Compito del Corso di Architettura degli Elaboratori 1

Anno Accademico 2004/2005

Appello dell' 8 Aprile 2005 - C - Soluzione di alcuni esercizi

Istruzioni

- Scrivere *Nome*, *Cognome* e *Matricola* su **ogni** foglio.
- Scrivere la risposta nello spazio bianco al di sotto della domanda; Non è possibile allegare fogli aggiuntivi, quindi cercate di essere chiari e non prolissi.
- In caso di errori indicate chiaramente quale parte della risposta deve essere considerata; annullate le parti non pertinenti.
- Assicurarsi che non manchi alcun foglio al momento della consegna.

es1

Sia data una memoria segmentata con paginazione in grado di indirizzare 64 segmenti, ognuno comprendente fino a 64K pagine di 4KB. (Si assume indirizzamento al byte). Di quanti bit è costituito un indirizzo logico ?:

a) 32;

b) 34;

c) 24;

d) nessuna delle risposte precedenti è corretta;

Soluzione

La risposta esatta è b). Infatti occorrono 6 bit per i segmenti ($256 = 2^6$), 16 bit per le pagine ($64K = 2^{16}$) e 12 bit per pagina ($4K = 2^{12}$), per un totale di 34 bit.

es2

Sia data la seguente sequenza di istruzioni assembler, dove i dati immediati sono espressi in esadecimale

```
LB   R3, 450(R0)
ADD  R2, R0, R0
LB   R1, 558(R2)
ADDI R2, R2, 5
SUB  R4, R3, R2
ADDI R1, R1, 7
SB   R1, 58(R2)
```

Si consideri la pipeline MIPS a 5 stadi vista a lezione, senza possibilità di data-forwarding, ma con possibilità di scrittura e successiva lettura dei registri in uno stesso ciclo di clock. L'esecuzione completa del codice avviene in un numero di cicli di clock pari a

a) 13

b) 15

c) 9

d) 17

e) in nessuno dei circuiti elencati precedentemente;

Soluzione

La risposta esatta è d). Infatti,

	CICLI CLOCK															
istruzione	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
LB R3, 450(R0)	IF	ID	EX	ME	WB												
ADD R2, R0, R0		IF	ID	EX	ME	WB											
LB R1, 558(R2)			IF	ID	ID	ID	EX	ME	WB								
ADDI R2, R2, 5				IF	IF	IF	ID	EX	ME	WB							
SUB R4, R3, R2							IF	ID	ID	ID	EX	ME	WB				
ADDI R1, R1, 7								IF	IF	IF	ID	EX	ME	WB			
SB R1, 58(R2)											IF	ID	ID	ID	EX	ME	WB

es9

Sia data la seguente sequenza di istruzioni assembler, dove i dati immediati sono espressi in esadecimale ed il registro `R0` contiene il valore 0:

```
LB   R3, 0(R0)     ! load byte da mem[0+[R0]]
ADD  R2, R0, R0    ! R2 = R0 + R0
LB   R1, 8(R2)     ! load byte da mem[8+[R2]]
ADDI R2, R2, 5     ! R2 = R2 + 5
SUB  R4, R3, R2    ! R4 = R3 - R2
ADDI R1, R1, 7     ! R1 = R1 + 7
SB   R1, 8(R2)     ! store byte in mem[8+[R2]]
BGTZ R4, -6        ! PC = PC - 6 se [R4] > 0
                   ! cioe' salta alla istruzione LB   R1, 8(R2)
```

Si assuma la presenza di due cache, una dati ed una istruzioni. La cache dati, in particolare, è di ampiezza 8B, con dimensione di blocco 2B, inizialmente vuota, ed associazione 2-way (con politica di rimpiazzo LRU e politica di scrittura write-back). Si assuma che la memeoria abbia il contenuto esadecimale mostrato di seguito (si esprimano gli indirizzi su 12 bit):

Indirizzo	byte	byte	byte	byte
00	06	00	07	02
04	00	00	00	00
08	AE	13	A1	23
0C	A1	42	90	75
10	B9	16	00	00
14	0A	07	03	71

Si mostri come sia il contenuto della cache dati che il contenuto della memoria cambia a causa della esecuzione del codice assembler.

