ESTRUCTURA DE COMPUTADORES
PROPUESTA JUNIO
Curso 2011-2012

SEGMENTACIÓN

El siguiente fragmento de código se ejecuta en un procesador segmentado de 5 etapas:

```
LOOP:   LD      F2, 0(R1)
        ADDD    F4, F2, F0
        LD      F8, 0(R1)
        MULTD   F6, F4, F8
        SUB     F6,F4,F8
        MULTD   F10,F4,F8
        SD       0(R1), F6
        SUBI    R1, R1, 8
        BNEQZ   R1, LOOP
        NOP
```

Suponiendo que:

- un dato se puede escribir en un registro y leer su valor en el mismo ciclo
- los riesgos de lde se detectan en decodificación y se solucionan mediante paradas
- se dispone de lógica de cortocircuito
- los riesgos de escritura después de escritura se solucionan inhibiendo la escritura de la primera instrucción
- no permite dos instrucciones en memoria o escritura
- riesgos estructurales se evitan mediante paradas
- los saltos se resuelven en la etapa de decodificación
- los riesgos de control se solucionan mediante paradas
- las unidades funcionales tienen las siguientes características:

UF	Cantidad	Latencia	Segmentación
FP ADD	1	3	SI
FP SUB	1	3	SI
FP MUL	1	4	NO
Int ALU	1	1	No

a) Representar el diagrama instrucción-tiempo para la primera iteración e indicar los cortocircuitos realizados. Indicar claramente las paradas y sus causas.
b) Determinar el CPI teniendo en cuenta que r1=0x1000 al inicio de la ejecución.

Solución

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
LD F2, 0(R1)	F	D	E	M	W															
ADDD F4, F2, F0		F	D_P	D	A1	A2	A3	M	W											
LD F8, 0(R1)			F_P	F	D	E	M	W												
MULTD F6, F4, F8					F	D_P	D	X1	X2	X3	X4	M	W							
SUB F6,F4,F8						F_P	F	D_P	D	S1	S2	S3	M	W						
MULTD F10,F4,F8								F_P	F	D_P	D	X1	X2	X3	X4	M	W			
SD 0(R1), F6										F_P	F	D	E	M	W					
SUBI R1, R1, 8												F	D	E	M	W				
BNEQZ R1, LOOP													F	D_P	D	E	M	W		
LD F2, 0(R1)														F_P	F_P	F	D	E	M	W

XXX riesgo LDE;
XXX riesgo estructural: parada en la instrucción anterior,
XXX riesgo de ede: inhibición por EDE;
XXX riesgo estructural: dos instrucciones en M
XXX riesgo estructural: dos instrucciones intentan acceder al multiplicador que no está segmentado
XXX riesgo de control

Cpi=15/9=1,66

Memoria

PROBLEMA 9

Considerar un computador con una memoria principal de 64K bytes, direccionable en bytes, al que se dota de una memoria cache de 1K bytes, con líneas de 128 bytes, y prebúsqueda bajo fallo (en caso de fallo se trae el bloque que lo provoca y el siguiente).
Sea la secuencia de acceso a memoria principal dada por las siguientes direcciones: 3345h,14BFh, 1484h,43A1h,55ABh,55A1h
Mostrar la evolución de la caché de etiquetas indicando los fallos y las prebúsquedas que se producen.

a) una organización directa
b) una organización asociativa por conjuntos de 2 vías, con algoritmo de reemplazo LRU

SOLUCIÓN PROBLEMA 9

Nota: En las tablas cada columna representa el estado del directorio cache para cada una de las referencias a memoria mp, excepto la primera columna que indica el marco de bloque o el conjunto. En cada casilla se indica el valor de la información de directorio en **binario** y un símbolo indicando fallo, prebúsqueda y acierto. (F) indica fallo, (P)prebúsqueda y (A) acierto.
a)
Lo primero que hay que hacer es buscar como se interpreta la dirección de Mp en la memoria caché. Sabemos que:

- la memoria principal tiene 64k = 2^{16} --> necesitamos 16 bits para construir la dirección.
- La memoria cache tiene 1K bytes = 2^{10} como cada línea son 128 bytes=2^7 bytes esto implica que la memoria caché tiene $2^{10}/2^7 = 2^3$ líneas -->necesitamos 3 bits para direccionarlas
- Como las líneas son de 128 bytes =2^7 --> 7 bits para determinar la palabra dentro de la línea.

Luego la dirección es interpretada por la Mc de la siguiente manera

Cache tag	Cache index	Byte select
15:10	9:7	6:0

Para rellenar las tabla tenemos que interpretar las direcciones físicas que nos llegan como direcciones de caché. Las direcciones están dadas en hexadecimal, tenemos que buscar el valor de los bits correspondientes al cache index para saber cómo va evolucionando el directorio de cache. La tabla se tiene que rellenar con los bits de la cache tag, por lo tanto sólo nos interesan los dos primeros dígitos hexadecimales más representativos, puesto que los dos primeros corresponden a la selección del byte. En la siguiente tabla ponemos la interpretación de la dirección para cada una de las referencias

	tag	index	select
	15:10	9:7	6:0
3345h	0011 00	11 0	100 0101
14BFh	0001 01	00 1	011 1111
1484h	0001 01	00 1	000 0100
43A1h	0100 00	11 1	010 0001
55ABh	0101 01	01 1	010 1011
55A1H	0101 01	01 1	111 0001

	T1	T2	T3	T4	T5	T6
Dirección	001100-110	000101-001	000101-001	010000-111	010101-011	010101-011
0				010001(P)	010001	010001
1		000101(F)	000101(A)	000101	000101	000101
2		000101(P)	000101	000101	000101	000101
3				000101	010101(F)	010101 (A)
4				000101	010101(P)	010101
5						
6	001100(F)	001100	001100	001100	001100	001100
7	001100(P)	001100	001100	010000(F)	010000	010000

b)
Igual que antes lo primero es determinar cómo se interpreta la dirección física
El tamaño total de la dirección sigue siendo el mismo: 16 bits
El número de bits para determinar el byte al que se está accediendo también es el mismo: 7bits
La diferencia es que ahora no se direccionan líneas , sino conjuntos de líneas . Como es una organización asociativa por conjuntos de 2 vías, esto nos indica que cada conjunto tiene 2 líneas luego el número de conjuntos es 2^3 líneas/2 líneas por conjunto = 4 conjuntos, luego necesitamos 2 bits para direccionarlos

Cache tag	Cache index	Byte select
15:9	8:7	6:0

Ahora la tabla de las direcciones queda:

	tag	index	select
	15:9	8:7	6:0
3345h	0011 001	1 0	100 0101
14BFh	0001 010	0 1	011 1111
1484h	0001 010	0 1	000 0100
43A1h	0100 001	1 1	010 0001
55ABh	0101 010	1 1	010 1011
55A1H	0101 010	1 1	111 0001

Dirección	0011001-10	0001010-01	0001010-01	0100001-11	0101010-11	0101010-11
0				0100 010(P)	0100 010	0100 010
					0101010(P)	0101010
1		0001010(F)	0001010(A)	0001010	0001010	0001010
2	0011001 (F)	0011001	0011001	0011001	0011001	0011001
		0001010(P)	0001010	0001010	0001010	0001010
3	0011001(P)	0011001	0011001	0011001	0101010(F)	0101010(A)
				0100 001(F)	0100 001	0100 001