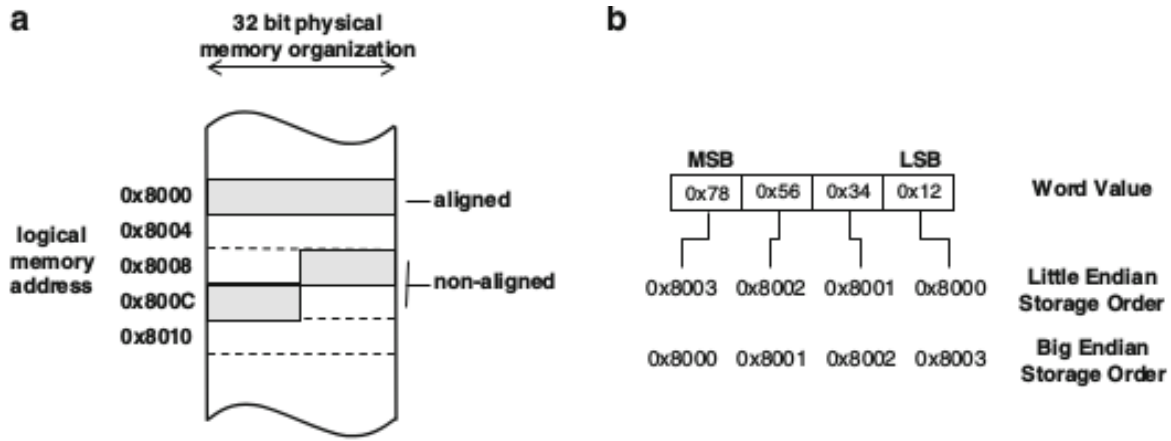
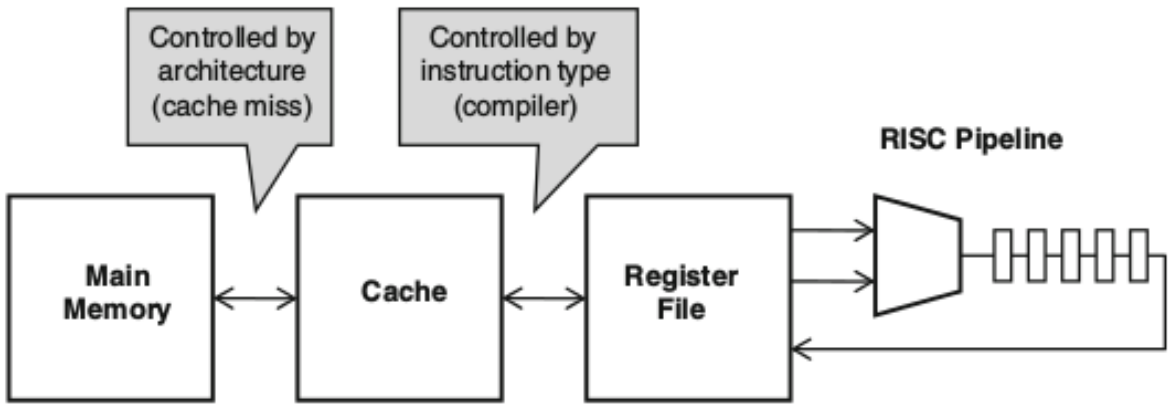
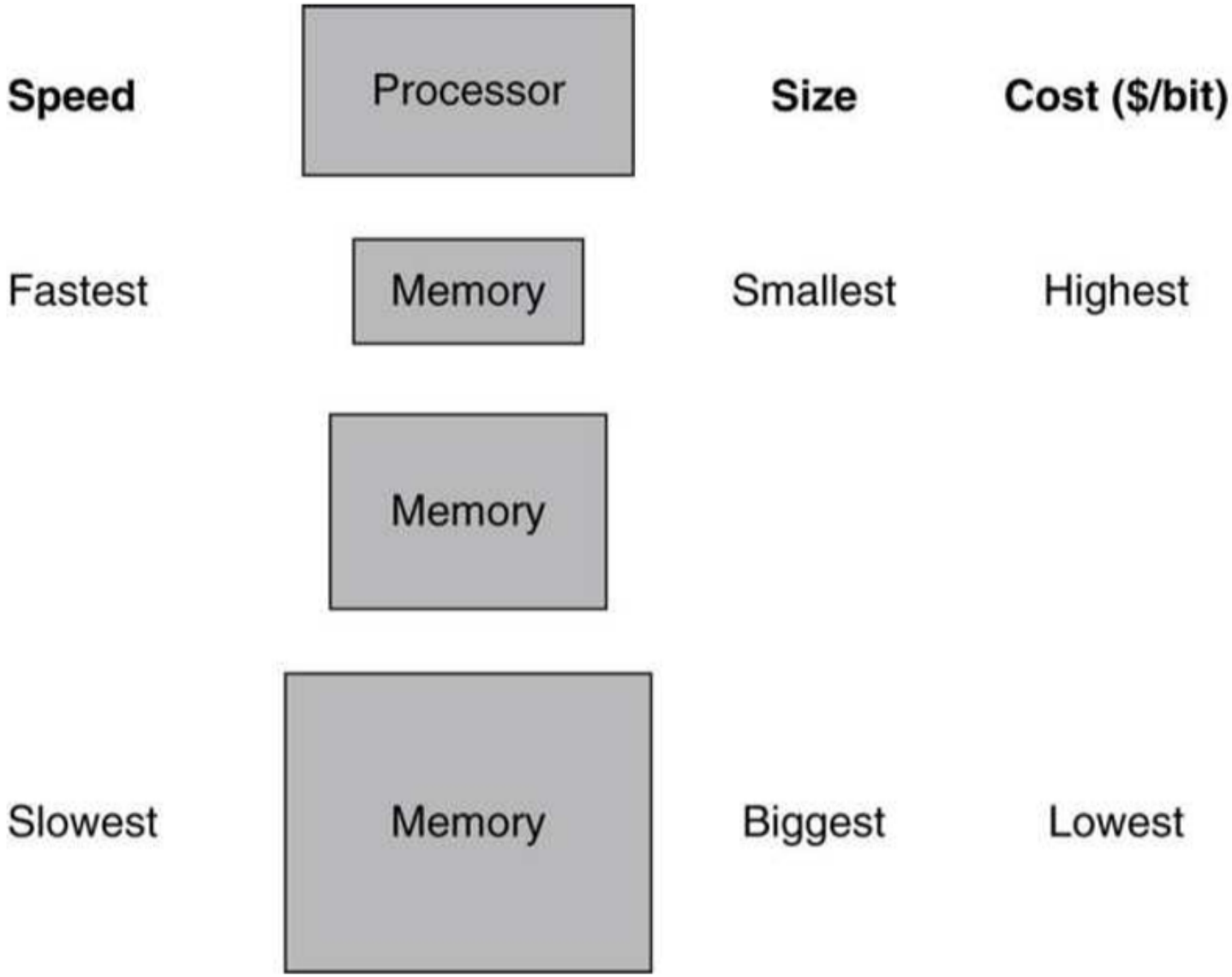


