Arquitectura de Computadoras

Solución primer parcial - 04 Marzo 2019

1. Seleccione cuáles de los siguientes registros son ”visibles”, es decir, su contenido puede ser accedido o modificado directamente con el repertorio de instrucciones del procesador. El procesador tiene 8 registros R0 a R7 y memoria cache de un nivel.

MAR	*Palabra en Memoria*	MBR	IR
Palabra en cache	*PC*	*R3*	T

2. Ensamble la siguiente expresión en (a) una arquitectura de una dirección (acumulador) y (b) una arquitectura con tres direcciones a registros de CPU.

$$Y = \frac{A - B + C}{G + H}$$

LD G	;$Acc \leftarrow M[G]$
ADD H	; $Acc \leftarrow Acc + M[H]$
STO Y	;$M[Y] \leftarrow Acc$
LD A	; $Acc \leftarrow M[A]$
SUB B	;$Acc \leftarrow Acc - M[B]$
ADD C	;$Acc \leftarrow Acc + M[C]$
DIV Y	;$Acc \leftarrow Acc/M[Y]$
STO Y	;$M[Y] \leftarrow Acc$

LD R1,A	;$R1 \leftarrow M[A]$
LD R2,B	;$R2 \leftarrow M[B]$
LD R3,C	;$R3 \leftarrow M[C]$
LD R4,G	;$R4 \leftarrow M[G]$
LD R5,H	;$R5 \leftarrow M[G]$
ADD R6,R4,R5	; $R6 \leftarrow G + H$
ADD R7,R2,R3	;$R7 \leftarrow B + C$
SUB R8,R1,R7	; $R8 \leftarrow A - B + C$
DIV R9,R8,R6	;$R8 \leftarrow (A - B + C)/(G + H)$
STO Y,R8	;$M[Y] \leftarrow R8$

3. Anote en la tabla si las modificaciones señaladas incrementarían (I), decrementarían (D) o dejarían sin cambio (-) cada uno de los miembros de la *Ley de acero del desempeño del procesador*. (No dejes de considerar qué impacto tiene con los saltos).

   - (A) Cambiar un pipeline de 2 etapas (F,E) por uno de tres etapas (F,E,W). Ambos tienen atajos.
   - Mover la lógica de saltos de la etapa E a la etapa D
   - Mezclar las etapas D y E (hace lectura de registros y ejecución de ALU -que es combinatoria- en el mismo ciclo

Cambio	Instruction/program	Cycles/Instruction	Seconds/cycle
A	-	-	*D*
B	-	*D*	*I*
C	-	*D*	*I*

4. Considere un procesador con 32 registros. El primero, R0, simplemente almacena un valor constante de cero. También tiene una ALU que recibe como fuente dos de los 32 registros. Los registros forman parte del código de operación. En operaciones aritméticas, se pueden activan las banderas de Z (cero), N (negativo), C (acarreo) o V (sobreflujo).

   El ciclo fetch toma tres tiempos. Diseñe la secuencia de microoperaciones necesarias para implementar la instrucción **BRZ Rx, offset** la cual tiene un código de operación de 7, Rx puede ser cualquiera de los registros y offset, tiene el valor que debe SUMARSE al PC en caso de que Rx valga cero. Si lo considera necesario, justifique cualquier decisión de diseño ”no convencional” (es decir, no considerada en el tipo de computadoras que diseñamos en clase).

   Como hemos hecho en el curso, supondremos que offset está en la siguiente palabra en memoria. En la primer microinstrucción supondremos:
   (1) Que R_0 puede ser destino ficticio para una operación aritmética, es decir, se puede indicar como destino aunque el resultado no se guarde;
   (2) Las banderas se conservan activas con el valor que obtuvieron en la última operación aritmética;
   (3) Los operandos fuente y destino se direccionan directamente desde campos del código de instrucción, como se hace en la arquitectura MIPS.

