P12021sol.pdf

Olmar_eps

Arquitectura de Ordenadores

3º Grado en Ingeniería Informática

Escuela Politécnica Superior
Universidad Autónoma de Madrid

Arquitectura de Ordenadores – Ejercicios Tema 1 y 2

E1.- Un programa de un millón de instrucciones tiene la distribución de instrucciones indicada en la siguiente tabla. También se indica el CPI de cada tipo de instrucción al ejecutarlo en un procesador RISC.

	Unidad INT	LD/ST	Unidad CF	Resto instrucciones
%Instrucciones	50	10	25	15
CPI	5	32	16	2

a) ¿Cuál es el CPI del programa?

CPI= 0,5x5+0,1x32+0,25x16+0,15x2= 10

b) Si la frecuencia del procesador es de 200 MHz ¿Cuál el es rendimiento en MIPS?

Rdto MIPS= f / (CPI x 10^6) = 200/10 = 20 MIPS

c) Se plantea aplicar una mejora a la unidad de CF y otra a la unidad LD/ST. ¿Qué mejora deben conseguirse en el CPI la unidad de LD/ST para que la aceleración global sea equivalente a cuando se aplica una mejora de un CPI = 4 para la unidad de CF?

CF: Am1= 4 ;
Fm1= 4/10 = 0,4 A1= 1/((1-Fm1)+Fm1/Am1)

LD/ST Am2= ¿?
Fm2=3,2/10=0,32 A2= 1/((1-Fm2)+Fm2/Am2) y queremos A1=A2

1/((1-0,4)+0,4/4) = 1/(1-0,32)+0,32/Am2
1/0,7= 1/(0,68+0,32/Am2)
0,7 = 0,68 + 0,32/Am2
0,02 = 0,32/Am2 Am2=0,32/0,02=16

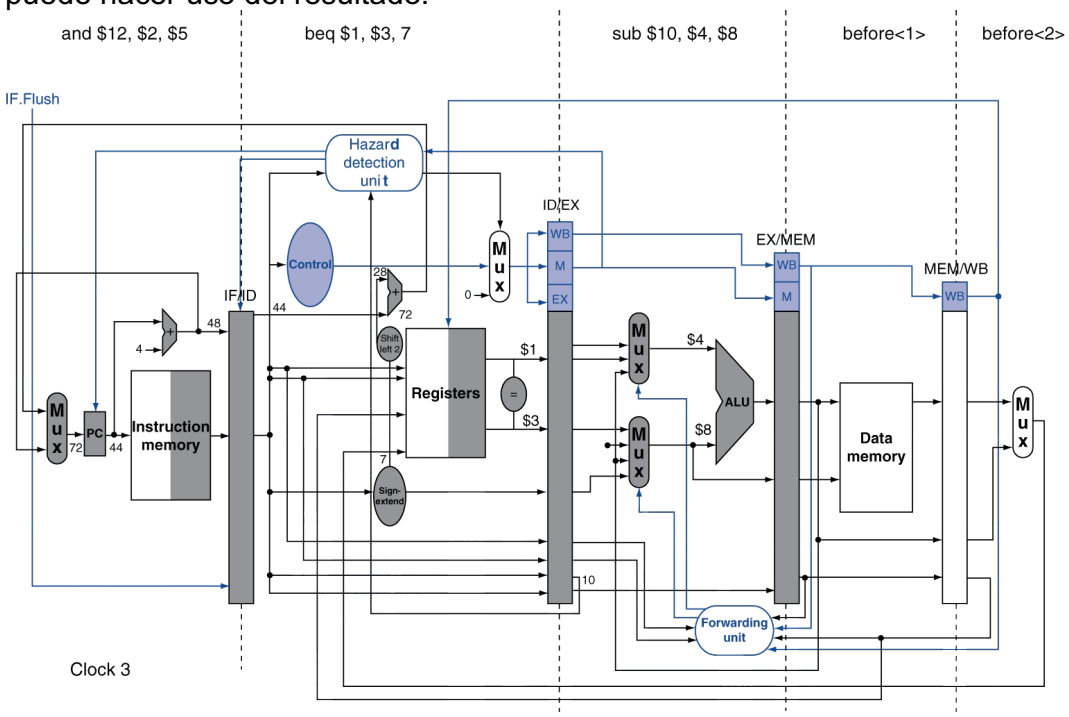
d) Si se aplica la misma mejora a la unidad funcional de CF y a la unidad funcional de LD/ST, ¿Qué situación es preferible, mejorar CF o mejorar LS/ST, para conseguir mayor aceleración global?