Fig. 7.7 (a) Alignment of data types. (b) Little-endian and Big-endian storage order

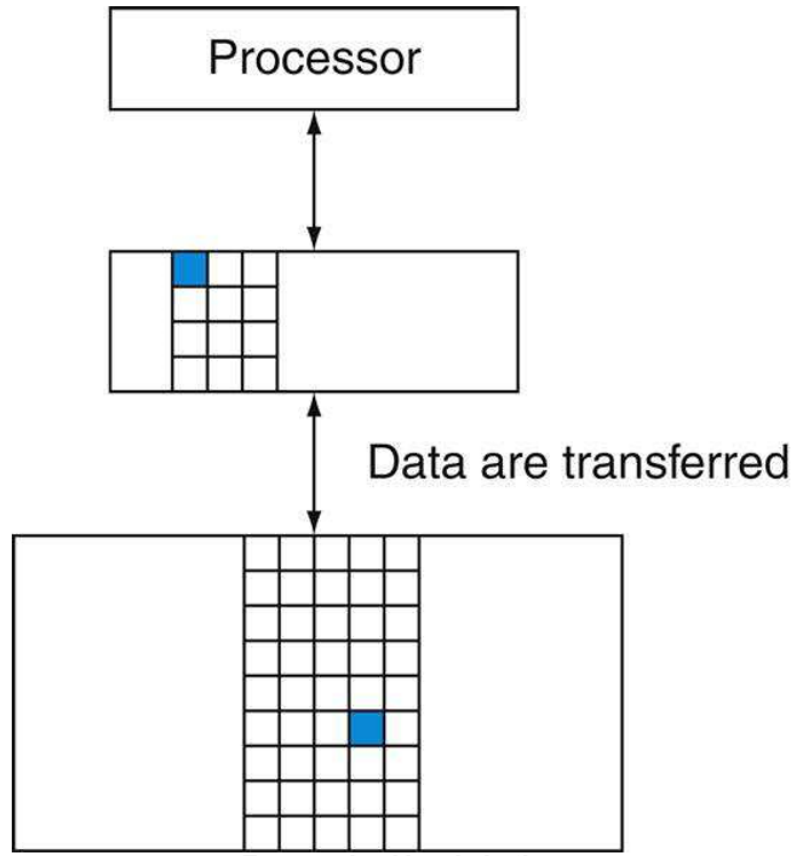


Jerarquía de Memoria

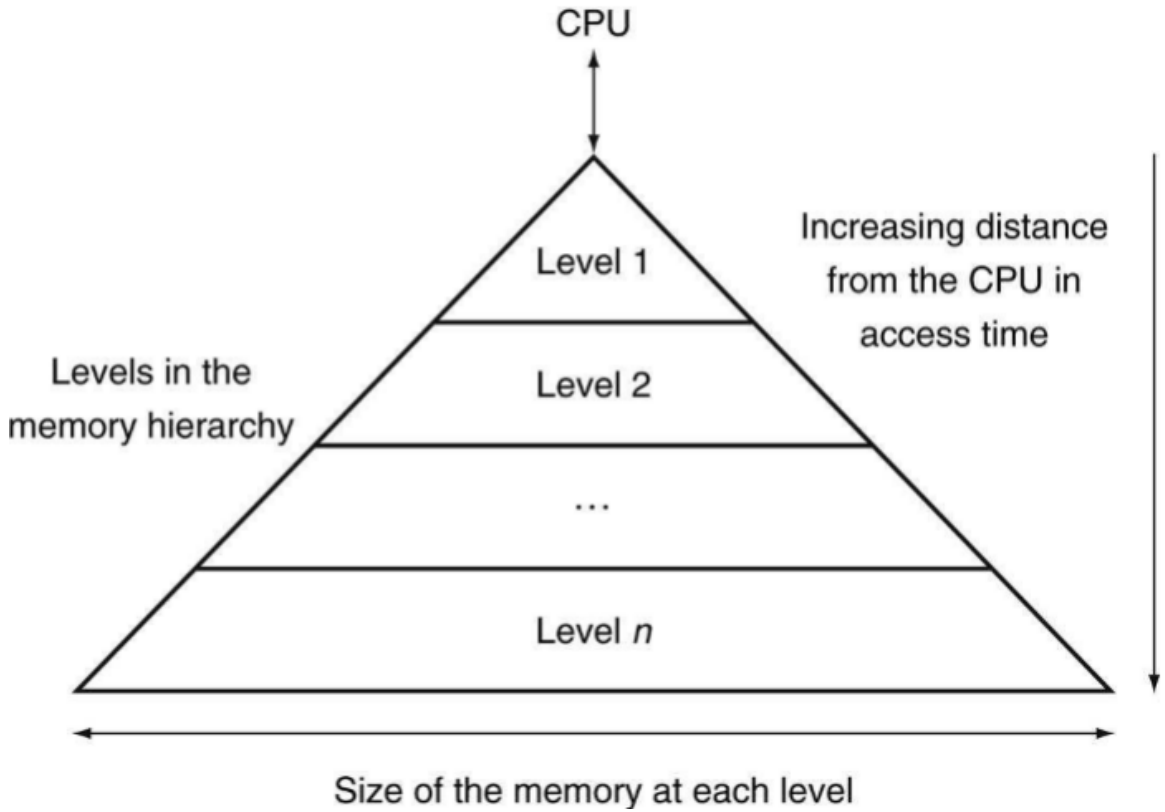


Jerarquía de Memoria

1. Blocks: Unidad mínima de información que puede estar presente o no en un caché.
2. Hit Rate: accesos a memoria que se encuentra en el nivel de la jerarquía de memoria.
3. Miss Rate: accesos de memoria que no se encuentra en la memoria

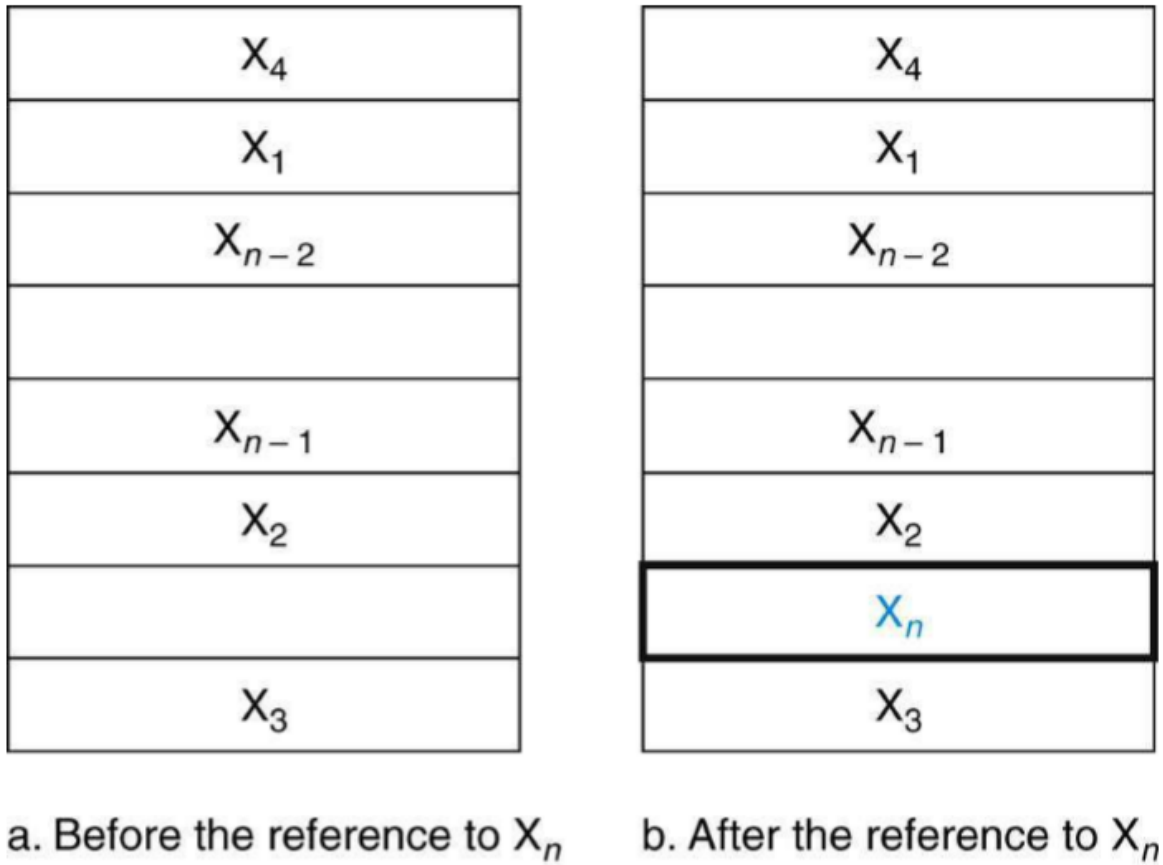


Jerarquía de Memoria



Hit time: tiempo requerido para acceder al nivel de memoria y tiempo necesario para determinar si es exitoso p no el acceso.
miss penalty: tiempo requerido para recuperar un bloque del nivel inferior + tiempo para acceder al bloque + tiempo transmisión de un nivel a otro + tiempo de insertar en el nivel que experimentó la falla.

