EJERCICIOS - ENUNCIADOS

ARQUITECTURA DE COMPUTADORES (GII-AC)

BLOQUE 2: DISEÑO DE LA JERARQUÍA DE MEMORIA

Tema 2-1: EJERCICIOS DE LA MEMORIA CACHE

Ejercicio 1 (11/11/16)

Suponiendo que se ejecuta el siguiente programa en un computador basado en Nios II/f

```
.org 0x0
_start: movia r2, A
    movia r3, B
    movia r4, N
    ldw r4, 0(r4)
    add r5, r0, r0
LOOP:   ldw r6, 0(r2)
    ldw r7, 0(r3)
    mul r8, r6, r7
    add r5, r5, r8
    addi r2, r2, 4
    addi r3, r3, 4
    subi r4, r4, 1
    bgt r4, r0, LOOP
    stw r5, D(r0)
STOP:   br STOP

.org 0x1000
N:   .word 20
.org 0x1100
A:   .word 1,2,…,20
 .org 0x1200
B:   .word 20,19,…,1
.org 0x1300
D:   .skip 4
```

y que el procesador dispone de una memoria cache de **instrucciones** de correspondencia directa y con 2 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits. Se solicita:

(a, 40%) obtener el índice de la memoria cache al que se asigna cada instrucción del programa explicando cómo se calcula,

SOLUCIÓN

¿Cómo se calcula el índice?	ÍNDICE	DIRECCIÓN DE MEMORIA	Programa
Número de conjuntos= 2 Número de bits de índice= 1			.org 0x0000
	0	0x00000000	_start: movia r2, A
	0	0x00000004	movia r3, B
	1	0x00000008	movia r4, N
	1	0x0000000C	ldw r4, 0(r4)

Bytes por bloque= 8 bytes Número bits de selección del byte= 3 Orden del bit de índice en la dirección= 3 (comenzando a contar desde 0)	0	0x00000000	add r5, r0, r0
	0	0x00000014	LOOP: ldw r6, 0(r2)
	1	0x00000018	ldw r7, 0(r3)
	1	0x0000001C	mul r8, r6, r7
	0	0x00000020	add r5, r5, r8
	0	0x00000024	addi r2, r2, 4
	1	0x00000028	addi r3, r3, 4
	1	0x0000002C	subi r4, r4, 1
	0	0x00000030	bgt r4, r0, LOOP
	0	0x00000034	stw r5, D(r0)
	1	0x00000038	STOP: br STOP
			.org 0x1000
			N: .word 20
			.org 0x1100
			A: .word 1,2,…,20
			.org 0x1200
			B: .word 20,19,…,1
			.org 0x1300
			D: .skip 4

(b, 60%) calcular la frecuencia de fallos en la memoria cache de instrucciones explicando cómo se ha realizado el cálculo.

SOLUCIÓN

Índice	Palabra 0 (4 bytes)	Palabra 1 (4 bytes)
0	movia r2, A	movia r3, B
1	movia r4, N	ldw r4, 0(r4)

Número de accesos a la memoria cache de instrucciones: 5 (instrucciones previas a LOOP) + 8 inst/iter x 20 iteraciones + 2 (final programa) = 167 accesos

Fallos en cache de instrucciones: 3 (instrucciones previas a LOOP) + 4 (primera iteración) + 5 fallos/iter x 19 iter + 1 (br) = 7 + 95 + 1 = 103

Frecuencia de fallos = 100% x 103 / 167 = 61%

Ejercicio 2 (1/12/2015)

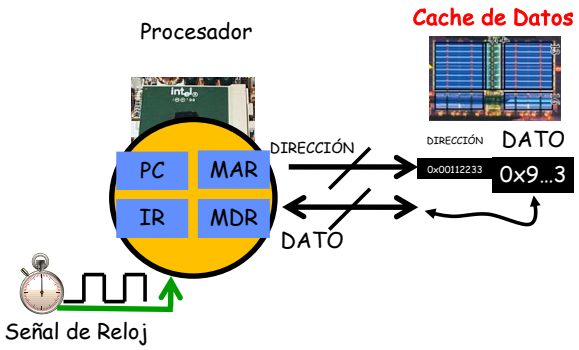
Suponiendo que se ejecuta el siguiente programa en un computador basado en Nios II/f

```
.org 0x0000
_start: movia r2, A
        movia r3, B
        movia r4, N
        ldw r4, 0(r4)
        add r5, r0, r0
LOOP:   ldw r6, 0(r2)
        ldw r7, 0(r3)
        mul r8, r6, r7
        add r5, r5, r8
        addi r2, r2, 4
        addi r3, r3, 4
        subi r4, r4, 1
        bgt r4, r0, LOOP
        stw r5, D(r0)
STOP:   br STOP

.org 0x1000
N:   .word 20
.org 0x1100
A:   .word 1,2,…,20
 .org 0x1200
B:   .word 20,19,…,1
.org 0x1300
D:   .skip 4
```

y que el procesador dispone de una memoria cache de datos, de correspondencia directa y con 8 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits. Se solicita: (a, 20%) obtener el índice de la memoria cache al que se asigna cada palabra de datos del programa explicando cómo se calcula, (b, 60%) calcular la frecuencia de fallos en la memoria cache explicando cómo se ha realizado el cálculo, (c, 20%) calcular el tiempo de acceso promedio a la jerarquía de memoria si los accesos que aciertan en la memoria cache tardan 1 ciclo de reloj y los accesos a la memoria principal tardan 50 ciclos de reloj.

SOLUCIÓN

Organización de la memoria cache de correspondencia directa y con 8 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits (palabra 0, palabra 1). Los 8 bloques se distribuyen en 4 índices.



(a) En la siguiente tabla se puede observar una representación de los bloques de datos de la memoria cache de datos. Observa que existen **8 líneas de datos**, cada una de ellas incluye un bloque de 2 palabras y una etiqueta. Cada línea se identifica por un valor de Índice distinto, de 0 a 7.

REPRESENTACIÓN DE LA MICROARQUITECTURA DE LA MEMORIA CACHE DE DATOS

	Índice	Palabra 0 (4 bytes)	Palabra 1 (4 bytes)	Etiqueta (26 bits)
Una línea de cache →	0			
	1			
	2			
	3			
	4			
Otra línea de cache →	5			
	6			
	7			

El programa del enunciado accede a todas las componentes de los vectores A y B: A[0], A[1], …, A[19], B[0], B[1], …, B[19], además de las variables N y D. En total, se realizan 42 accesos a la memoria cache de datos.

En la siguiente tabla se incluyen las direcciones de todas las variables que el programa usa y los correspondientes índices. El **índice** se obtiene a partir de la dirección de los datos emitida por el procesador, extrayendo los 2 bits que se encuentran a continuación de los bits del campo de la dirección destinados a la **selección del byte**. El campo de la dirección destinado a la selección del byte consta de 3 bits porque un bloque de la memoria cache de datos contiene 8 bytes: **3 bits = $\log_2$ (8 índices)**.