Si Am1 =Am2
como Fm1 (CF) es mayor Fm2, entonces la A1 (mejora CF) es mayor que A2

E2.- Suponga que se modifica el procesador visto en teoría e implementado en práctica, adelantado la resolución del salto a la 2da etapa (DE - decodificación).
El equipo de diseño ha decidido evitar los posibles riesgos de datos deteniendo el pipeline en las etapas IF e ID (en vez de adelantar los datos a ID), detectando el riesgo en ID de forma similar a la “Hazard Detection Unit” y generando los NOP correspondientes.

Arquitectura de Ordenadores – Ejercicios Tema 1 y 2

Recordar que cuando una instrucción escribe en la etapa WB, la instrucción que está en ID puede hacer uso del resultado.



Cuantos ciclos de detención genera estas secuencias de instrucciones

a) caso 1: Se pierde(n) 2 ciclo(s)

Instrucciones	01	02	03	04	05	06	07	08	09	10	11	12
ADD R1, R2, R3	IF	ID	EX	M	W							
BEQ R1, R4, inm		IF	(ID)	(ID)	ID	EX	M	W				
MUL R5, R6, R6			(IF)	(IF)	IF	ID	EX	M	W			
SUB R7, R5, R1						IF	ID	EX	M	W		

- BEQ necesita R1 para hacer comparación.
- "SUB R7, R5, R1" no tiene detenciones ya que el adelantamiento a EX si existen (Forwarding Unit)

b) caso 2: Se pierde(n) 1 ciclo(s)

Instrucciones	01	02	03	04	05	06	07	08	09	10	11	12
ADD R1, R2, R3	IF	ID	EX	M	WB							
XOR R7, R2, R8		IF	ID	EX	M	WB						
BEQ R1, R4, inm			IF	(ID)	DE	EX	M	WB				
MUL R5, R6, R6					IF	ID	EX	M	WB			

Se modifica el procesador de modo que se puedan efectuar todos los adelantamientos posibles a ID. Como cambiarían los retardos en los casos anteriores. Indicar desde que etapa a que etapa se producen adelantamientos.

a) caso 1: Se pierde(n) 1 ciclo(s)

Instrucciones	01	02	03	04	05	06	07	08	09	10	11	12

ADD R1, R2, R3	IF	ID	EX	M	W							
BEQ R1, R4, inm		IF	(ID)	ID	EX	M	W					
MUL R5, R6, R6			(IF)	IF	ID	EX	M	W				
SUB R7, R5, R1					IF	ID	EX	M	W			

No se puede adelantar en el ciclo 3, ya que se está calculando en EX el dato necesario.
Puede adelantarse desde M (el dato producido en el ciclo anterior en EX) a ID

b) caso 2: Se pierde(n) **0** ciclo(s)

Instrucciones	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**12**
ADD R1, R2, R3	IF	ID	EX	M	WB							
XOR R7, R2, R8		IF	ID	EX	M	WB						
BEQ R1, R4, inm			IF	DE	EX	M	WB					
MUL R5, R6, R6				IF	ID	EX	M	WB				