Cache

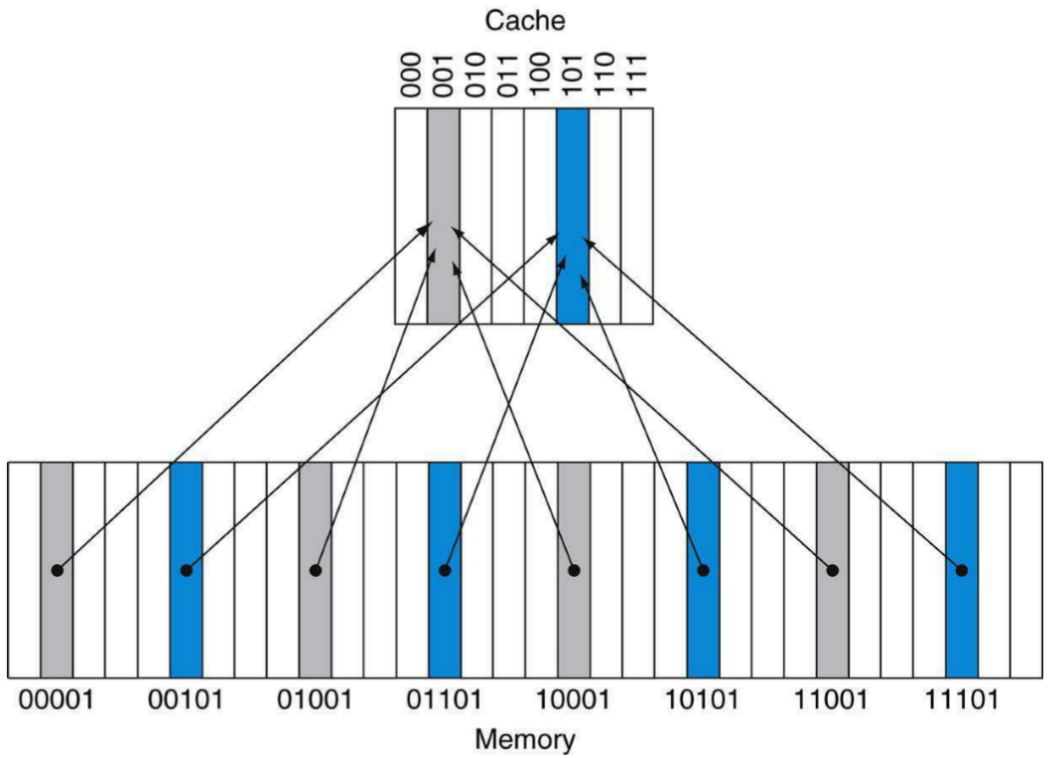


a. Before the reference to X_n b. After the reference to X_n

¿Cómo sabemos si un elemento de datos está en la memoria caché? ¿cómo lo encontramos?

direct-mapped cache

Cada ubicación de memoria se asigna exactamente a una ubicación de la memoria caché.

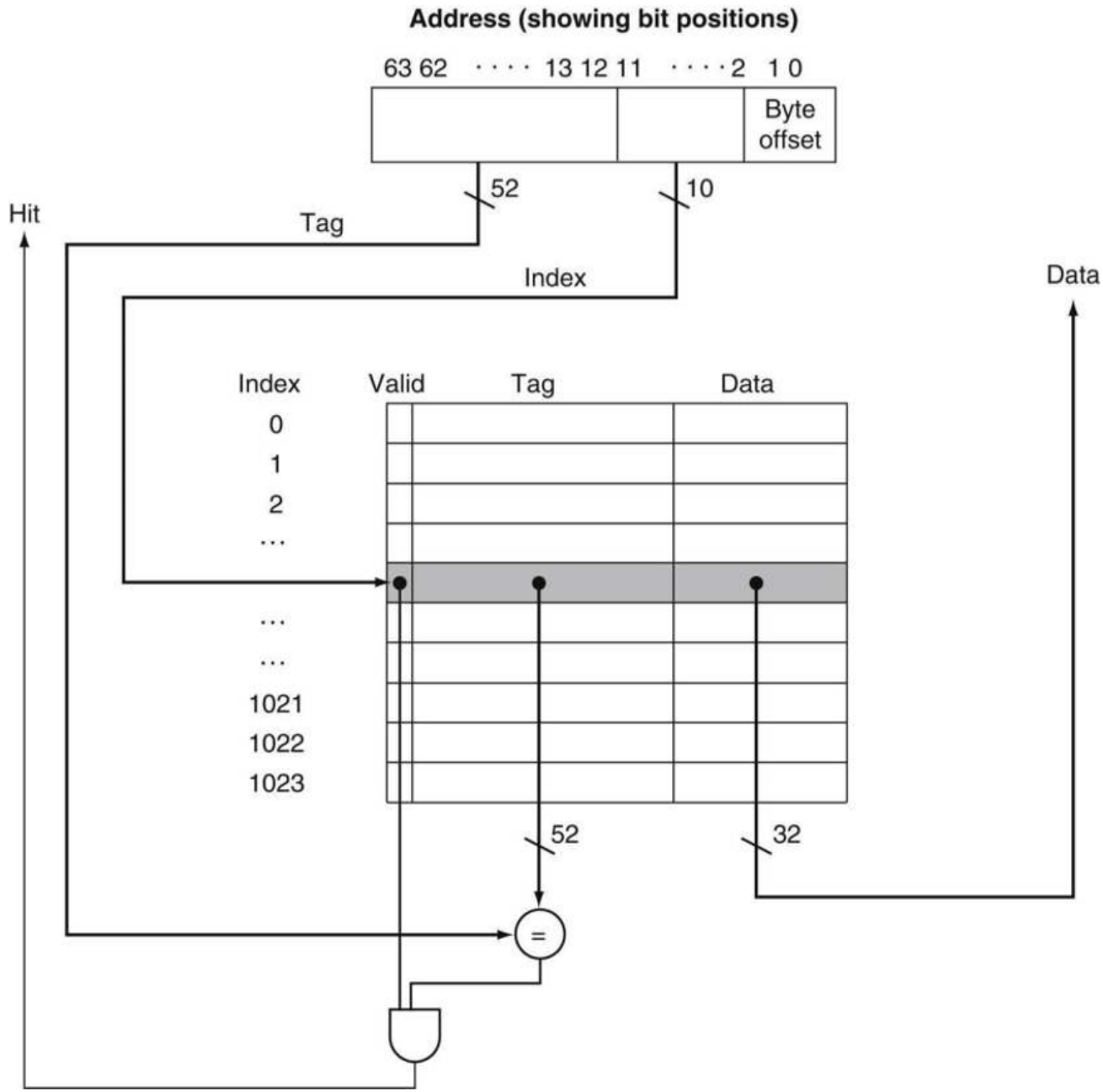


¿cómo sabemos si una palabra solicitada está en la memoria caché o no?

direct-mapped cache

1. **Tag**: utilizada para identificar si una jerarquía de memoria contiene el bloque asociado a la palabra solicitada en la dirección. ejm: 00011011
2. **valid bit**: indica que el bloque asociado en la jerarquía contiene datos válidos

Lectura Cache



Lectura Cache Ejercicio

Decimal address of reference	Binary address of reference	Assigned cache block (where found or placed)
22	10110_{two}	$(10110_{two} \bmod 8) = 110_{two}$
26	11010_{two}	$(11010_{two} \bmod 8) = 010_{two}$
22	10110_{two}	$(10110_{two} \bmod 8) = 110_{two}$
26	11010_{two}	$(11010_{two} \bmod 8) = 010_{two}$
16	10000_{two}	$(10000_{two} \bmod 8) = 000_{two}$
3	00011_{two}	$(00011_{two} \bmod 8) = 011_{two}$
16	10000_{two}	$(10000_{two} \bmod 8) = 000_{two}$
18	10010_{two}	$(10010_{two} \bmod 8) = 010_{two}$
16	10000_{two}	$(10000_{two} \bmod 8) = 000_{two}$

Index	V	Tag	Data
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	N		
111	N		

Número de bit de Cache (direct-mapped cache)