En la siguiente tabla se representan los distintos **campos de la dirección de memoria** de 32 bits emitida por el procesador cuando ejecuta una instrucción de acceso a memoria.

DIRECCIÓN DE MEMORIA EMITIDA POR EL PROCESADOR, 32 bits

Etiqueta (26 bits)	Índice (3 bits)			Selección del byte dentro del bloque (3 bits)
31 30 … 7 6	5	4	3	2 1 0

En la siguiente tabla que se denomina **Tabla de índices** se representan los índices de la memoria cache de datos que se asignan a las distintas direcciones de memoria emitidas por el procesador cuando ejecuta una instrucción de acceso a memoria.

TABLA DE ÍNDICES DE LAS DIRECCIONES DE MEMORIA

Índice	Direcciones de memoria	0…3	4…7	Índice	Direcciones de memoria	8…B	C…F
0	0x1000	N					
0	0x1100	A[0]	A[1]	1	0x1108	A[2]	A[3]
2	0x1110	A[4]	A[5]	3	0x1118	A[6]	A[7]
4	0x1120	A[8]	A[9]	5	0x1128	A[10]	A[11]
6	0x1130	A[12]	A[13]	7	0x1138	A[14]	A[15]
0	0x1140	A[16]	A[17]	1	0x1148	A[18]	A[19]
0	0x1200	B[0]	B[1]	1	0x1208	B[2]	B[3]
2	0x1210	B[4]	B[5]	3	0x1218	B[6]	B[7]
4	0x1220	B[8]	B[9]	5	0x1228	B[10]	B[11]
6	0x1230	B[12]	B[13]	7	0x1238	B[14]	B[15]
0	0x1240	B[16]	B[17]	1	0x1248	B[18]	B[19]
0	0x1300	D					

En la siguiente tabla se puede observar una representación de los bloques de datos de la memoria cache de datos después de ejecutarse 6 iteraciones del bucle del programa. Observa que en cada línea de la cache se han asignado varias direcciones distintas de memoria. Para distinguir los datos que se encuentran en un momento determinado en una línea de memoria cache es necesario contrastar la etiqueta que se encuentra en ese momento en la correspondiente etiqueta de la línea con la etiqueta extraída de la dirección emitida por el procesador.

CONTENIDO DE LA MEMORIA CACHE DESPUÉS DE EJECUTARSE 6 ITERACIONES

Índice	Palabra 0 (4 bytes)	Palabra 1 (4 bytes)	Etiqueta (26 bits)
0	N → A[0] → B[0] →…	# → A[1] → B[1] →…	(N) 0b0001 0000 00 (A) 0b0001 0001 00 (B) 0b0001 0010 00
1	A[2] → B[2] →…	A[3] → B[3] →…	(A) 0b0001 0001 00 (B) 0b0001 0010 00
2	A[4] → B[4] →…	A[5] → B[5] →…	(A) 0b0001 0001 00 (B) 0b0001 0010 00
3			
4			
5			
6			
7			

Cuando se accede a la cache y se falla por las etiquetas de la dirección emitida por el procesador y la. Guardada en la correspondiente línea de cache, se copia en un bloque de cache dos palabras desde el siguiente nivel de la jerarquía de memoria, la cual suponemos en este ejercicio que coincide con la memoria principal. Este bloque que se copia en la cache contiene la dirección de memoria de la palabra que originó el fallo y la dirección de la palabra que se almacena a continuación en memoria principal. Sin embargo, como en cada iteración del programa se accede a componentes con el mismo orden de las matrices A y B (A[0]→B[0], A[1]→B[1], etc.), y ambas componentes de A y B que se acceden en una misma iteración coinciden en índice, esto hace que se produzcan 42 fallos en cache: 20 debido a los accesos de todos componentes de A, 20 debido a los accesos de todos las componentes de B, además de los 2 accesos a las variables N y D.

(b) Por lo tanto, la frecuencia de fallos que nos pide calcular el enunciado es la siguiente habiendo obtenido previamente 42 fallos y 42 accesos a memoria:

Frecuencia de fallos (FF) = 100% x 42 / 42 = 100%

Tasa de aciertos = 1 – Tasa de fallos = 0

(c) Y el tiempo de acceso promedio a la memoria tiene como valor el siguiente resultado habiendo obtenido una frecuencia de fallos en tanto por 1 de 1,0 y usando los datos del enunciado $t_{acierto}$ =1,0 ciclo y P=50,0 ciclos:

AMAT = $t_{acierto}$ + FF x P = 1 + 1 x 50 = 51 ciclos

Ejercicio 3 (17/1/17)

Suponiendo que se ejecuta el siguiente programa en un computador basado en Nios II/f

```
.org 0x1100                          subi r4, r4, 1

_start:movia r2, A                   bgt  r4, r0, LOOP

   movia r3, B                       stw  r5, D(r0)

   movia r4, N                       STOP:br STOP

   ldw   r4, 0(r4)

   add   r5, r0, r0                  .org 0x2000

LOOP:  ldw   r6, 0(r2)               N:   .word 35

   ldw   r7, 0(r3)                   .org 0x2100

   mul   r8, r6, r7                  B:   .word 1,2,…,35

   mul   r8, r8, r8                   .org 0x2200

   add   r5, r5, r8                  A:   .word 35,34,…,1

   addi  r2, r2, 4                   .org 0x2300

   addi  r3, r3, 4                   D:   .skip 4
```

y que el procesador dispone de una <u>memoria cache de instrucciones</u> asociativa por conjuntos de 2 vías y con <u>4 bloques</u> de datos, donde cada bloque de datos puede alojar <u>2 palabras de 32 bits</u>. Se solicita:

(a, 20%) **obtener el índice de la memoria cache al que se asigna cada palabra que se encuentra en la zona de instrucciones del programa explicando cómo se calcula**

(b, 30%) **calcular el número de fallos en la memoria cache de instrucciones**

(c, 30%) **calcular el número de aciertos en la memoria cache de instrucciones**

(d, 10%) **calcular la frecuencia de fallos en la memoria cache de instrucciones explicando cómo se ha realizado el cálculo**

(e, 10%) **calcular el tiempo de acceso promedio de los accesos a memoria del programa, suponiendo un tiempo de acierto en la cache de 1 ciclo y un tiempo de penalización de 50 ciclos**

SOLUCIÓN

(a) Selección del byte: 3 bits ($\log_2 8$), índice: 1 bit ($\log_2 2$)

ÍNDICE (4° bit de la dirección)	DIRECCIÓN DE MEMORIA	Programa
		.org 0x1100
0	0x1100	_start: movia r2, A
0	0x1104	movia r3, B

1	0x1108	movia r4, N
1	0x110C	ldw r4, 0(r4)
0	0x1110	add r5, r0, r0
0	0x1114	LOOP: ldw r6, 0(r2)
1	0x1118	ldw r7, 0(r3)
1	0x111C	mul r8, r6, r7
0	0x1120	mul r8, r8, r8
0	0x1124	add r5, r5, r8
1	0x1128	addi r2, r2, 4
1	0x112C	addi r3, r3, 4
0	0x1130	subi r4, r4, 1
0	0x1134	bgt r4, r0, LOOP
1	0x1138	stw r5, D(r0)
1	0x113C	STOP: br STOP
		.org 0x2000
		N: .word 35
		.org 0x2100
		B: .word 1,2,…,35
		.org 0x2200
		A: .word 35,34,…,1
		.org 0x2300
		D: .skip 4
		.end

