COMPITINO #2 di CALCOLATORI ELETTRONICI 1 del 11-12-02 **MATRICOLA**_____

(versione riveduta da errori di stampa) **COGNOME**_____

NOME_____

1) Descrivere il funzionamento di un sistema a dischi RAID-5, illustrandone pregi e difetti rispetto ad un sistema a dischi RAID-4.

2) Un calcolatore con processore R3000 avente frequenza di clock pari a 2GHz esegue il seguente programma. La cache e' divisa in cache istruzioni di dimensione 64 byte, ad accesso diretto, blocco da 16 parole e cache dati di dimensione di 32 byte, set-associative a due vie, blocco da due parole. Il tempo di accesso alla cache in caso di hit e' pari a 0.5ns e la penalty in caso di miss e' pari a 2ns, per entrambe le cache. Calcolare il tempo di esecuzione di questo programma.

```
        .text 0x3000
        addi  $4,$0,4
        addi  $2,$0,0x1000
        addi  $3,$0,0x1020
        addi  $7,$0,0x2000
L2:     slt   $1,$2,$3
        beq   $1,$0,esci
        lw    $5,0($2)
        add   $2,$2,$4
        add   $6,$4,$5
        sw    $6,0($7)
        add   $7,$7,$4
        add   $7,$7,$4
        j     L2
esci:   nop
```

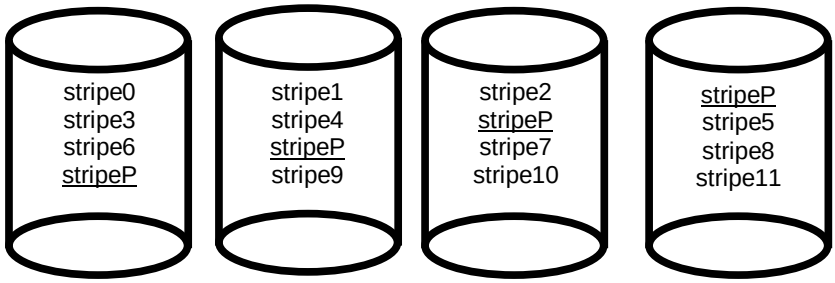
Nota: si assuma che il processore R3000 ammetta la scrittura e la lettura di un registro nello stesso ciclo di clock, che sia possibile sfruttare il cosiddetto "delay-slot" generato dalle istruzioni di salto, che sia possibile decidere il salto nello stadio di decodifica e che siano disabilitati i circuiti per la propagazione (forwarding).

Riepilogo del significato delle istruzioni

Instruction	Example	Meaning	Comments
add	`add $1,$2,$3`	$1 = $2 + $3	3 operands; exception possible
add immediate	`addi $1,$2,100`	$1 = $2 + 100	+ constant; exception possible
load word	`lw $1,100($2)`	$1 = Memory[$2+100]	Data from memory to register
store word	`sw $1,100($2)`	Memory[$2+100] = $1	Data from register to memory
branch on equal	`beq $1,$2,100`	if ($1 == $2) go to PC+4+100	Equal test; PC relative
set on less than	`slt $1,$2,$3`	if ($2 < $3) $1 = 1; else $1 = 0	Compare less than; 2`s complement
no operation	`nop`	execute but do nothing	
jump	`j 10000`	go to 10000	Jump to target address

COMPITINO #2 di CALCOLATORI ELETTRONICI 1 del 11-12-02 **SOLUZIONE**

1)



Sistema RAID-5

Il sistema RAID (Redundant Array of Inexpensive Disks) consente di migliorare il throughput e l'affidabilita' del sottosistema di memorizzazione stabile di un calcolatore.
Consideriamo un sistema RAID-5 costituito, ad esempio, da 4 dischi. Il dato viene diviso in tre fette (stripe) e viene aggiunta una fetta per verificarne la parita' (stripeP). Questo avviene anche nel sistema RAID-4. Nel caso del RAID-4 la stripe della parita' viene pero' scritta sempre sullo stesso disco, mentre nel caso di RAID-5 le stripe relative a dati diversi hanno una probabilita' solo 1/n (n numero di dischi) di capitare sullo stesso disco. La distribuzione delle parita' e' essenziale per poter parallelizzare il piu' possibile le operazioni di lettura e scrittura e evitare conflitti nell'accesso ai dati distribuiti sui dischi disponibili.

