

SOLUZIONI DELLA PROVA SCRITTA DEL CORSO DI
CALCOLATORI ELETTRONICI
NUOVO E VECCHIO ORDINAMENTO DIDATTICO
12 Luglio 2002

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

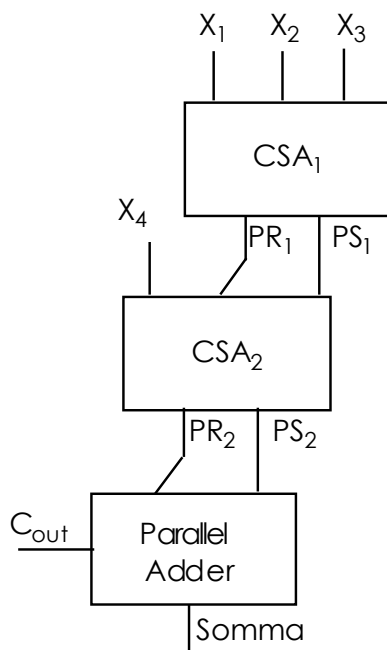
ESERCIZIO 1 (NO: 10 punti – VO: 8 punti)

Si vogliano sommare i seguenti quattro numeri a 7 bit: $X_1 = 1001010$, $X_2 = 0100010$, $X_3 = 0010111$, $X_4 = 0111000$.

1. Disegnare lo schema che permette di eseguire tale somma usando due addizionatori del tipo "Carry Save Adder" ed un "Parallel Adder" finale. Gli addizionatori "Carry Save Adder" lavorano su tre operandi. (2 punti)
2. Precisare il valore assunto dalla Pseudosomma e dal Pseudoriporto all'uscita del primo e del secondo "Carry Save Adder" e le operazioni eseguite per ottenere la somma finale. (NO: 4 punti – VO: 3 punti)
3. Indicato con N il numero di bit di ciascun addendo, descrivere come sia possibile implementare ciascun Carry Save Adder con dei Full Adder. (Suggerimento: pensare a come vengono calcolati pseudosomma e pseudoriporto). (NO: 2 punti – VO: 1 punto)
4. Indicato con d il ritardo dovuto a ciascun Full Adder, e indicato con N il numero di bit di ogni addendo, calcolare il vantaggio che si ottiene usando addizionatori del tipo carry-save al posto di addizionatori paralleli in termini di rapporto tra i ritardi nel primo (CSA+PA) e nel secondo caso (solamente PA), al tendere di N a infinito. (2 punti)

Soluzione

1-2. Schema:



L'addizionatore "Carry Save Adder" fornisce in uscita due parole a 7 bit: la Pseudosomma e il Pseudoriporto. La Pseudosomma (PS) rappresenta il risultato della somma dei tre addendi senza considerare i riporti, mentre il Pseudoriporto (PR) rappresenta solo i riporti relativi a ciascuno stadio. Il risultato completo lo si ottiene sommando $PS + 2 \cdot PR$. Un possibile schema per eseguire la somma richiesta è riportato in figura, dove con un tratto obliquo si è indicato lo "scorrimento" verso sinistra (moltiplicazione per 2) necessario per sommare il PR al PS.

Somma di $X_1 + X_2 + X_3$:

$$PS_1 = 1111111, PR_1 = 0000010; 2 \cdot PR_1 = 0000100$$

Somma di $X_4 + PS_1 + 2 \cdot PR_1$:

$$PS_2 = 1000011, PR_2 = 0111100; 2 \cdot PR_2 = 1111000$$

$$\text{Somma finale} = PS_2 + 2 \cdot PR_2 = 0111011, C_{out} = 1$$

3. Ogni CSA è costituito da N Full Adder in parallelo. Ogni full adder presenta in ingresso tre bit, uno per ogni addendo, in uscita fornisce la somma dei tre bit e il riporto. L'insieme degli N bit all'uscita di ogni Full Adder relativi alla somma costituisce la PseudoSomma, e l'insieme degli N bit relativi al riporto costituisce lo PseudoRiporto.

4. Il ritardo dovuto a ciascun Carry Save Adder è pari a quello di ogni Full Adder componente, perché i FA sono disposti in parallelo. Il ritardo invece prodotto dal Parallel Adder è pari Nd . Quindi il ritardo complessivo del sistema è $d + d + Nd = (N+2)d$.

Per sommare i quattro addendi con soli addizionatori paralleli, dovremmo collocarne tre in serie, per un ritardo complessivo pari a $3Nd$.

Il rapporto tra i due ritardi è pari a $(N+2)/3N$, e come si vede tende a $1/3$ al crescere del numero dei bit per ciascun operando. Ciò significa che con tale sistema siamo in grado di risparmiare fino a $2/3$ del tempo impiegato da dei semplici addizionatori paralleli per effettuare la somma.