El número total de bits necesarios para la memoria caché es función del tamaño de la caché y el tamaño de la dirección. La memoria caché incluye el almacenamiento de los datos y las etiquetas:

1. El tamaño de la caché es 2^n bloques. Se utilizan n bits para el índice.
2. El tamaño del bloque es de 2^m palabras (2^{m+2} bytes), Se utilizan m bits para la palabra dentro del bloque y 2 bits para la dirección de byte
3. El tamaño del tag es $TamañoDirección - (n + m + 2)$

El número total de bits en una memoria caché de asignación directa es:

$$2^n(2^m \text{ X } 32 + (64 - n - m - 2) + 1)$$

con $TamañoDirección = 64$

Ejercicio: Número de bit de Cache (direct-mapped cache)

¿Cuántos bits totales se requieren para la caché de asignación directa, si se requiere almacenar 16 KB de datos y cada bloques es de 4 palabras?. Suponga una dirección de 64 bits

Solución: Número de bit de Cache (direct-mapped cache)

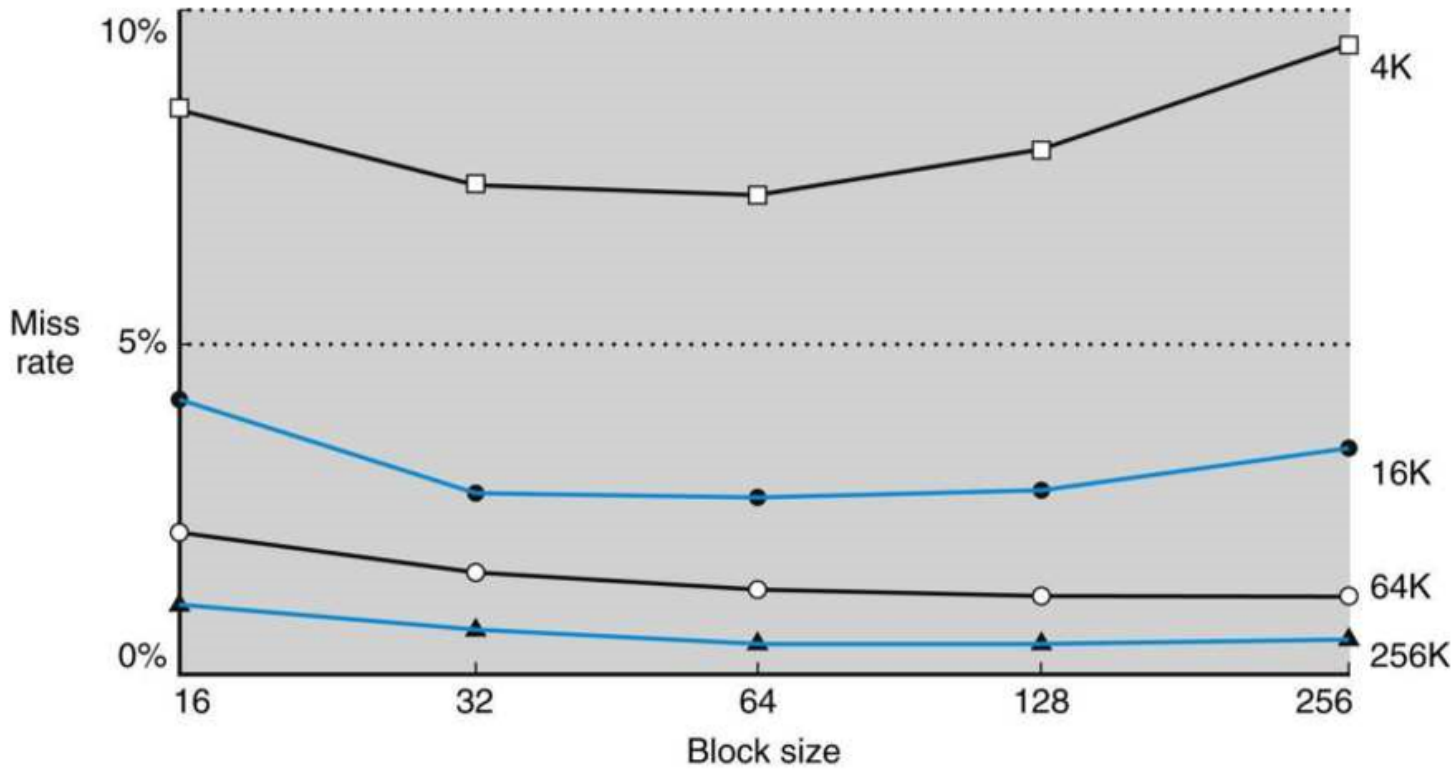
¿Cuántos bits totales se requieren para la caché de asignación directa, si se requiere almacenar 16 KB de datos y cada bloques es de 4 palabras?. Suponga una dirección de 64 bits