$$
\begin{aligned}
q_7t_3 &: R_0 \leftarrow R_0 + R_x; && \textit{Se enciende la bandera Z si } R_x \textit{ vale 0}\\
q_7t_4\bar{Z} &: PC \leftarrow PC + 1; && \textit{Simplemente ignora palabra con offset}\\
&\quad T \leftarrow 0 &&\\
q_7t_4Z &: MAR \leftarrow PC &&\\
q_7t_5Z &: MBR \leftarrow M[MAR]; && \textit{Tomamos el offset de memoria}\\
q_7t_6Z &: PC \leftarrow PC + MBR; && \textit{... y lo sumamos al PC}\\
&\quad T \leftarrow 0 &&
\end{aligned}
$$

5. Considere una memoria cache 4-way associative de 256 bytes (en total) con bloques de 8 bytes.

   (a) ¿Cuántos sets tiene la memoria?
   $256/4 = 64$ *bancos o ways. Cada línea (o set) tiene 8 bytes. Por consiguiente, se tienen* $64/8 = 8$ *sets*

   (b) ¿Cuál es la tasa de hit si se accede secuencialmente a las siguientes localidades de memoria: 0, 512, 1024, 1536, 2048, 1536, 1024, 512, 0?

Dirección	*Valor binario*
0	*0000 0000 0000*
512	*0001 0000 0000*
1024	*0010 0000 0000*
1536	*0011 0000 0000*
2048	*0100 0000 0000*

Todas las localidades de memoria tienen los últimos 6 bits (3 de índice y 3 de offset) en cero. En mapeo directo, tendríamos muchos conflictos pero tratándose de una 4-way set associative, podemos almacenar cuatro bloques con la misma pareja [index,offset].
Se almacenan los bloques donde están las palabras 0, 512, 1024 y 1536, pero la primera vez que se acceden, hay un compulsory miss; se tiene un conflicto con la palabra 2048 y se debe remplazar con cualquiera, por ejemplo con el bloque de la palabra 0.
Los siguientes tres accesos (1536, 1024 y 512) son un hit, pero cuando se accede de nuevo a la palabra 0, hay un miss y tiene que ser remplazada nuevamente.
En conclusión, de los nueve accesos, sólo tres fueron un hit. La tasa de hit es de 33%*. Por supuesto, este es un resultado sesgado por la cantidad de fallos obligatorios al estar la cache vacía.*

6. Si las direcciones a memoria principal son de 32 bits, se utiliza una memoria cache de mapeo directo, el campo de índice es de 8 bits y el de offset es de 4 bits, ¿De qué tamaño son:
   - Las etiquetas *20 bits*
   - Los bloques (en bytes) *16*
   - El número de sets *256*
   - La memoria cache *4kB*

7. ¿Cuál es el tiempo de acceso efectivo a memoria para un sistema que tiene un tiempo de acceso a cache de 3 pulsos de reloj, acceso a memoria de 15 pulsos de reloj y tasa de hit de 90%?

$$3 + (0.1)15 = 4.5 \, \textit{pulsos de reloj}$$

8. Responda cierto o falso. El renombramiento de registros:
   - *falso* Elimina bloqueos derivados de dependencias de nombres en memoria
   - *cierto* Elimina bloqueos por dependencias de salida (WAW)
   - *cierto* Elimina bloqueos por antidependencias (WAR)
   - *falso* Elimina bloqueos por dependencias reales (RAW)

9. ¿Cuáles son las tres principales razones por las cuales el pipeline de un procesador no siempre está lleno?

   Porque la ejecución de instrucciones genera "conflictos" de tres tipos: Estructurales (no hay hardware disponible), de acceso a Datos (dependencias de datos) y de control (saltos en el código)

10. Un procesador cuenta con cinco etapas que tardan respectivamente 50, 50, 60, 50 y 50 nSeg. Al introducir pipelining, los buffers auxiliares agregan un retardo adicional de 5 nSeg cada uno. ¿Cuál es la aceleración esperada *speedup* con la nueva arquitectura?

El primer procesador tarda en ejecutar una instrucción $4(50) + 60 = 260$ *nSeg.*
Con la introducción del pipeline, todas las etapas se ajustan a la más lenta, que en este caso será 65 nSeg. La aceleración es:

$$SpUp = \frac{260}{65} = 4$$

11. (1 punto) Se ha estimado que cierta aplicación de cómputo científico pasa el 75% de su ejecución realizando operaciones de punto flotante. Un nuevo procesador introduce una unidad funcional de punto flotante en pipeline, que incrementa 3 veces la velocidad de ejecución de operaciones. ¿En cuánto se incrementa la velocidad de ejecución de la aplicación?

Para un código determinado, el primer procesador tardaría $25 + 75 = 100$ *unidades de tiempo. El segundo tardaría* $25 + 75/3 = 50$ *unidades de tiempo. Es decir, el segundo procesador ejecuta dos veces más rápido la aplicación.*

12. (2 puntos) Una computadora tiene las siguientes operaciones:

Código	Mnemónico	Comentario
00	LD INDIR	$A \leftarrow M[PTR]$
01	LDI PTR	$PTR \leftarrow dato$
02	INC PTR	$PTR \leftarrow PTR + 1$
03	MOVR	$R \leftarrow A$
04	ADDR	$A \leftarrow A + R$
05	ADDI	$A \leftarrow A + dato$
06	SAVE	$M[dir] \leftarrow A$
07	DJNZ dir	$B \leftarrow B - 1$. Si B ahora vale 0, contínúa con la siguiente instrucción; de lo contrario, salta a dir

(a) Implemente las primeras operaciones LD INDIR, ADDR y DJNZ en RTL

LD INDIR	*ADDR*	*DJNZ dir*
$q_0t_3 : MAR \leftarrow PTR$	$q_4t_3 : A \leftarrow A + R,$ $T \leftarrow 0$	$q_6t_3 : MAR \leftarrow PC$
$q_0t_4 : MBR \leftarrow M[MAR]$		$q_6t_4 : MBR \leftarrow M[MAR],$ $PC \leftarrow PC + 1$
$q_0t_5 : A \leftarrow MBR,$ $T \leftarrow 0$		$q_6t_5 : B \leftarrow B - 1$ $q_6t_6Z : T \leftarrow 0$ $q_6t_6\bar{Z} : PC \leftarrow MBR, T \leftarrow 0$

(b) Con el repertorio de instrucciones, diseñe un programa en ensamblador que permita ejecutar el segmento de código:

```
Suma: = 0;
For I : = 1 to 10
        Suma : = Suma + Arreglo [I];
```

La variable Suma se encuentra almacenada en la localidad 10; el arreglo Arreglo en la localidad 15 y el programa inicia en la localidad 50.

En el examen faltó la instrucción parar cargar B. Es un error, lo lamento, aunque era de esperar que esta instrucción exista, de lo contrario, no tiene sentido la instrucción DJNZ.

LOCALIDAD	*CONTENIDO*	*MNEMO*	
50, 51		*LDI B, 10*	*; B = 10*
52, 53	*01, 15*	*LDI PTR, 15*	*; PTR = 15 = &Arreglo*
54	*00*	*LD INDIR*	*;A = Arreglo[0]*
55	*03*	*MOVR*	*; R = Suma (= 0 + Arreglo[0])*
56	*02*	*INC PTR*	*;PTR = 16 = &Arreglo[1]*
57	*00*	*LD INDIR*	*;A = Arreglo[1]*
58	*04*	*ADDR*	*;A = Suma + Arreglo[1]*
59, 60	*07, 55*	*DJNZ 55*	*; Nueva iteración si B ¿ 0*
61, 62	*06, 10*	*SAVE Suma*	*; Termina. Guarda Suma*

13. (2 puntos) Considere una computadora con las siguientes características:

- 95% de todos los accesos a memoria se encuentran en cache (hit)
- Un bloque de cache es de dos palabras.
- 75% de los accesos a memoria son lecturas; el 25% restante son escrituras.
- El bus puede transferir únicamente una palabra de memoria a cache por ciclo
- El 30% de los bloques son *dirty*.
- La estrategia de remplazo en un fallo de escritura (*write miss*) es *write allocate*.