Soluzione

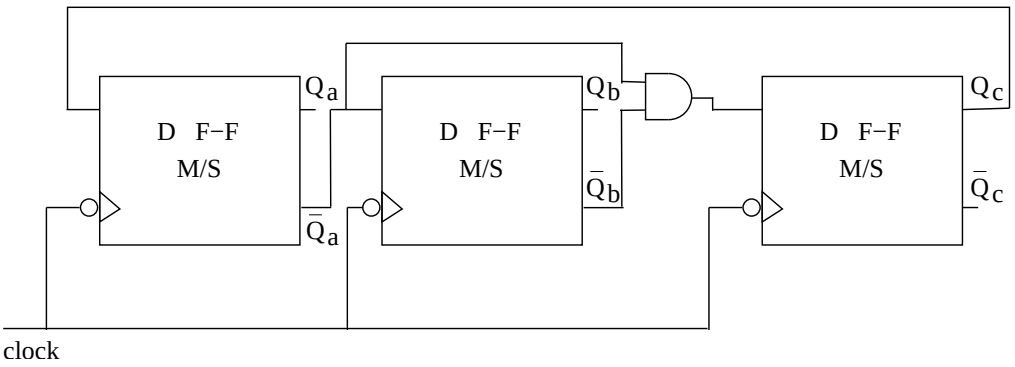
Poiché un blocco è costituito da 2B, e la cache è di 8B si avranno $8/2 = 4$ linee. Essendo l'associatività a due linee (2-way), ci saranno due insiemi (insieme 0 e insieme 1) ognuno di 2 linee. Quindi gli 8 bit di indirizzo saranno suddivisi nel seguente modo: il bit meno significativo individuerà il byte all'interno del blocco, il secondo bit da destra individuerà l'insieme (0 o 1), ed i restanti bit costituiranno il tag. Mostriamo di seguito l'evoluzione del contenuto dei registri, dei riferimenti a memoria, della cache dati (solo quando cambia) e della memoria (solo quando cambia). Per la cache dati, nel caso in cui tutte e due le linee di un insieme (set) siano libere, si sceglie la linea con indirizzo minore per la allocazione (scelta arbitraria: si poteva usare un criterio diverso). Ricordiamo che la politica write-back prescrive l'aggiornamento della memoria principale solo quando il blocco modificato deve essere rimpiazzato in cache. Pertanto, con un (*) vicino ad un blocco ricordiamo che si tratta di un blocco "sporco" (dirty) e che quindi deve essere copiato in memoria principale quando viene rimpiazzato.

codice eseguito	[R1] hex	[R2] hex	[R3] hex	[R4] hex	ind. rif. memoria hex / binario	cache dati: set 0 linea 0 / linea 1 — set 1 linea 2 / linea 3 (t: tag, r: rif.)	modifica memoria mem[ind.] = cont.
LB R3, 0(R0)	?	?	6	?	00 / 00000000	[06|00] t:000000 r:miss	
ADD R2, R0, R0	?	0	6	?		[ ]	
LB R1, 8(R2)	AE	0	6	?	08 / 00001000	[06|00] t:000000 r: [AE|13] t:000010 r: miss	
ADDI R2, R2, 5	AE	5	6	?			
SUB R4, R3, R2	AE	5	6	1			
ADDI R1, R1, 7	B5	5	6	1			
SB R1, 8(R2)	B5	5	6	1	0D / 00001101	[A1|42] t:000011 r:miss (LRU) [AE|13] t:000010 r: [A1|B5]* t:000011 r:write [AE|13] t:000010 r:	
BGTZ R4, -6	B5	5	6	1			
LB R1, 8(R2)	B5	5	6	1	0D / 00001101	[A1|B5]* t:000011 r:hit [AE|13] t:000010 r:	

ADDI R2, R2, 5	B5	A	6	1				
SUB R4, R3, R2	B5	A	6	-4				
ADDI R1, R1, 7	BC	A	6	-4				
SB R1, 8(R2)	BC	A	6	-4	12	[A1|B5]*	[00|00]	
					00010010	t:000011	t:000100	
						r:	r:miss	
						[AE|13]	[ ]	
						t:000010	t:	
						r:	r:	
						[A1|B5]*	[BC|00]*	
						t:000011	t:000100	
						r:	r:write	
						[AE|13]	[ ]	
						t:000010	t:	
						r:	r:	
BGTZ R4, -6	BC	4	6	-4				

es10

Si descriva l'evoluzione temporale dei segnali per la seguente rete sequenziale sincrona, dato lo stato inziale $Q_a = Q_b = Q_c = 0$



Soluzione

Considerando che $D_a(t) = Q_c(t)$, $D_b(t) = \overline{Q}_a(t)$, e $D_c(t) = \overline{Q}_a(t) \wedge \overline{Q}_b(t)$, si ha la seguente evoluzione temporale: