HOJA DE EJERCICIOS

PRÁCTICA 7: JERARQUÍA DE MEMORIA III

DINÁMICA DE LA SESIÓN:

En esta sesión vamos a realizar un estudio sobre el comportamiento de diferentes memorias caché ante una determinada secuencia de accesos a memoria. Para ello dividiremos la práctica en 2 partes:

- Parte 1 – Ejercicios a realizar a mano: el alumno practica la estimación de fallos y la detección de los tipos de fallos que se produce en una memoria caché.
- Parte 2 – Ejercicios a realizar con simulador: mediante el simulador se estudia cómo afecta los distintos parámetros de una caché sobre el rendimiento a partir de una traza dada.

PARTE 1: Vamos a realizar un ejercicio SIN SIMULADOR, suponiendo un computador con direcciones de 16 bits que posee cachés separadas de 256 Bytes de Mapeado Directo CB-WA con tamaño de bloque de 16 Bytes, que ejecuta el siguiente programa:

```
unsigned char x[256] = {…}, y[256] = {…};  // un "char" ocupa 1 Byte en memoria
int i;
for (i=0; i< 256; i=i+8)      //valor de "zancada" = 8 (incremento del índice)
      y[i] = x[i] + 5;
```

a) Represente la decodificación de la dirección que realiza el controlador de caché:

ETIQUETA	*ÍNDICE*	*DESPLAZAMIENTO*
_____ BITS	_____ BITS	_____ BITS

b) Resolvemos los accesos más representativos completando los espacios en blanco en la siguiente traza. A medida que resuelve los accesos, **represente (en la siguiente página) de forma simplificada** cómo va quedando la memoria caché y la memoria principal. Aspectos a tener en cuenta:

- Solo se tendrán en cuenta los **accesos a los vectores**; así pues, **usará solo la caché de datos**.
- El vector ***x*** está cargado a partir de la **dirección 0x0000** de memoria.
- El vector ***y*** está cargado a partir de la **dirección 0x0100** de memoria.
- El índice y el desplazamiento son múltiplos de 4 (resolución rápida trabajando en hexadecimal).

IMPORTANTE: Aunque puede realizar la traza completa para practicar (64 accesos en total), podrá encontrar un patrón sin necesidad de ello. No obstante, al menos se recomienda desarrollar:

- ***Las 4 primeras iteraciones (i = 0, 8, 16, 24).***
- ***Las 4 últimas iteraciones (i = 224, 232, 240, 248).***

Op	Elemento	Dirección	Nº de Bloque	Etiqueta	Índice	Despl.	Acierto/Tipo Fallo
L	x[0]	0x0000	0x000	0x00	0x0	0x0	Fallo Forzoso
E	y[0]	0x0100					
L	x[8]	0x0008					
E	y[8]	0x0108					
L	x[16]	0x0010					
E	y[16]						
L	x[24]						
E	y[24]						
…							
L	x[224]						
E	y[224]						
L	x[232]						
E	y[232]						
L	x[240]						
E	y[240]						
L	x[248]						
E	y[248]						

c) Tras mostrar el estado de ambas memorias durante la resolución de los accesos del apartado anterior, interprete el comportamiento de la **caché de datos** y calcule los valores estadísticos que se obtienen tras la ejecución completa del código detallando los cálculos realizados.

Mem. Principal (contenido)

B0	x[0] – x[15]
B4095	

Memoria Caché

	V	D	Et.	Bloque de datos
L0	1	0	0	B0
L1				
L2				
L3				
L12				
L13				
L14				
L15				

Cálculo:

Nº de Accesos:

Nº de fallos forzosos:

Nº de fallos capacidad:

Nº de fallos conflicto:

Frec. Fallos:

PARTE 2: Ejercicios con simulador

Ejercicio 1: Compruebe el resultado del ejercicio anterior con el simulador. Para ello simule el siguiente código MIPS el cual realiza la misma secuencia de accesos a la caché de datos que el programa C anterior.

Recomendamos que configure la interfaz gráfica del simulador (versión **0.10.5.330** o superior) como a continuación se indica para una mejor interpretación de los accesos: pulse Ctrl+3 (Vista de 'Acceso a memoria'), cierre la ventana 'Memoria Caché de instrucciones', filtre por 'Acceso a datos en memoria' en la ventana 'Traza a datos en memoria' y pulse sobre el botón de estadísticas de la ventana 'Memoria Caché de datos'.

Para ver el comportamiento de la caché acceso a acceso utilice F8, o pulse F9 para simular hasta el final.

```
.eqv    LONG_VECTOR   256
.eqv    ZANCADA       8
.eqv    TAM_CACHE     256
.eqv    TAM_BLOQUE    16
.eqv    NUM_VIAS      1
.eqv    P_ESCRIT      cbwa   ;cbwa   o   wtnwa
.eqv    P_REEMPL      lru    ;random, lru, fifo, lfu
.config
        readprotect off
        split on               ; Cachés separadas
        dcache TAM_CACHE TAM_BLOQUE NUM_VIAS P_ESCRIT P_REEMPL ; Config Caché de datos.
.data
v:      .space LONG_VECTOR
.text
        li $5,0
        li $7,LONG_VECTOR
i:
        lb   $6, 0($5)           ; Lee x[i]
        addi $6,$6,5             ; Suma 5
        sb   $6, 256($5)         ; Escribe en y[i]
        addi $5,$5,ZANCADA       ; Incrementa el índice
        blt $5,$7,i              ; ¿Fin Vector?
```

Ejercicio 2: A continuación, vamos a realizar un estudio de cómo afectan al rendimiento algunos parámetros de la caché utilizando como programa test el código MIPS anterior (se empleará las políticas CB-WA y LRU). Rellene los casos de la siguiente tabla según se le solicite en posteriores apartados. Para modificar la caché del simulador simplemente modifique el valor de las constantes (.eqv) declaradas en el código anterior.

Caso	Zancada	Memoria Caché	Nº Forzosos	Nº Conflicto	Nº Capacidad	Nº Accesos
1	8	256 Bytes, Bloque 16, 1 vía	32	30	2	64
2	8	256 Bytes, Bloque 32, 1 vía				
3	8	512 Bytes, Bloque 16, 1 vía				
4	8	256 Bytes, Bloque 16, 2 vías				
5	16	256 Bytes, Bloque 16, 1 vía				
6	16	256 Bytes, Bloque 32, 1 vía				
7	16	512 Bytes, Bloque 16, 1 vía				
8	16	256 Bytes, Bloque 16, 2 vías				
9	32	256 Bytes, Bloque 16, 1 vía				
10	32	256 Bytes, Bloque 32, 1 vía				
11	32	512 Bytes, Bloque 16, 1 vía				
12	32	256 Bytes, Bloque 16, 2 vías				

Intente responder a las siguientes preguntas a partir de los datos obtenidos y las justificaciones estudiadas en clase de teoría:

a) ¿Qué sucede si aumento el tamaño de bloque?
b) ¿Qué sucede si aumento la zancada?
c) ¿Qué sucede si aumento el tamaño de la caché?
d) ¿Qué sucede si aumento la asociatividad?
e) ¿Variaría algo si la política de escritura fuera WT-NWA?
f) ¿Variaría algo si cambiamos la política de reemplazo?