

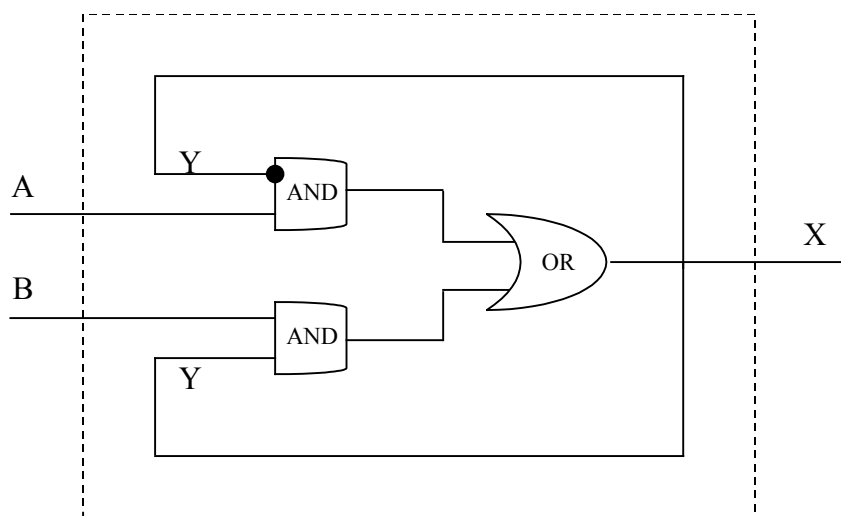
**SOLUZIONI DELLA PROVA SCRITTA DEL CORSO DI
CALCOLATORI ELETTRONICI
NUOVO E VECCHIO ORDINAMENTO DIDATTICO
14 Gennaio 2004**

MOTIVARE IN MANIERA CHIARA LE SOLUZIONI PROPOSTE A CIASCUNO DEGLI ESERCIZI SVOLTI

ESERCIZIO 1 (8 punti)

(a) (5 punti) A partire dallo schema circuitale in figura, disegnare la tabella delle transizioni e indicare qual è la funzionalità svolta dal circuito.

N.B. Il pallino davanti alla porta AND significa che l'ingresso è negato. Y non è altro che l'uscita X riportata in ingresso alle porte AND.



(b) (3 punti) Esprimere in modo chiaro e sintetico la differenza tra un latch ed un flip flop.

Soluzione.

(a) Dall'esame del circuito si ha:

$$X = \bar{Y}A + YB$$

A	B	Y	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Poiché $Y=X$, si ricava dalla formula che la rete proposta implementa un latch JK, dove $J=A$, $K=B'$ (l'apice indica la negazione), e il bit solitamente usato per indicare lo stato in uscita corrisponde a X.

(b) Vedi lucidi del corso.

ESERCIZIO 2 (solo NO: 8 punti)

- (a) (6 punti) Scrivere una funzione Assembler MIPS che ricevendo in ingresso due valori interi positivi x e y , produca in uscita il valore z pari a 0 se x è divisibile per y , 1 altrimenti. Si supponga che $\$4 \leftarrow x$, $\$5 \leftarrow y$, $\$6 \leftarrow z$. Ottimizzare la scrittura del codice riducendo al minimo il numero di salti (condizionati e/o incondizionati).
- (b) (2 punti) Specificare il tipo di indirizzamento richiesto dal MIPS per le istruzioni `lw` e `sw`.

Soluzione.

(a)

divisibile:

```

slt $6, $5, $4      # if(Y<X) then Z=1 else Z=0;
sub $4, $4, $5      # X = X - Y
bne $6, $0, divisibile # if(Z) then goto divisibile
add $4, $4, $5      # X = X + Y
slt $6, $0, $4      # if (X>0) then Z=1 else Z=0;
jr $31              # restore and exit

```

(b) Vedi lucidi del corso.

ESERCIZIO 2 (solo VO: 6 punti)

Si supponga di disporre di tre architetture: a pila, a uno e a due indirizzi. Per ognuna di queste si abbiano le seguenti istruzioni:

A pila		A un indirizzo		A due indirizzi	
Istruzione	Semantica	Istruzione	Semantica	Istruzione	Semantica
PUSH X	$M[X] \rightarrow \text{push}$	STORE X	$\text{ACC} \rightarrow M[X]$	MOV X1, X2	$M[X1] \rightarrow M[X2]$
POP X	$\text{pop} \rightarrow M[X]$	LOAD X	$M[X] \rightarrow \text{ACC}$	ADD X1, X2	$M[X1] + M[X2] \rightarrow M[X2]$
ADD	$\text{pop} + \text{pop} \rightarrow \text{push}$	ADD X	$\text{ACC} + M[X] \rightarrow \text{ACC}$	DIV X1, X2	$M[X1] / M[X2] \rightarrow M[X2]$
DIV	$\text{pop1} / \text{pop2} \rightarrow \text{push}$	DIV X	$\text{ACC} / M[X] \rightarrow \text{ACC}$	MUL X1, X2	$M[X1] * M[X2] \rightarrow M[X2]$
MUL	$\text{pop} * \text{pop} \rightarrow \text{push}$	MUL X	$\text{ACC} * M[X] \rightarrow \text{ACC}$		

ACC è il registro accumulatore della macchina a un indirizzo.

$M\{X\}$ indica il dato nella locazione di memoria X .

Si scriva, per ognuna delle tre macchine, la sequenza delle istruzioni necessarie per realizzare la seguente operazione (rispettando le precedenze delle operazioni):

$$Z = (A + B) / (C + D) * E$$

Le lettere indicano le locazioni di memoria dove si trovano i dati.

Nella macchina a due indirizzi si faccia uso di una ulteriore locazione P dove introdurre i risultati parziali, pena la perdita dei dati iniziali nelle locazioni date.

Soluzione.

A pila	A un indirizzo	A due indirizzi
PUSH C	LOAD C	MOV A, P
PUSH D	ADD E	ADD B, P
ADD	STORE Z	MOV C, Z
PUSH A	LOAD A	ADD D, Z
PUSH B	ADD B	DIV P, Z
ADD	DIV Z	MUL E, Z
DIV	MUL E	
PUSH E	STORE Z	
MUL		
POP Z		

ESERCIZIO 3 (NO: 9 punti – VO: 8 punti)

Si consideri un calcolatore che dispone di una memoria principale di 256 Mbyte e di una memoria cache di 128 Kbyte. E' possibile accedere al singolo byte e la memoria è suddivisa in blocchi da 64 byte.

- 1) (NO: 4 punti – VO: 2 punti) Spiegare, precisando il significato e la funzione dei diversi campi, come vengono interpretati gli indirizzi logici per recuperare l'informazione contenuta nella cache nel caso venga usata la modalità di indirizzamento:
 - (a) Diretto
 - (b) "associativo su insiemi", e ciascun insieme contenga otto blocchi
- 2) (NO: 5 punti – VO: 3 punti) Si consideri la cache di cui alla domanda precedente. Ipotizzare che il processore acceda in sequenza ai byte dall'indirizzo 0000000 a 00000FF e da AEC0000 a AEC00FF in questo ordine, e ripeta questa sequenza di accesso per 10 volte consecutive. Si ipotizzi inoltre che la cache sia inizialmente vuota. Calcolare l'hit ratio sia nel caso di indirizzamento diretto che nel caso di indirizzamento "associativo su insiemi" spiegata nel punto precedente.
- 3) (solo VO: 3 punti) Mostrare lo stato finale della memoria nei due casi (diretto e associativo su insiemi), specificando quali blocchi e quali insiemi vengono occupati.