1. 16 kB = 4096 words (2^12).
2. 1 block = 4 words (2^2). $4X32 = 128bits$
3. total blocks = 1024 blocks (2^10).
4. tamaño de tag = $64 - 10 - 2 - 2 = 50bits$.

El tamaño total de bits para la cache de 16KB es:

$$2^{10}(128 + 50 + 1) = 2^{10}179 = 179Kbit = 22{,}375KB$$

Que pasa si se aumenta el tamaño de los blocks



A priori se evidencia disminución de miss pero ...
se aumenta la penalidad de fallo. Mayor latencia.

Recuerde: miss penalty es tiempo requerido para recuperar un bloque del nivel inferior y cargarlo en la memoria cache.

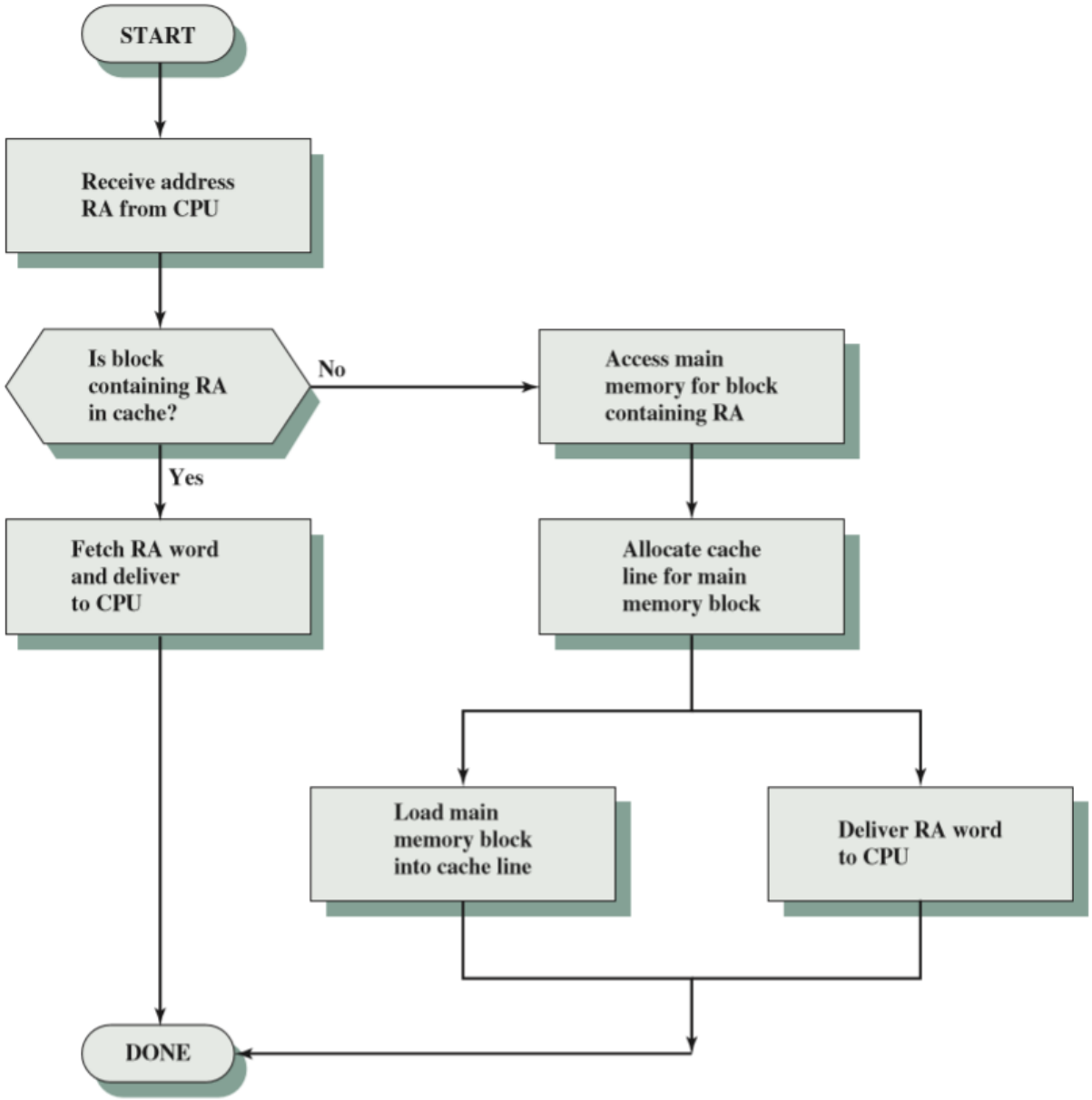
Ej: ¿Cómo mejorar la latencia? early restart, requested word first

Manejo de Cache Miss

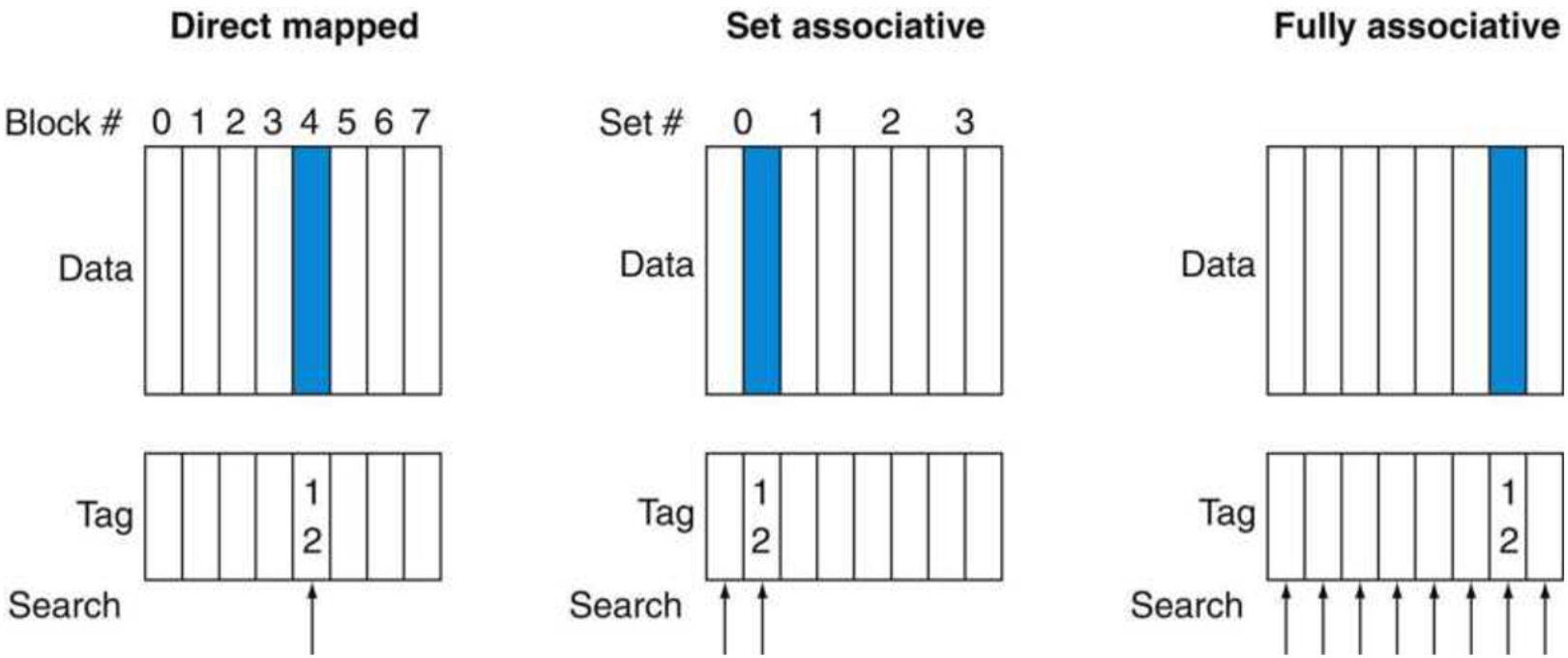
Básicamente se detiene el procesador (pipeline stall) hasta que la memoria responda con las instrucciones y datos.

1. Envíe el valor del PC a la memoria (anterior).
2. la UC solicita lectura de la instrucción a la memoria principal y espere a que la memoria complete su acceso.
3. se Escribe los datos en la caché, el tag (desde la ALU) y se activa el bit válido.
4. Se retorna la ejecución de la instrucción en el primer paso, lo que recuperará la instrucción, esta vez con caché Hit.