ESERCIZIO 2 (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) (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: 4 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) (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 ÷ 000000F ⇒ index 0; indirizzi 0000010 ÷ 000001F ⇒ index 1; ... ; 00003F0 ÷ 0000FF ⇒ index 3; AEC0000 a AEC000F ⇒ index 0; ... ; AEC03F0 a AEC00FF ⇒ 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*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*63 hit, mentre nei 9 cicli successivi avremo 8*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 3 (NO: 7 punti – VO: 6 punti)

Si consideri un disco rigido con le seguenti caratteristiche: velocità di rotazione = 5400 giri/min, tempo medio di posizionamento = 5 ms, 34 settori per traccia di 512 byte ciascuno.

- Calcolare il tempo medio di trasferimento di un file da 8 kbyte considerando:
 - il caso migliore (NO: 3 punti – VO: 2 punti);
 - il caso medio (NO: 4 punti – VO: 2 punti).
- Supporre che il file debba essere trasferito in memoria principale, attraverso un modulo DMA. Indicare e descrivere quale strategia è seguita per il trasferimento. (solo VO: 2 punti)

Soluzione

$$T_{rot} = 60/5400 \text{ sec} = 11.11 \text{ msec.}$$

$$T_{lat} = T_{rot}/2 = 5.555 \text{ msec.}$$

$$T_{ls} = T_{rot} / 34 = 0.327 \text{ msec. (tempo di lettura di un settore)}$$

$$T_{pos} = 5 \text{ msec.}$$

$$\text{Numero di settori richiesti dal file } N = 8\text{kbyte} / 512\text{byte} = 16 \text{ settori.}$$

- Caso migliore: il file è posizionato su settori consecutivi della stessa traccia e la testina è posizionata all'inizio del primo settore. Dato che il file può essere memorizzato in una sola traccia:

$$T_{lett} = N * T_{ls} = 5.232 \text{ msec.}$$

Caso medio: il file è posizionato su settori collocati in tracce diverse e la testina si trova in un punto qualsiasi del disco.

$$T_{lett} = N * (T_{lat} + T_{pos} + T_{ls}) = 16 * 10.882 = 174.112 \text{ msec}$$

- b) La strategia utilizzata è senz'altro la DMA "block transfer", per evitare tempi d'attesa sul disco. In questo caso infatti la periferica diventa master del bus e il trasferimento interrompe le operazioni della CPU fino a che non viene completato.

ESERCIZIO 3 (NO: 7 punti – VO: 6 punti)

La memoria di un calcolatore è gestita con una tecnica di 'paginazione su richiesta'. Si consideri la seguente richiesta di pagine:

3 4 6 4 5 1 4 2 3 3 4 1 4 8 1 2 1 4 7 4 4

Se la memoria contiene complessivamente quattro pagine calcolare l'hit ratio nei seguenti due casi:

1. strategia di rimpiazzamento delle pagine FIFO. (3 punti)
2. strategia di rimpiazzamento delle pagine LRU. (NO: 4 punti – VO: 3 punti)

Soluzione:

Page trace nel caso di strategia di rimpiazzamento delle pagine FIFO (x = hit)

Tempo	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Richieste	3	4	6	4	5	1	4	2	3	3	4	1	4	8	1	2	1	4	7	4	4
Pagine	3	4	6	6	5	1	1	2	3	3	4	4	4	8	1	2	2	2	7	4	4
		3	4	4	6	5	5	1	2	2	3	3	3	4	8	1	1	1	2	7	7
			3	3	4	6	6	5	1	1	2	2	2	3	4	8	8	8	1	2	2
					3	4	4	6	5	5	1	1	1	2	3	4	4	4	8	1	1
Hit				X			X			X		X	X				X	X			X

Hit ratio = 8/21

Page trace nel caso di strategia di rimpiazzamento delle pagine LRU (x = hit)

Tempo	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Richieste	3	4	6	4	5	1	4	2	3	3	4	1	4	8	1	2	1	4	7	4	4
Pagine	3	4	6	4	5	1	4	2	3	3	4	1	4	8	1	2	1	4	7	4	4
		3	4	6	4	5	1	4	2	2	3	4	1	4	8	1	2	1	4	7	7
			3	3	6	4	5	1	4	4	2	3	3	1	4	8	8	2	1	1	1
					3	6	6	5	1	1	1	2	2	3	3	4	4	8	2	2	2
Hit				X			X			X	X	X	X		X		X	X		X	X

Hit ratio = 11/21

ESERCIZIO 5 (5 punti)

Illustrare sinteticamente le caratteristiche e il formato tipico di una istruzione RISC,

Soluzione

Vedi le dispense del corso.