Llene la siguiente tabla. Indique el número de palabras transferidas en el bus en cada caso y calcule el número de hits en lectura y escritura como porcentaje de las instrucciones. A manera de ejemplo, el primer renglón de la tabla ya está lleno.

Análisis en hit				
		No. accesos a memoria principal		Probabilidad
		Read	Write	
Read hit		**0**	**0**	95% × 75%
Write hit	Write through	*0*	*1*	95% × 25%
	Write back	*0*	*0*	95% × 25%
Análisis en miss				
Read miss	Write through	*2*	*0*	5% × 75%
	Wr Back cache dirty	*2*	*2*	5% × 30% × 75%
	Wr Back cache clean	*2*	*0*	5% × 70% × 75%
Write miss	Write through	*2*	*1*	5% × 25%
	Wr Back cache dirty	*2*	*2*	5% × 30% × 25%
	Wr Back cache clean	*2*	*0*	5% × 70% × 25%

14. (1 punto) Una memoria cache de 8 kBytes con política de mapeo directo, está formada por bloques de 32 bytes. El procesador puede direccionar 4 GBytes. Además de las etiquetas para identificar si una palabra está en cache, cada entrada tiene un valid bit y un dirty bit. ¿Cuántos bits adicionales tiene la memoria cache?

    Las direcciones son de $\log_2(4GB) = 32$ *bits.*
    La memoria tiene $8kB/32 = 256$ *sets.*
    Las etiquetas son de $32 - 13 = 19$ *bits. A cada set se añade un bit valid y un bit dirty.*
    En total, los bits adicionales son: $(19 + 2)(256) = 5,376$

15. (2 puntos) Considere un procesador pipeline de 5 etapas (F,D,E,M,W), en el que la etapa de ejecución es variable dependiendo de la operación: Cuatro ciclos para la multiplicación de punto flotante, dos para las sumas de punto flotante, uno para carga/guarda y dos para las operaciones de aritmética de enteros.

    Suponga que los conflictos se manejan con bloqueos del pipeline y que el único atajo disponible ocurre a través del banco de registros.

    (a) Llene el diagrama de tiempos hasta el ciclo 20 para el código que se muestra en la figura.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
ADDU R2,R2,R3	F	D	E	E	M	W														
L.D F0,0(R2)		F	D	D	D	D	E	M	W											
MUL.D F2,F2,F0			F	F	F	F	D	D	D	E	E	E	E	M	W					
L.D F8,1000(R2)						F	F	F	F	D	D	D	D	E	M	W				
MUL.D F4,F4,F8										F	F	F	F	D	D	D	E	E	E	E
ADD.D F6,F2,F4														F	F	F	D	D	D	D
S.D F6,0(R4)																	F	F	F	F
SUBU R4,R4,#4																				
BEQ R2,R4,$L46																				

    (b) Identifique el primer conflicto que bloquea el pipeline, clasifíquelo y describa cuál sería la manera más eficiente de eliminarlo.
    Es un conflicto de dependencia real (RAW): La segunda instrucción necesita el valor de R2, el cual es calculado en la primera instrucción. Lo más conveniente es tener un atajo en la etapa E: la salida de la ALU se retroalimenta para que el valor pueda ser utilizado inmediatamente.

16. (2 puntos) Las etapas de un procesador con pipeline con 5 etapas tienen las siguientes latencias. F: 1 ns; D: 1.5 ns; E: 1 ns; M: 2 ns; W: 1.5 ns. Los registros entre etapas introducen una latencia de 0.1 ns.

    (a) ¿Cuál es el periodo de reloj que controla este pipeline?
    El más lento de las etapas: 2.1 ns.

    (b) Si hay un bloqueo de un ciclo cuada cuatro instrucciones, ¿Cuál es el CPI efectivo?

$5\,ciclos/4\,instrucciones = 1.25\,CPI$

(c) Cuál es la aceleración (*speedup*) obtenida al introducir el pipeline en la arquitectura
Sin pipeline, el tiempo de ejecución por instrucción es $1 + 1.5 + 1 + 2 + 1.5 = 7nS$

$$Sp = \frac{I \times 1 \times 7}{I \times 1.25 \times 2.1} = 2.67$$