Resumen lectura Cache

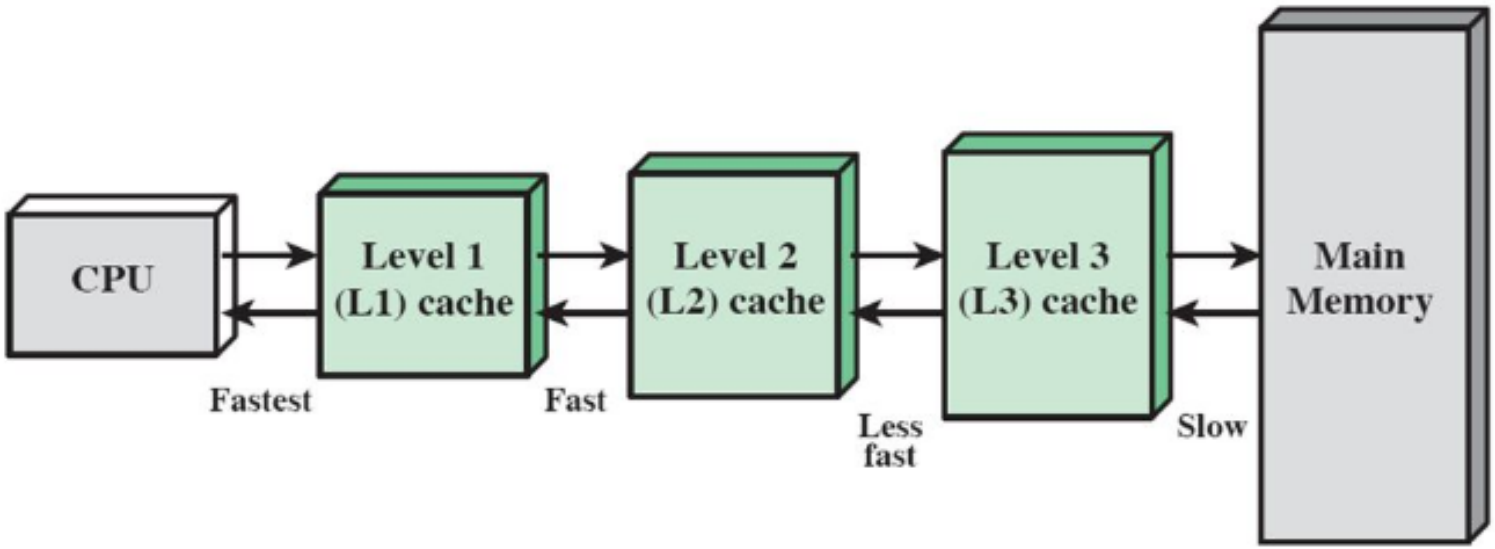


Técnicas de almacenamiento en Cache



Ej: Buscar ventajas o desventajas

Niveles de Cache



Mejora la penalidad de fallos

Ejercicio

Objetivo: Mejora la tasa de Éxito (hit rate) de lectura de la cache:

1. Cambiando el tamaño de memoria
2. Cambiando el tamaño de bloques
3. Cambiando el patrón de acceso a memoria

Herramientas - Ejercicio

1. Se Utiliza el programa online Venus o off-line Jupiter, que facilitan la visualización de hit a la memoria cache
2. El link de Venus es https://venus.cs61c.org/
3. se utiliza el siguiente pseudo código:

```
int array[tam];
for (k = 0; k < repcount; k++) {
  for (index = 0; index < arraysize; index += stepsize)
    if(option==0)
      array[index] = 0;
    else
      array[index] = array[index] + 1;
  }
}
# DONDE:
#  a0 = array size in bytes       -> tam
#  a1 = step size                 -> stepsize
#  a2 = number of times to repeat -> k
#  a3 = 0 (W) / 1 (RW)            -> option
```

Herramientas - Ejercicio

1. Los elementos de la matriz 'ARRAY' se determinan por el tamaño del registro a0, el cambio en memoria de dichos elementos están determinado por por el registro a1 y, cuantas veces se opera este acceso a memora lo determina el registro a2.
2. Estos parámetros afectarán directamente la cantidad de aciertos de memoria (hit) caché frente a errores (miss) que ocurrirán.
3. La opción del registro a3 también modifica el Hit rate de la memoria cache.

RECUERDE: Revisar las políticas de HIT y MISS.

Simulación - Ejercicio

1. Simule 3 escenarios y registe el hit rate de la cache, se gun el programa cache.s dado en clase.
2. Razone cuál seria el hit rate ANTES de simular el código.
3. luego de simular el código, debe entender el porque el resultado

Escenario 1 - Ejercicio

Configuración de Cache

1. Niveles de caché: 1
2. Tamaño de bloque: 8
3. Número de bloques: 4
4. Política de ubicación: Direct Mapping
5. Política de reemplazo de bloque: LRU

Configuración inicial de Código

1. a0: 128 (bytes)
2. a1: 8
3. a2: 4
4. a3: 0

Preguntas - Ejercicio

1. ¿Qué combinación de los parámetros de cache produce la tasa de aciertos?.
2. ¿Cómo afecta el Hit rate al aumentar el valor del registro a2? ¿Por qué?
3. ¿Cómo podríamos modificar un parámetro del Código para aumentar nuestra tasa de aciertos?

Escenario 2 - Ejercicio

Configuración de Cache

1. Niveles de caché: 1
2. Tamaño de bloque: 16
3. Número de bloques: 16
4. Política de ubicación: N-Way Set Associative
5. Asociatividad: 4
6. Política de reemplazo de bloque: LRU

Configuración inicial de Código

1. a0: 256 (bytes)
2. a1: 2
3. a2: 1
4. a3: 1

Preguntas - Ejercicio

1. ¿Cuántos accesos a memoria hay por cada iteración en el ciclo interno?.
2. ¿Cuál es el patrón que se repite de aciertos y fallos?
   - 2.1 miss/hit
   - 2.2 miss/hit/hit
   - 2.3 miss/hit/hit/hit
   - 2.4 hit/miss/miss/miss
   - 2.5 hit/miss

Preguntas - Ejercicio

1. ¿Qué ocurre si se altera la cantidad de iteraciones de manera arbitraria?
   - 1.1 La cantidad de cache misses aumenta.
   - 1.2 La cantidad de cache hits aumenta.
   - 1.3 La cantidad de cache misses se queda igual.
   - 1.4 La cantidad de cache hits se queda igual.
   - 1.5 El hit rate aumenta.
   - 1.6 El hit rate disminuye.
   - 1.7 El hit rate se mantiene constante.
2. ¿Por qué le ocurre, lo indicado en la pregunta anterior, al hit rate?
   - 2.1 Porque se aprovecha la localidad espacial.
   - 2.2 Porque se aprovecha la localidad temporal.
   - 2.3 Porque no se aprovecha la localidad espacial.
   - 2.4 Porque no se aprovecha la localidad temporal.
   - 2.5 Porque la cantidad de repeticiones no tiene impacto en el hit rate.