Organización interna de la memoria cache de instrucciones asociativa por conjuntos de 2 vías

		Índice		
Palabra-0	Palabra-1	0	Palabra-0	Palabra-1
Palabra-0	Palabra-1	1	Palabra-0	Palabra-1

(b, c)

Ayuda: Se propone rellenar esta tabla

	Parte del programa	accesos	aciertos	fallos
	Preámbulo	5	2	3
	Iteración-1	9	5	4
	Iteración-2..35	9	6	3
	Prólogo	2	1	1
	Total	5+9+34x9+2=322	2+5+34x6+1=212	3+4+34x3+1=110

(d) Frecuencia de fallos= 100% x 110/322 = 34,16%

(e) AMAT = 1 + 0,3416 x 50 = 18,08 ciclos

Ejercicio 4 (23/6/17)

Suponiendo que se ejecuta el siguiente programa en un computador basado en Nios II/f:

```
.org 0x1100
_start: movia r2, A
        movia r3, B
        movia r4, N
        ldw   r4, 0(r4)
        add   r5, r0, r0
LOOP:   ldw   r6, 0(r2)
        ldw   r7, 0(r3)
        mul   r8, r6, r7
        mul   r8, r8, r8
        add   r5, r5, r8
        addi  r2, r2, 4
        addi  r3, r3, 4
        subi  r4, r4, 1
        bgt   r4, r0, LOOP
        stw   r5, D(r0)
STOP:br STOP

.org 0x2000
N:   .word 35
.org 0x2100
B:   .word 1,2,…,35
 .org 0x2200
A:   .word 35,34,…,1
.org 0x2300
D:   .skip 4
```

y que el procesador dispone de una <u>memoria cache de datos</u> asociativa por conjuntos de 2 vías y con <u>4 bloques</u> de datos, donde cada bloque de datos puede alojar <u>2 palabras de 32 bits</u>, se solicita:

(a, 20%) **obtener el índice de la memoria cache al que se asigna cada palabra que se encuentra en la zona de datos del programa explicando cómo se calcula**

(b, 30%) **calcular el número de fallos en la memoria cache de datos**

(c, 30%) **calcular el número de aciertos en la memoria cache de datos**

(d, 10%) **calcular la frecuencia de fallos en la memoria cache de datos explicando cómo se ha realizado el cálculo**

(e, 10%) **calcular el tiempo de acceso promedio de los accesos a memoria del programa, suponiendo un tiempo de acierto en la cache de 1 ciclo y un tiempo de penalización de 50 ciclos**

SOLUCIÓN

Organización de la memoria cache de 2 vías y con 4 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits (palabra 0, palabra 1). Los 4 bloques se distribuyen en 2 índices.

Índice (1 bit)	Vía 0		Vía 1	
	Palabra 0	**Palabra 1**	**Palabra 0**	**Palabra 1**
0	N, B[0]	B[1]	A[0]	A[1]
1	A[2]	A[3]	B[2]	B[3]

Dirección de memoria (32 bits) emitida por el procesador:

Etiqueta (26 bits)	**Índice (1 bit)**	**Selección del byte dentro del bloque (3 bits)**
31 30 … 7 6 5 4	3	2 1 0

El programa accede a todas componentes A[0],A[1],…,A[34],B[0],B[1],…,B[34], además de N y D; en total 72 accesos (35 x 2 + 2). En la siguiente tabla se incluyen las direcciones de todas las variables que el programa usa y los correspondientes índices. El índice se obtiene a partir de la dirección de los datos extrayendo el bit que se encuentra a continuación de los bits del campo de la dirección destinados a la selección del byte (3 bits porque el bloque consta de 8 bytes).

Índice	**Direcciones de memoria**	**0…3**	**4…7**	**Índice**	**Direcciones de memoria**	**8…B**	**C…F**
0	0x2000	N	#				
0	0x2100	B[0]	B[1]	1	0x2108	B[2]	B[3]
0	0x2110	B[4]	B[5]	1	0x2118	B[6]	B[7]
0	0x2120	B[8]	B[9]	1	0x2128	B[10]	B[11]
0	0x2130	B[12]	B[13]	1	0x2138	B[14]	B[15]
0	0x2140	B[16]	B[17]	1	0x2148	B[18]	B[19]
0	0x2150	B[20]	B[21]	1	0x2158	B[22]	B[23]
0	0x2160	B[24]	B[25]	1	0x2168	B[26]	B[27]
0	0x2170	B[28]	B[29]	1	0x2178	B[30]	B[31]
0	0x2180	B[32]	B[33]	1	0x2188	B[34]	
0	0x2200	A[0]	A[1]	1	0x2208	A[2]	A[3]
0	0x2210	A[4]	A[5]	1	0x2218	A[6]	A[7]
0	0x2220	A[8]	A[9]	1	0x2228	A[10]	A[11]
0	0x2230	A[12]	A[13]	1	0x2238	A[14]	A[15]
0	0x2240	A[16]	A[17]	1	0x2248	A[18]	A[19]
0	0x2250	A[20]	A[21]	1	0x2258	A[22]	A[23]
0	0x2260	A[24]	A[25]	1	0x2268	A[26]	A[27]
0	0x2270	A[28]	A[29]	1	0x2278	A[30]	A[31]
0	0x2280	A[32]	A[33]	1	0x2288	A[34]	
0	0x2300	D					

Índice	Vía 0		Vía 1	
	0	1	0	1
0	N, B[0]	#, B[1]	A[0]	A[1]
1	A[2]	A[3]	B[2]	B[3]

Cuando se accede a la cache y se falla se inicializa un bloque de cache con dos palabras: la que originó el fallo y la que se almacena a continuación en memoria. Esto hace que se produzcan 38 fallos en cache (marcados en azul): 18 fallos debido a los accesos de los índices pares de A, 18 fallos debido a los accesos de los índices pares de B, además de los accesos a las variables N y D.

Número de fallos = 38

Tasa de fallos (FF) = 100% x 38 / 72 = 53%

Número de aciertos = 72 - 38 = 34

Tasa de aciertos = 1 – 38/72 = 0,48 (47 %)

AMAT = 1 + 0,53 x 50 = 26 ciclos

Ejercicio derivado: ¿Qué ocurre con la frecuencia de fallos y el AMAT si N = 25?