2) Una volta disegnato il diagramma di funzionamento della pipeline (F=Fetch, D=Decode, X=Execute, M=data Memory access, W=Write-back), osservando in particolare che, a causa del delay slot dopo `beq` e `j`, vengono comunque eseguite le istruzioni immediatamente successive alle istruzioni di salto e diramazione, e rispettando gli stalli dovuti alle dipendenze fra i registri, si ricava che i cicli necessari per eseguire il programma trascurando gli stalli dovuti agli accessi in memoria sono:

C=10+8*17+5=151

Il fattore 8 e' dovuto alla ripetizione della porzione centrale del codice che si trova all'interno del ciclo etichettato L2.

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
addi	$4,$0,4	F	D	X	M	W																											
addi	$2,$0,0x1000		F	D	X	M	W																										
addi	$3,$0,0x1020			F	D	X	M	W																									
addi	$7,$0,0x2000				F	D	X	M	W																								
slt	$1,$2,$3					F	---	D	X	M	W																						
beq	$1,$0,esci							F	---	---	D	X	M	W																			
lw	$5,0($2)										F	D	X	M	W																		
add	$2,$2,$4											F	D	X	M	W																
add	$6,$4,$5												F	---	D	X	M	W														
sw	$6,0($7)														F	---	---	D	X	M	W												
add	$7,$7,$4																	F	D	X	M	W											
add	$7,$7,$4																		F	---	---	D	X	M	W								
	L2																					F	D	X	M	W							
nop																							F	D	X	M	W						
slt	$1,$2,$3																							F	D	X	M	W					
beq	$1,$0,esci																								F	---	---	D	X	M	W		
lw	$5,0($2)																											F	D	X	M	W	
nop																													F	D	X	M	W

10 cicli — 17 cicli — 5 cicli

Valutazione degli stalli dovuti alla memoria:

i) per quanto riguarda la cache istruzioni, si osserva che il programma ha 14 istruzioni (a 32 bit) mentre il blocco di cache puo'contenere 16 parole (a 32 bit), quindi il programma dato "sta tutto in un solo blocco" di cache e di conseguenza ci sara' un solo miss iniziale per il caricamento di tale blocco;

ii) per quanto riguarda la cache dati, dal precedente diagramma si puo' ricavare innanzitutto la traccia dei riferimenti ai dati. Sia X il generico riferimento, A=associativita'=2, B=dimensione del blocco=8, C=capacita' della cache=32.

COMPITINO #2 di CALCOLATORI ELETTRONICI 1 del 11-12-02 **SOLUZIONE**

Si ricava S=C/B/A=# di set della cache=32/8/2=2, X1=X/B, XS=X1%S, XT=X1/S:

X	X1	XS	XT	Hit/Miss
4096	512	0	256	**M**
8192	1024	0	512	**M**
4100	512	0	256	**H**
8200	1025	1	512	**M**
4104	513	1	256	**M**
8208	1026	0	513	**M**
4108	513	1	256	**H**
8216	1027	1	513	**M**
4112	514	0	257	**M**
8224	1028	0	514	**M**
4116	514	0	257	**H**
8232	1029	1	514	**M**
4120	515	1	257	**M**
8240	1030	0	515	**M**
4124	515	1	257	**H**
8248	1031	1	515	**M**
4128	516	0	258	**M**

Avendo 13 miss su 17 riferimenti il miss rate della cache dati risulta m_D=13/17.

Si ricava infine:

$$T_{CPU} = \frac{N_{CPU}}{f_C} \cdot \overline{CPI} = \frac{N_{CPU}}{f_C} \cdot \left(\overline{CPI}_{ideal} + \overline{CPI}_{mem-i} + \overline{CPI}_{mem-d} \right)$$

$$CPI_{ideal} = \frac{C_{CPU}}{N_{CPU}} = \frac{151}{N_{CPU}}$$

$$CPI_{mem-i} = m_I \cdot C_{pen} = \frac{1}{N_{CPU}} \cdot 4 = \frac{4}{N_{CPU}}$$

$$CPI_{mem-d} = REFINST_D \cdot m_D \cdot C_{pen} = \frac{17}{N_{CPU}} \cdot \frac{13}{17} \cdot 4 = \frac{52}{N_{CPU}}$$

$$T_{CPU} = \frac{N_{CPU}}{f_C} \cdot \left(\frac{151}{N_{CPU}} + \frac{4}{N_{CPU}} + \frac{52}{N_{CPU}} \right) = \frac{207}{2 \cdot 10^9} = 103.5 ns$$