Soluzione

1) Per indirizzare 256 Mbyte occorre un indirizzo di almeno 28 bit. Per indirizzare il singolo byte all'interno di un blocco occorrono 6 bit ($64 = 2^6$), che coincidono con i 6 bit meno significativi dell'indirizzo di memoria primaria. I restanti 22 bit costituiscono l'indirizzo del "block frame". Per indirizzare la cache, il "block frame" viene interpretato diversamente a seconda che l'indirizzamento sia di tipo "diretto" o "associativo su insiemi".

(a) Indirizzamento diretto. In questo caso devo poter indirizzare ciascuno dei 32K blocchi contenuti nella cache ($128\text{Kbyte}/(64\text{byte}/\text{blocco})$). Occorrono 11 bit che coincidono con i 11 bit meno significativi del "block frame" Pertanto i 28 bit di indirizzo della memoria primaria vengono interpretati come:

tag	Cache index	Offset
11 bit	11 bit	6 bit

(b) Indirizzamento "associativo su insiemi". In questo caso devo poter indirizzare ciascuno dei 256 insiemi in cui sono suddivisi i blocchi contenuti nella cache. Occorrono 8 bit che coincidono con gli 8 bit meno significativi del "block frame". Pertanto i 28 bit di indirizzo della memoria primaria vengono interpretati come:

tag	cache index	Offset
14 bit	8 bit	6 bit

2) La memoria è divisa in blocchi di 64 byte ciascuno in modo che la richiesta di un dato non presente in cache causa il trasferimento del blocco a cui appartiene il dato richiesto dalla memoria principale alla cache. Nel caso proposto i dati richiesti sono così suddivisi (N.B. per semplicità i valori di index sono riportati in formato decimale, mentre gli indirizzi in esadecimale):

indirizzi 0000000 ÷ 000003F ⇒ index 0; indirizzi 0000040 ÷ 000007F ⇒ index 1; ... ; 0000080 ÷ 00000BF ⇒ index 3; AEC0000 a AEC003F ⇒ index 0; ... ; AEC0040 a AEC007F ⇒ index 3.

(N.B. in questa sequenza l'index è identico nel caso di indirizzamento diretto e associativo su insiemi).

La prima volta che viene richiesta la parola 0000000 avremo un "cache miss" che provoca il caricamento del blocco 0 nella linea di cache di indirizzo 0. Le successive

richieste dei dati di indirizzo 0000001 ÷ 000000F vengono quindi soddisfatte dalla cache ("cache hit"). Analogamente avviene per tutti i blocchi fino al blocco 3, che vengono allocati in linee consecutive della cache fino alla 3. Quando viene richiesta la parola AEC0000 (index 0), nel caso di indirizzamento diretto questa sovrascrive il blocco 0000000 ÷ 000000F e così via per tutte le richieste consecutive.

Nel caso di indirizzamento associativo su insiemi a due vie uno stesso index corrisponde a un insieme di due blocchi. Pertanto le parole da AEC0000 a AEC00FF non sovrascrivono quelle precedentemente inserite, ma vengono memorizzate nel secondo blocco disponibile in ciascun insieme memorizzato. Nei cicli successivi al primo il processore richiede nuovamente tutti i dati, a partire da 0. Nel caso di indirizzamento diretto avremo un miss per il caricamento della prima parola di ciascun blocco e 15 hit per le parole dello stesso blocco, per tutti gli 8 blocchi. Nel caso di indirizzamento associativo su insiemi si hanno solo hit perché tutti i blocchi sono presenti in cache. In sintesi:

Nel caso di indirizzamento diretto avremo per ciascun ciclo $8 \cdot 63$ hit. L'hit ratio H della gerarchia di memoria risulta pertanto pari a:

$$H = \text{cachehit} / \text{richieste totali} = (10 \cdot 8 \cdot 63) / (10 \cdot 8 \cdot 64) = 0.9844$$

Nel caso di indirizzamento associativo su insiemi a otto vie avremo nel primo ciclo $8 \cdot 63$ hit, mentre nei 9 cicli successivi avremo $8 \cdot 64$ hit. L'hit ratio H della gerarchia di memoria risulta pertanto pari a:

$$H = \text{cachehit} / \text{richieste totali} = (8 \cdot 63 + 9 \cdot 8 \cdot 64) / 10 \cdot 8 \cdot 64 = 0.9984$$

4) Lo stato finale della memoria è il seguente.