SOLUCIÓN

Número de accesos = 52

Número de fallos = 13 + 13 + 2 = 28

Tasa de fallos (FF) = 100% x 28 / 52 = 54%

Número de aciertos = 52 - 28 = 24

Tasa de aciertos = 1 – 28/52 = 0,48 (46 %)

AMAT = 1 + 0,54 x 50 = 27 ciclos

Ejercicio 5

Suponiendo que se ejecuta el siguiente programa en un computador basado en Nios II/f

```
        .org 0x0000
_start:  movia r2, A
         movia r3, B
         movia r4, N
         ldw r4, 0(r4)
         add r5, r0, r0
LOOP:    ldw r6, 0(r2)
         ldw r7, 0(r3)
         mul r8, r6, r7
         add r5, r5, r8
         addi r2, r2, 4
         addi r3, r3, 4
         subi r4, r4, 1
         bgt r4, r0, LOOP
         stw r5, D(r0)
STOP:    br STOP

        .org 0x1000
N:      .word 20 /* número de elementos*/
A:      .word 1,2,…,20 /* 20 elementos del vector A */
B:      .word 20,19,…,1 /* 20 elementos del vector B */
D:      .skip 4 /* resultado del producto escalar */
```

Y que el procesador dispone de una memoria cache de instrucciones y una cache de datos, ambas de correspondencia directa y con solo 4 bloques, calcular:

- La frecuencia o tasa de fallos de la memoria cache de instrucciones donde cada bloque puede almacenar 4 palabras de 32 bits
- La frecuencia o tasa de fallos de la memoria cache de instrucciones donde cada bloque puede almacenar 2 palabras de 32 bits
- La tasa de fallos de la memoria cache de datos donde cada bloque puede almacenar 4 palabras de 32 bits
- Si no supieras que la memoria cache de datos tiene un bloque de datos de una capacidad de 16 bytes, ¿cómo podrías saber ese tamaño de bloque?
- En todos los casos, el tiempo de acceso promedio a la jerarquía de memoria si los accesos que aciertan en ambas memorias cache tardan 1 ciclo de reloj y los accesos a la memoria principal tardan 50 ciclos de reloj

SOLUCIÓN

Y que el procesador dispone de una memoria cache de instrucciones y una cache de datos, ambas de correspondencia directa y con solo 4 bloques, calcular:

- La frecuencia o tasa de fallos de la memoria cache de instrucciones donde cada bloque puede almacenar 4 palabras de 32 bits

DIRECCIONES iniciales palabra	.org 0x0000
0x0000	_start: movia r2, A
0x0004	movia r3, B
0x0008	movia r4, N

0x000C		ldw r4, 0(r4)
0x0010		add r5, r0, r0
0x0014	LOOP:	ldw r6, 0(r2)
0x0018		ldw r7, 0(r3)
0x001C		mul r8, r6, r7
0x0020		add r5, r5, r8
0x0024		addi r2, r2, 4
0x0028		addi r3, r3, 4
0x002C		subi r4, r4, 1
0x0030		bgt r4, r0, LOOP
0x0034		stw r5, D(r0)
0x0038	STOP:	br STOP

  - Número de accesos a la memoria cache de instrucciones = 5 (entrada) + 20 iteraciones x 8 instrucciones/iteración + 2 (salida) = 167
  - Fallos en la cache de instrucciones = 4 (fallos en frío por los accesos a direcciones 0x0, 0x10, 0x20, 0x30)
  - Tasa de fallos = 100% x 4 / 167 = 2,4%
- La frecuencia o tasa de fallos de la memoria cache de instrucciones donde cada bloque puede almacenar 2 palabras de 32 bits

DIRECCIONES iniciales palabra	.org 0x0000	
0x0000	_start:	movia r2, A
0x0004		movia r3, B
0x0008		movia r4, N
0x000C		ldw r4, 0(r4)
0x0010		add r5, r0, r0
0x0014	LOOP:	ldw r6, 0(r2)
0x0018		ldw r7, 0(r3)
0x001C		mul r8, r6, r7
0x0020		add r5, r5, r8
0x0024		addi r2, r2, 4
0x0028		addi r3, r3, 4
0x002C		subi r4, r4, 1
0x0030		bgt r4, r0, LOOP
0x0034		stw r5, D(r0)
0x0038	STOP:	br STOP

  - Número de accesos a la memoria cache de instrucciones = 5 (entrada) + 20 iteraciones x 8 instrucciones/iteración + 2 (salida) = 167
  - Fallos en la cache de instrucciones = 3 (entrada, fallos en frío por los accesos a direcciones 0x0, 0x8, 0x10) + 4 (1ª iteración, fallos en frío por los accesos a direcciones 0x18, 0x20, 0x28, 0x30) + 2 fallos (accesos a 0x14 y 0x30 que coinciden en bloque 2 de la cache)/iteración x 19 iteraciones + 1 (salida, 0x38) = 46
  - Tasa de fallos = 100% x 46 / 167 = 27,5%
- La tasa de fallos de la memoria cache de datos donde cada bloque puede almacenar 4 palabras de 32 bits

Índice	Direcciones	0…3	4…7	8…B	C…F
0	0x1000	N	A[0]	A[1]	A[2]
1	0x1010	A[3]	A[4]	A[5]	A[6]
2	0x1020	A[7]	A[8]	A[9]	A[10]

3	0x1030	A[11]	A[12]	A[13]	A[14]
0	0x1040	A[15]	A[16]	A[17]	A[18]
1	0x1050	A[19]	B[0]	B[1]	B[2]
2	0x1060	B[3]	B[4]	B[5]	B[6]
3	0x1070	B[7]	B[8]	B[9]	B[10]
0	0x1080	B[11]	B[12]	B[13]	B[14]
1	0x1090	B[15]	B[16]	B[17]	B[18]
2	0x10A0	B[19]	D		

    - Número de accesos a la memoria cache de datos = 1 (entrada, N) + 20 iteraciones x 2 instrucciones (A[], B[])/iteración + 1 (salida, D) = 42
    - Fallos en la cache de datos = 1 (entrada, fallos en frío por los accesos a dirección 0x1000, N) + 1 (1ª iteración, fallos en frío por los accesos a direcciones 0x1054, B[0]) + 2 fallos (4ª iteración, accesos a 0x1010, A[3], índice= 1, y 0x1060, B[3], índice= 2) + 2 fallos (8ª iteración, accesos a 0x1020, A[7], índice= 2, y 0x1070, B[7], índice= 3) + 2 fallos (12ª iteración) + 2 fallos (16ª iteración) + 2 fallos (20ª iteración) = 1 + 1 + 2 x 5 = 12
    - Tasa de fallos = 100% x 12 / 42 = 28,6%
- Si no supieras que la memoria cache de datos tiene un bloque de datos de una capacidad de 16 bytes, ¿cómo podrías saber ese tamaño de bloque?
    - Cada 4 iteraciones se producen se acceden a 8 palabras distintas (A, B) y se producen 2 fallos. Como las palabras de A y B no se solapan en el mismo bloque de cache, los aciertos en tres iteraciones es debido a la localidad espacial. Cada fallo en una dirección de una componente A o B tiene luego 3 aciertos. 1 fallo y 3 aciertos consecutivos corresponden a 4 palabras: 16 bytes.
- En todos los casos, el tiempo de acceso promedio a la jerarquía de memoria si los accesos que aciertan en ambas memorias cache tardan 1 ciclo de reloj y los accesos a la memoria principal tardan 50 ciclos de reloj
    - (1 cache instrucciones 4 palabras) AMAT = 1 + 0,024 x 50 = 2,2 ciclos
    - (2 cache instrucciones 2 palabras) AMAT = 1 + 0,275 x 50 = 14,75 ciclos
    - (3 cache datos 4 palabras) AMAT = 1 + 0,286 x 50 = 15,3 ciclos