Puede adelantarse desde M (el dato producido en el ciclo anterior en EX) a ID

Arquitectura de Ordenadores – Ejercicios Tema 1 y 2

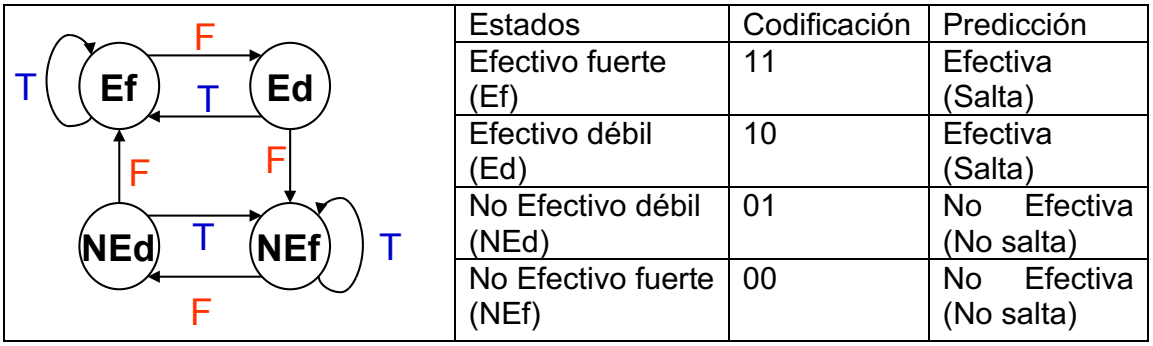
E3.- Para un BTB de 16 entradas, con el contenido que se muestra en la tabla, y que corresponde a una situación anterior a la ejecución del salto en el siguiente fragmento de código. Las entadas no completas, no afectan a la solución.

```
2000 003C    LOAD R4, 64(R0)        ; 64 en decimal = 0x40 hexadecimal
2000 0040    BEQ  R1, R3, 7         ;  7 en decimal = 0x07 hexadecimal
2000 0044    ADD  R1, R2, R5
2000 0048    SUB  R2, R1, R0
…
2000 0058    MUL  R3, R1, R4
2000 005C    SUB  R2, R12, R1
2000 0060    ADD  R1, R3, R0
2000 0064    ADD  R4, R2, R5
```

Contenido del BTB:

Dirección del código	Dirección del salto	Predicc
…	…	-
0000 1000	1000 003C	11
0000 10A0	1000 00D2	00
…	…	-
2000 0040	2000 0060	~~10~~ 11
2000 1010	2000 0040	01
…	…	-
…	…	-

El valor del campo predicción posee la siguiente codificación:



Estados	Codificación	Predicción
Efectivo fuerte (Ef)	11	Efectiva (Salta)
Efectivo débil (Ed)	10	Efectiva (Salta)
No Efectivo débil (NEd)	01	No Efectiva (No salta)
No Efectivo fuerte (NEf)	00	No Efectiva (No salta)

a) ¿Qué predicción realiza el BTB para el salto BEQ R1, R3, 7?
La entrada del BTB correspondiente a este salto es la tercera y realiza predicción efectiva (10)

b) ¿Cuál es la siguiente instrucción a entrar en el pipeline tras BEQ R1, R3, 7?
Al existir un acierto en BTB, la próxima instrucción será la de la dirección que figura en el TLB : 2000 0060, es decir ADD R1, R3, R0. La dirección del salto es PC+4+7*4 = PC + 32d = PC + 20h

c) Indique (sobre la misma figura) cómo se modifica el contenido del BTB después de ejecutar el salto, suponiendo que el salto ha sido efectivo y que la evolución del sistema de predicción es la indicada en la máquina de estados.

Arquitectura de Ordenadores – Ejercicios Tema 1 y 2

E4.- La siguiente secuencia de código se ejecuta en un sistema con arquitectura Harvard y un procesador segmentado con 5 etapas como el descrito en el libro de Patterson y Hennessy: **F** (captura de instrucciones), **D** (decodificación, detección de riesgos y lectura de registros), **E** (opera en la ALU, calcula la dirección efectiva, evalúa la condición en saltos), **M** (acceso a memoria de datos y actualización en saltos del PC*) y **W** (escritura en el banco de registros). El banco de registros permite escritura y lectura en el mismo ciclo.

*Nota: En la etapa M de las instrucciones de salto la entrada del registro PC contiene la dirección de la instrucción destino del salto.