METODO DIRETTO

BLOCCO 0	BLOCCO 1	BLOCCO 2	BLOCCO 3
AEC0000	AEC0040	AEC0080	AEC00C0
AEC0001	AEC0041	AEC0081	AEC00C1
...
AEC003F	AEC007F	AEC00BF	AEC00FF

METODO ASSOCIATIVO SU INSIEMI A OTTO VIE

	INSIEME 0	INSIEME 1	INSIEME 2	INSIEME 3	
BLOCCO 0	0000000 0000001 ... 000003F	0000040 0000041 ... 000007F	0000080 0000081 ... 00000BF	00000C0 00000C1 ... 00000FF	BLOCCO 0
BLOCCO 1	AEC0000 AEC0001 ... AEC003F	AEC0040 AEC0041 ... AEC007F	AEC0080 AEC0081 ... AEC00BF	AEC00C0 AEC00C1 ... AEC00FF	BLOCCO 1
BLOCCO i					BLOCCO i
BLOCCO 7					BLOCCO 7

ESERCIZIO 4 (NO: 8 punti – VO: 6 punti)

Si consideri la seguente lista di processi. Il Sistema Operativo usa uno scheduling di tipo SJF mono programmato.

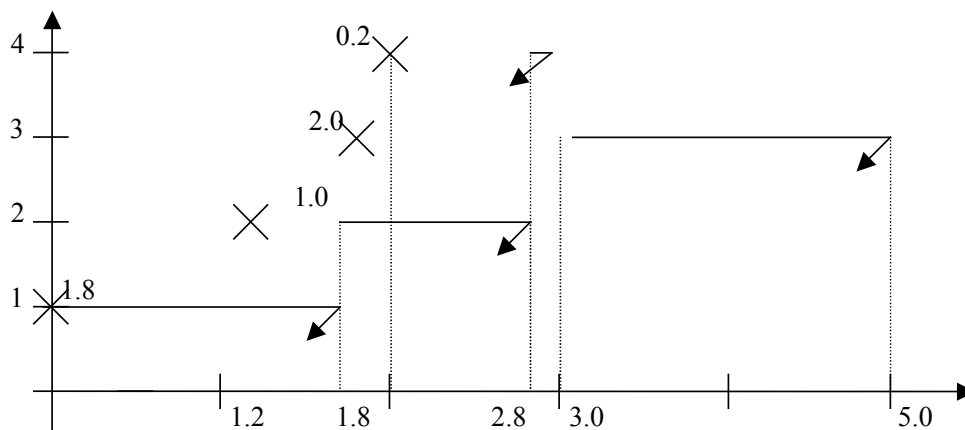
Processo	Arrivo	CPU
1	0.0	1.8
2	1.2	1.0
3	1.9	2.0
4	2.0	0.2

Considerando una memoria complessiva di 9 pagine:

- (NO: 3 punti – VO: 2 punti) Mostrare, utilizzando il metodo grafico, la sequenza di esecuzione dei processi.
- (2 punti) Calcolare il tempo di *turnaround* medio e il tempo di *turnaround* pesato medio.
- (NO: 3 punti – VO: 2 punti) Illustrare in modo chiaro e sintetico la strategia di allocazione basata su paginazione.

Soluzione.

1.



2.

Processo	Arrivo	Fine	Turnaround	WT
1	0.0	1.8	1.8	1.0
2	1.2	2.8	1.6	1.6
3	1.9	5.0	3.1	1.6
4	2.0	3.0	1.0	5.0
Media			1.9	2.3

3. Vedi lucidi del corso.

ESERCIZIO 5 (solo VO: 5 punti)

Descrivere le principali differenze tra le architetture CISC e RISC, illustrando lo schema di principio delle due architetture.

Soluzione.

Vedi lucidi del corso.