Ejercicio 6

Suponiendo la situación del ejercicio anterior (memoria cache de instrucciones y memoria cache de datos, ambas de correspondencia directa y con solo 4 bloques), si se cambia el programa de las dos formas siguientes (se resaltan en rojo las instrucciones o directivas modificadas),

Programa 1	Programa 2

.org 0x0000
_start: movia r2, A
movia r3, B
movia r4, N
ldw r4, 0(r4)
add r5, r0, r0
LOOP: ldw r6, 0(r2)
ldw r7, 0(r3)
mul r8, r6, r7
add r5, r5, r8
addi r2, r2, 8
addi r3, r3, 8
subi r4, r4, 2
bgt r4, r0, LOOP
stw r5, D(r0)
STOP: br STOP
 	
.org 0x1000
N: .word 20
A: .word 1,2,…,20
B: .word 20,19,…,1
D: .skip 4
	.org 0x0000
	_start: movia r2, A
	movia r3, B
	movia r4, N
	ldw r4, 0(r4)
	add r5, r0, r0
	LOOP: ldw r6, 0(r2)
	ldw r7, 0(r3)
	mul r8, r6, r7
	add r5, r5, r8
	addi r2, r2, 4
	addi r3, r3, 4
	subi r4, r4, 1
	bgt r4, r0, LOOP
	stw r5, D(r0)
	STOP: br STOP
	
	.org 0x1000
	N: .word 20
	A: .word 1,2,…,20
	.org 0x2000
	B: .word 20,19,…,1
	D: .skip 4

Calcular para cada uno de los dos programas:

- La tasa de fallos de la memoria cache de datos donde cada bloque puede almacenar 4 palabras de 32 bits
- El tiempo de acceso promedio a la jerarquía de memoria si los accesos que aciertan en la memoria cache de datos tardan 1 ciclo de reloj y los accesos a la memoria principal tardan 50 ciclos de reloj
- Suponer ahora que la memoria cache de datos pasa a ser asociativa por conjuntos de 2 vías, manteniendo la misma capacidad de 64 bytes y tamaño de bloque de 4 palabras de 32 bits (16 bytes). ¿Cuál sería la frecuencia de fallos y el tiempo de acceso promedio de los accesos a la memoria cache de datos cuando se ejecuta cada uno de los dos programas de este ejercicio?
- Suponer ahora que la memoria cache de datos es asociativa por conjuntos de 2 vías, y con 8 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits. Calcular: la frecuencia de fallos en la memoria cache, el tiempo de acceso promedio a la jerarquía de memoria si los accesos que aciertan en la memoria cache tardan 1 ciclo de reloj y los accesos a la memoria principal tardan 50 ciclos de reloj.

SOLUCIÓN

Programa 1	Programa 2

.org 0x0000
_start: movia r2, A
movia r3, B
movia r4, N
ldw r4, 0(r4)
add r5, r0, r0
LOOP: ldw r6, 0(r2)
ldw r7, 0(r3)
mul r8, r6, r7
add r5, r5, r8
addi r2, r2, 8
addi r3, r3, 8
subi r4, r4, 2
bgt r4, r0, LOOP
stw r5, D(r0)
STOP: br STOP
 	
.org 0x1000
N: .word 20
A: .word 1,2,…,20
B: .word 20,19,…,1
D: .skip 4
	.org 0x0000
	_start: movia r2, A
	movia r3, B
	movia r4, N
	ldw r4, 0(r4)
	add r5, r0, r0
	LOOP: ldw r6, 0(r2)
	ldw r7, 0(r3)
	mul r8, r6, r7
	add r5, r5, r8
	addi r2, r2, 4
	addi r3, r3, 4
	subi r4, r4, 1
	bgt r4, r0, LOOP
	stw r5, D(r0)
	STOP: br STOP
	
	.org 0x1000
	N: .word 20
	A: .word 1,2,…,20
	.org 0x2000
	B: .word 20,19,…,1
	D: .skip 4

- La tasa de fallos de la memoria cache de datos donde cada bloque puede almacenar 4 palabras de 32 bits

  Programa 1: accede a componentes pares A[0],A[2],A[4],…,A[18],B[0],B[2],B[4],…,B[18]

Índice	Direcciones	0…3	4…7	8…B	C…F
0	0x1000	N	A[0]	A[1]	A[2]
1	0x1010	A[3]	A[4]	A[5]	A[6]
2	0x1020	A[7]	A[8]	A[9]	A[10]
3	0x1030	A[11]	A[12]	A[13]	A[14]
0	0x1040	A[15]	A[16]	A[17]	A[18]
1	0x1050	A[19]	B[0]	B[1]	B[2]
2	0x1060	B[3]	B[4]	B[5]	B[6]
3	0x1070	B[7]	B[8]	B[9]	B[10]
0	0x1080	B[11]	B[12]	B[13]	B[14]
1	0x1090	B[15]	B[16]	B[17]	B[18]
2	0x10A0	B[19]	D		

  - Número de accesos a la memoria cache de datos = 1 (entrada, N) + 10 iteraciones x 2 instrucciones (A[], B[])/iteración + 1 (salida, D) = 22
  - Fallos en la cache de datos = 1 (entrada, fallos en frío por los accesos a dirección 0x1000, N) + 1 (1ª iteración, fallos en frío por los accesos a direcciones 0x1054, B[0]) + 2 fallos (3ª iteración, accesos a 0x1014, A[4], índice= 1, y 0x1064, B[4], índice= 2) + 2 fallos (5ª iteración, accesos a 0x1024, A[8], índice= 2, y 0x1074, B[8], índice= 3) + 2 fallos (7ª iteración) + 2 fallos (9ª iteración) + 1 fallo (salida, D) = 1 + 1 + 2 x 4 + 1 = 11

- Tasa de fallos = 100% x 11 / 22 = 50%

Programa 2: accede a todas componentes A[0],A[1],…,A[19],B[0],B[1],…,B[19]

Índice	Direcciones	0…3	4…7	8…B	C…F
0	0x1000	N	A[0]	A[1]	A[2]
1	0x1010	A[3]	A[4]	A[5]	A[6]
2	0x1020	A[7]	A[8]	A[9]	A[10]
3	0x1030	A[11]	A[12]	A[13]	A[14]
0	0x1040	A[15]	A[16]	A[17]	A[18]
1	0x1050	A[19]			
0	0x2000	B[0]	B[1]	B[2]	B[3]
1	0x2010	B[4]	B[5]	B[6]	B[7]
2	0x2020	B[8]	B[9]	B[10]	B[11]
3	0x2030	B[12]	B[13]	B[14]	B[15]
0	0x2040	B[16]	B[17]	B[18]	B[19]
1	0x2050	D			

- Número de accesos a la memoria cache de datos = 1 (entrada, N) + 20 iteraciones x 2 instrucciones (A[], B[])/iteración + 1 (salida, D) = 42
- Fallos en la cache de datos = 1 (entrada, fallos en frío por los accesos a dirección 0x1000, N) + 1 (1ª iteración, fallos en frío por los accesos a dirección 0x2000, B[0], índice=0) + 2 fallos (2ª iteración, fallos por remplazamientos de los accesos a direcciones 0x1008, A[1], y 0x2004, B[1], índice=0) + 2 fallos (3ª iteración, fallos por remplazamientos de los accesos a direcciones 0x100C, A[2], y 0x2008, B[2], índice=0) + 1 fallo (4ª iteración, fallos en frío de los accesos a direcciones 0x1010, A[3], índice=1) + 1 fallo (5ª iteración, fallos en frío de los accesos a direcciones 0x2010, B[4], índice=1) + 2 fallos x 2 iteraciones (6ª,7ª iteración, A[5], B[5], A[6], B[6], índice=1) + 1 fallo (8ª iteración, A[7]) + 1 fallo (9ª iteración, B[8]) + 2 fallos x 2 iteraciones (10ª,11ª iteración, A[9], B[9], A[10], B[10], índice=2) + 1 fallo (12ª iteración, A[11]) + 1 fallo (13ª iteración, B[12]) + 2 fallos x 2 iteraciones (14ª,15ª iteración, A[13], B[13], A[14], B[14], índice=1) + 1 fallo (16ª iteración, A[15]) + 1 fallo (17ª iteración, B[16]) + 2 fallos x 2 iteraciones (18ª,19ª iteración, A[17], B[17], A[18], B[18], índice=1) + 1 fallo (20ª iteración, A[19]) + 1 fallo (salida, D) = 1 + 1 + 2 x 2it + 1 + 1 + 2 x 2it + 1 + 1 + 2 x 2it + 1 + 1 + 2 x 2it + 1 + 1 + 2 x 2it + 1 +1 = 32 (falla en todos los accesos 42 menos en 10 accesos: 32=42-10).
- Tasa de fallos = 100% x 32 / 42 = 76,2%

- El tiempo de acceso promedio a la jerarquía de memoria si los accesos que aciertan en la memoria cache de datos tardan 1 ciclo de reloj y los accesos a la memoria principal tardan 50 ciclos de reloj
  - (Programa 1) AMAT = 1 + 0,50 x 50 = 26 ciclos
  - (Programa 2) AMAT = 1 + 0,762 x 50 = 39,1 ciclos

- Suponer ahora que la memoria cache de datos pasa a ser asociativa por conjuntos de 2 vías, manteniendo la misma capacidad de 64 bytes y tamaño de bloque de 4 palabras de 32 bits (16 bytes). ¿Cuál sería la frecuencia de fallos y el tiempo de acceso promedio de los accesos a la memoria cache de datos cuando se ejecuta cada uno de los dos programas de este ejercicio?

Índice	Vía 0				Vía 1			
	0	1	2	3	0	1	2	3
0								
1								

Programa 1: accede a componentes pares A[0],A[2],…,A[18],B[0],B[2],…,B[18].
Índice = bit menos significativo del segundo nible de la dirección de memoria.

Índice	Direcciones	0…3	4…7	8…B	C…F
0	0x1000	N	A[0]	A[1]	A[2]
1	0x1010	A[3]	A[4]	A[5]	A[6]
0	0x1020	A[7]	A[8]	A[9]	A[10]
1	0x1030	A[11]	A[12]	A[13]	A[14]
0	0x1040	A[15]	A[16]	A[17]	A[18]
1	0x1050	A[19]	B[0]	B[1]	B[2]
0	0x1060	B[3]	B[4]	B[5]	B[6]
1	0x1070	B[7]	B[8]	B[9]	B[10]
0	0x1080	B[11]	B[12]	B[13]	B[14]
1	0x1090	B[15]	B[16]	B[17]	B[18]
0	0x10A0	B[19]	D		

Índice	Vía 0				Vía 1			
	0	1	2	3	0	1	2	3
0	N	A[0]	A[1]	A[2]	B[3]	B[4]	B[5]	B[6]
1	A[19]	B[0]	B[1]	B[2]	A[3]	A[4]	A[5]	A[6]

Número de accesos a la memoria cache de datos = 1 (entrada, N) + 10 iteraciones x 2 instrucciones (A[], B[])/iteración + 1 (salida, D) = 22

- Fallos en la cache de datos = 1 (entrada, fallos en frío por los accesos a dirección 0x1000, N) + 1 (1ª iteración, fallos en frío por los accesos a direcciones 0x1054, B[0]) + 2 fallos (3ª iteración, accesos a 0x1014, A[4], índice= 1, y 0x1064, B[4], índice= 2) + 2 fallos (5ª iteración, accesos a 0x1024, A[8], índice= 2, y 0x1074, B[8], índice= 3) + 2 fallos (7ª iteración) + 2 fallos (9ª iteración) + 1 fallo (salida, D) = 1 + 1 + 2 x 4 + 1 = 11
- Tasa de fallos = 100% x 11 / 22 = 50% (no cambia respecto a una cache de correspondencia directa del mismo tamaño total y tamaño de bloque)
- (Programa 1) AMAT = 1 + 0,50 x 50 = 26 ciclos

Programa 2: accede a todas componentes A[0],A[1],…,A[19],B[0],B[1],…,B[19]

Índice	Direcciones	0…3	4…7	8…B	C…F
0	0x1000	N	A[0]	A[1]	A[2]
1	0x1010	A[3]	A[4]	A[5]	A[6]
0	0x1020	A[7]	A[8]	A[9]	A[10]
1	0x1030	A[11]	A[12]	A[13]	A[14]
0	0x1040	A[15]	A[16]	A[17]	A[18]
1	0x1050	A[19]			
0	0x2000	B[0]	B[1]	B[2]	B[3]
1	0x2010	B[4]	B[5]	B[6]	B[7]
0	0x2020	B[8]	B[9]	B[10]	B[11]
1	0x2030	B[12]	B[13]	B[14]	B[15]

0	0x2040	B[16]	B[17]	B[18]	B[19]
1	0x2050	D			

Índice	Vía 0				Vía 1			
	0	1	2	3	0	1	2	3
0	N	A[0]	A[1]	A[2]	B[0]	B[1]	B[2]	B[3]
1	A[3]	A[4]	A[5]	A[6]	B[4]	B[5]	B[6]	B[7]

- Número de accesos a la memoria cache de datos = 1 (entrada, N) + 20 iteraciones x 2 instrucciones (A[], B[])/iteración + 1 (salida, D) = 42
- Fallos en la cache de datos = 1 (entrada, fallos en frío por los accesos a dirección 0x1000, N) + 10 (en accesos de las iteraciones, ver tabla, elementos que están en azul) + 1 fallo (salida, D) = 1 + 10 + 1 = 12
- Tasa de fallos = 100% x 12 / 42 = 28,5% (76,2% en correspondencia directa de mismo tamaño total y de bloque)
- (Programa 2) AMAT = 1 + 0,285 x 50 = 15,3 ciclos (39,1 ciclos en correspondencia directa de mismo tamaño total y de bloque)

- Suponer ahora que la memoria cache de datos es asociativa por conjuntos de 2 vías, y con 8 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits. Calcular: la frecuencia de fallos en la memoria cache, el tiempo de acceso promedio a la jerarquía de memoria si los accesos que aciertan en la memoria cache tardan 1 ciclo de reloj y los accesos a la memoria principal tardan 50 ciclos de reloj.