Analice el funcionamiento del siguiente programa, empezando en la captura de la instrucción I1 y suponiendo que el salto (instrucción I4) es efectivo (salta al destino).

```
I1 ADD R1, R2, R3          ; R1 = R2+R3
I2 LOAD R4, 0(R1)          ; R4 = M[0+R1]
I3 MUL R5, R4, R4          ; R5 = R4·R4
I4 BNE R1, R4, L1          ; Si R1 /= R4, salta a L1 (sí se salta)
I5 SUB R5, R5, R2          ; R5 = R5-R2
I6 ADD R7, R4, R5          ; R7 = R4+R5
I7 ADDI R7, R7, 1          ; R7 = R7+1
I8 / L1: ADD R6, R1, R5    ; R6 = R1+R5
I9: OR R1, R7, R8          ; R1 = R7 or R8
I10: ST R6, 0(R1)          ; M[0+R1] = R6
```

a) Enumere los riesgos de datos (solo RAW) que se presentan en la secuencia de código y que hay que detectar al ejecutarlo en este procesador.

I2 con I1 por R1
I3 con I2 por R4
I4 con I2 por R4
I4 con I1 por R1
I5 con I3 por R5
I6 con I5 por R5
I6 con I2 por R1
I7 con I6 por R7
I8 con I5 por R5
I8 con I1 por R1
I9 con I7 por R7
I10 con I9 por R1
I10 con 18 por R6

Complete los siguientes cronogramas de ejecución bajo los siguientes supuestos:

b) Sin adelantamientos. Los saltos condicionales se predicen no efectivos.

Instrucción \ tiempo	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
I1 ADD R1, R2, R3	F	D	E	M	W															
I2 LOAD R4, 0(R1)		F	(D)	(D)	D	E	M	W												
I3 MUL R5, R4, R4					F	(D)	(D)	D	E	M	W									
I4 BNE R1, R4, L1								F	D	E	M	W								
I5 SUB R5, R5, R2									F	(D)	~~D~~	~~E~~	~~M~~	~~W~~						
I6 ADD R7, R4, R5										F	~~(F)~~	~~(D)~~	~~(D)~~	~~D~~	~~E~~	~~M~~	~~W~~			
I7 ADDI R7, R7, 1											~~F~~									
I8 / L1: ADD R6, R1, R5												F	D	E	M	W				
I9: OR R1, R7, R8													F	D	E	M	W			
I10: ST R6, 0(R1)														F	(D)	(D)	D	E	M	W
…																				

c) Con adelantamientos. Los saltos condicionales se predicen no efectivos.

Instrucción \ tiempo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
I1 ADD R1, R2, R3	F	D	E	M	W															
I2 LOAD R4, 0(R1)		F	D	E	M	W														
I3 MUL R5, R4, R4			F	(D)	D	E	M	W												
I4 BNE R1, R4, L1					F	D	E	M	W											
I5 SUB R5, R5, R2						F	D	~~E~~	~~M~~	~~W~~										
I6 ADD R7, R4, R5							F	~~D~~	~~E~~	~~M~~	~~W~~									
I7 ADDI R7, R7, 1								~~F~~												
I8 / L1: ADD R6, R1, R5									F	D	E	M	W							
I9: OR R1, R7, R8										F	D	E	M	W						
I10: ST R6, 0(R1)											F	D	E	M	W					
…																				

d) Con adelantamientos e incorporando un sistema de predicción basado en un BTB y suponiendo que el salto está en el BTB y acierta en la predicción de salto.

Instrucción \ tiempo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
I1 ADD R1, R2, R3	F	D	E	M	W															
I2 LOAD R4, 0(R1)		F	D	E	M	W														
I3 MUL R5, R4, R4			F	(D)	D	E	M	W												
I4 BNE R1, R4, L1					F	D	E	M	W											
I5 SUB R5, R5, R2																				
I6 ADD R7, R4, R5																				
I7 ADDI R7, R7, 1																				
I8 / L1: ADD R6, R1, R5						F	D	E	M	W										
I9: OR R1, R7, R8							F	D	E	M	W									
I10: ST R6, 0(R1)								F	D	E	M	W								
…																				