Organización de la memoria cache de 2 vías y con 8 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits (palabra 0, palabra 1). Los 8 bloques se distribuyen en 4 índices.

Índice	Vía 0		Vía 1	
	Palabra 0	Palabra 1	Palabra 0	Palabra 1
0				
1				
2				
3				

Programa 2: accede a todas componentes A[0],A[1],…,A[19],B[0],B[1],…,B[19], además de N y D; en total 42 accesos

El programa accede a todas componentes A[0],A[1],…,A[19],B[0],B[1],…,B[19], además de N y D; en total 42 accesos. En la siguiente tabla se incluye las direcciones de todas las variables que el programa usa y los correspondientes índices. El índice se obtiene a partir de la dirección de los datos extrayendo los 2 bits que se encuentra a continuación de los bits del campo de la dirección destinados a la selección del byte (3 bits porque el bloque consta de 8 bytes).

Índice	Direcciones	0…3	4…7	**Índice**	Direcciones	8…B	C…F
0	0x1000	N	A[0]	1	0x1008	A[1]	A[2]
2	0x1010	A[3]	A[4]	3	0x1018	A[5]	A[6]
0	0x1020	A[7]	A[8]	1	0x1028	A[9]	A[10]

2	0x1030	A[11]	A[12]	3	0x1038	A[13]	A[14]
0	0x1040	A[15]	A[16]	1	0x1048	A[17]	A[18]
2	0x1050	A[19]		3	0x1058		
0	0x2000	B[0]	B[1]	1	0x2008	B[2]	B[3]
2	0x2010	B[4]	B[5]	3	0x2018	B[6]	B[7]
0	0x2020	B[8]	B[9]	1	0x2028	B[10]	B[11]
2	0x2030	B[12]	B[13]	3	0x2038	B[14]	B[15]
0	0x2040	B[16]	B[17]	1	0x2048	B[18]	B[19]
2	0x2050	D					

Índice	Vía 0		Vía 1	
	0	1	0	1
0	N	A[0]	B[0]	B[1]
1	A[1]	A[2]	B[2]	B[3]
2	A[3]	A[4]	B[4]	B[5]
3	A[5]	A[6]	B[6]	B[7]

Cuando se accede a la cache y se falla se inicializa un bloque de cache con dos palabras: la que originó el fallo y la que se almacena a continuación en memoria. Esto hace que se produzcan 22 fallos en cache: 10 debido a los accesos de los índices impares de A, 10 debido a los accesos de los índices pares de B, además de los accesos a las variables N y D.

Tasa de fallos = 100% x 22 / 42 = 52%
AMAT = 1 + 0,52 x 50 = 27 ciclos

Ejercicio 7.

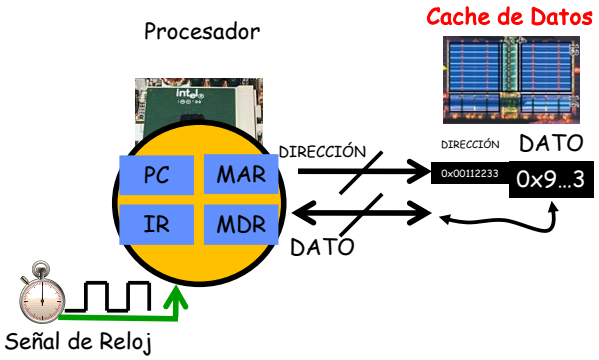
Suponiendo que se ejecuta el siguiente programa en un computador basado en NIOS II/f

```
.org 0x0000
_start: movia r2, A
        movia r3, B
        movia r4, N
        ldw r4, 0(r4)
        add r5, r0, r0
LOOP:   ldw r6, 0(r2)
        ldw r7, 0(r3)
        mul r8, r6, r7
        add r5, r5, r8
        addi r2, r2, 4
        addi r3, r3, 4
        subi r4, r4, 1
        bgt r4, r0, LOOP
        stw r5, D(r0)
STOP:   br STOP

.org 0x1000
N:   .word 20
A:   .word 1,2,…,20

.org 0x2000
B:   .word 20,19,…,1
D:   .skip 4
```

y que el procesador dispone de una memoria cache de datos, asociativa por conjuntos de 2 vías y con 8 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits. Calcular: (a, 80%) la frecuencia de fallos en la memoria cache, (b, 20%) el tiempo de acceso promedio a la jerarquía de memoria si los accesos que aciertan en la memoria cache tardan 1 ciclo de reloj y los accesos a la memoria principal tardan 50 ciclos de reloj.

SOLUCIÓN

Organización de la memoria cache de datos que es asociativa por conjuntos de 2 vías y con 8 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits (Palabra 0, Palabra 1). Los 8 bloques se distribuyen en 4 índices.



En la tabla que aparece a continuación se puede observar la microarquitectura de la memoria cache de datos asociativa por conjuntos de 2 vías. Los campos de la dirección de memoria de 32 bits emitida por el procesador cuando se ejecuta una instrucción de acceso a memoria son los siguientes:

- Selección del byte: 3 bits, 3 = $\log_2$(8 bytes)
- Índice: 2 bits, 2 = $\log_2$(número bloques / asociatividad) = $\log_2$(8/2)
- Etiqueta: 27 bits = 32 (bits de dirección completa) – 3 (selección byte) – 2 (índice)

REPRESENTACIÓN DE LA MICROARQUITECTURA DE LA MEMORIA CACHE DE DATOS ASOCIATIVA

	Vía 0			Índice (2 bits)	Vía 1		
	Etiqueta (27 bits)	Palabra 0	Palabra 1		Palabra 0	Palabra 1	Etiqueta (27 bits)
		N	A[0]	0	B[0]	B[1]	
Un conjunto de cache →		A[1]	A[2]	1	B[2]	B[3]	
				2			
				3			

El programa accede a todas las componentes A[0],A[1],…,A[19],B[0],B[1],…,B[19], además de las variables N y D. En total, se realizan 42 accesos a la memoria cache de datos. En la siguiente tabla se incluye las direcciones de todas las variables que el programa usa y los correspondientes índices. El índice se obtiene a partir de la dirección de los datos extrayendo los 2 bits que se encuentran a continuación de los bits del campo de la dirección destinados a la selección del byte que son 3 bits porque el bloque consta de 8 bytes.

A continuación, se muestra la Tabla de Índices de las direcciones de memoria asignadas a cada dato que maneja el programa del enunciado.

TABLA DE ÍNDICES DE LAS DIRECCIONES DE MEMORIA

Índice	Direcciones de memoria	0…3	4…7	Índice	Direcciones de memoria	8…B	C…F
0	0x1000	N	A[0]	1	0x1008	A[1]	A[2]
2	0x1010	A[3]	A[4]	3	0x1018	A[5]	A[6]
0	0x1020	A[7]	A[8]	1	0x1028	A[9]	A[10]
2	0x1030	A[11]	A[12]	3	0x1038	A[13]	A[14]
0	0x1040	A[15]	A[16]	1	0x1048	A[17]	A[18]
2	0x1050	A[19]		3	0x1058		
0	0x2000	B[0]	B[1]	1	0x2008	B[2]	B[3]
2	0x2010	B[4]	B[5]	3	0x2018	B[6]	B[7]
0	0x2020	B[8]	B[9]	1	0x2028	B[10]	B[11]
2	0x2030	B[12]	B[13]	3	0x2038	B[14]	B[15]
0	0x2040	B[16]	B[17]	1	0x2048	B[18]	B[19]
2	0x2050	D					

En la siguiente tabla se puede observar el contenido de la memoria cache de datos asociativa después de ejecutarse las primeras 7 iteraciones del programa del enunciado.

CONTENIDO DE LA MEMORIA CACHE
DESPUÉS DE EJECUTARSE 7 ITERACIONES

Índice	Vía 0		Vía 1	
	0	1	0	1
0	N	A[0]	B[0]	B[1]
1	A[1]	A[2]	B[2]	B[3]
2	A[3]	A[4]	B[4]	B[5]
3	A[5]	A[6]	B[6]	B[7]

Cuando se accede a la cache y se falla se inicializa un bloque de cache con dos palabras: la que originó el fallo y la que se almacena a continuación en memoria. Esto hace que se produzcan 22 fallos en cache: 10 debido a los accesos de los índices impares de A, 10 debido a los accesos de los índices pares de B incluyendo el índice 0, además de los accesos a las variables N y D.

(a) La frecuencia de fallos se obtiene de la relación entre el número de fallos, 22, y el número de accesos a la memoria cache de datos, 42.

Frecuencia de fallos (FF) = 100% x 22 / 42 = 52%

Tasa de aciertos = 1 – 22/42 = 0,48 (48 %)

(b) Y el tiempo de acceso promedio a la memoria tiene como valor el siguiente resultado habiendo obtenido una frecuencia de fallos en tanto por 1 de 0,52 y usando los datos del enunciado $t_{acierto}$ =1,0 ciclo y P=50,0 ciclos:

AMAT = $t_{acierto}$ + FF x P = 1 + 0,52 x 50 = 27 ciclos

Ejercicio 8.
Suponiendo que se ejecuta el siguiente programa en un computador basado en NIOS II/f

```
.org 0x0000
_start: movia r2, A
        movia r3, B
        movia r4, N
        ldw r4, 0(r4)
        add r5, r0, r0
LOOP:   ldw r6, 0(r2)
        ldw r7, 0(r3)
        mul r8, r6, r7
        add r5, r5, r8
        addi r2, r2, 4
        addi r3, r3, 4
        subi r4, r4, 1
        bgt r4, r0, LOOP
        stw r5, D(r0)
STOP:   br STOP

.org 0x1000
N:   .word 20
.org 0x1100
A:   .word 1,2,…,20
 .org 0x1200
B:   .word 20,19,…,1
.org 0x1300
D:   .skip 4
```

y que el procesador dispone de una memoria cache de datos, de correspondencia directa y con 8 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits. Se solicita: (a, 20%) obtener el índice de la memoria cache al que se asigna cada palabra de datos del programa explicando cómo se calcula, (b, 60%) calcular la frecuencia de fallos en la memoria cache explicando cómo se ha realizado el cálculo, (c, 20%) calcular el tiempo de acceso promedio a la jerarquía de memoria si los accesos que aciertan en la memoria cache tardan 1 ciclo de reloj y los accesos a la memoria principal tardan 50 ciclos de reloj.

SOLUCIÓN

Organización de la memoria cache de 2 vías y con 8 bloques de datos, donde cada bloque de datos puede alojar 2 palabras de 32 bits (palabra 0, palabra 1). Los 8 bloques se distribuyen en 4 índices.

Índice	Palabra 0 (4 bytes)	Palabra 1 (4 bytes)
0		
1		
2		
3		
4		
5		
6		
7		

El programa accede a todas componentes A[0],A[1],…,A[19],B[0],B[1],…,B[19], además de N y D; en total 42 accesos. En la siguiente tabla se incluye las direcciones de todas las variables que el programa usa y los correspondientes índices. El índice se obtiene a partir de la dirección de los datos extrayendo los 2 bits que se encuentran a continuación de los bits del campo de la dirección destinados a la selección del byte (3 bits porque el bloque consta de 8 bytes).

Dirección de memoria:

Etiqueta	Índice			Selección del byte dentro del bloque
31 30 … 7 6	5	4	3	2 1 0

Índice	Direcciones de memoria	0…3	4…7	**Índice**	Direcciones de memoria	8…B	C…F
0	0x1000	N					
0	0x1100	A[0]	A[1]	1	0x1108	A[2]	A[3]
2	0x1010	A[4]	A[5]	3	0x1018	A[6]	A[7]
4	0x1020	A[8]	A[9]	5	0x1028	A[10]	A[11]
6	0x1030	A[12]	A[13]	7	0x1038	A[14]	A[15]
0	0x1040	A[16]	A[17]	1	0x1048	A[18]	A[19]
0	0x1200	B[0]	B[1]	1	0x1200	B[2]	B[3]
2	0x1210	B[4]	B[5]	3	0x1218	B[6]	B[7]
4	0x1220	B[8]	B[9]	5	0x1228	B[10]	B[11]
6	0x1230	B[12]	B[13]	7	0x1238	B[14]	B[15]
0	0x1240	B[16]	B[17]	1	0x1248	B[18]	B[19]
0	0x1300	D					

Índice	Palabra 0 (4 bytes)	Palabra 1 (4 bytes)
0	N, A[0], B[0]	A[1], B[1]
1	A[2], B[2]	A[3], B[3]
2	A[4], B[4]	A[5], B[5]
3		
4		
5		
6		
7		

Cuando se accede a la cache y se falla se inicializa un bloque de cache con dos palabras: la que originó el fallo y la que se almacena a continuación en memoria. Sin embargo, como en cada iteración del programa se accede a componentes con el mismo orden de las matrices A y B, y ambas componentes coinciden en índice, esto hace que se produzcan 42 fallos en cache: 20 debido a los accesos de todos los índices de A, 20 debido a los accesos de todos los índices de B, además de los accesos a las variables N y D.

Tasa de fallos = 100% x 42 / 42 = 100%

AMAT = 1 + 1 x 50 = 51 ciclos