

PROVA SCRITTA DEL CORSO DI
CALCOLATORI ELETTRONICI
NUOVO ORDINAMENTO DIDATTICO
18 Giugno 2009

NOME:

COGNOME:

MATRICOLA:

ESERCIZIO 1 (9 punti)

1. (3 punti) Spiegare in modo chiaro e sintetico le caratteristiche di un Carry-Save Adder, e in quale caso, e perché, presenta vantaggi rispetto al Parallel Adder.
2. (4 punti) Si vogliano sommare i seguenti quattro numeri a 8 bit: $X_1 = 01101010$, $X_2 = 10100100$, $X_3 = 00101010$, $X_4 = 00010010$. 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. 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.
3. (2 punti) Indicando genericamente con d il tempo di ritardo di una rete logica a 2 o 3 livelli (ad esempio, il tempo impiegato da un full adder prima che il bit di riporto in uscita sia valido), calcolare il tempo di ritardo del sommatore implementato al punto 2, esprimendolo come multiplo del tempo di ritardo d .

ESERCIZIO 2 (8 punti)

1. I trasferimenti di parole a /dalla memoria di un calcolatore sono codificate utilizzando il codice di Hamming. Si consideri la stringa in ricezione di 12 bit 000110101111, già codificata con il codice di Hamming (il bit meno significativo è a sinistra). Spiegando bene ogni passo del ragionamento:
 - a. (1 punto) calcolare il numero di bit della stringa di origine;
 - b. (3 punti) decodificare la stringa data, rilevando eventuali errori presenti in essa.
2. (4 punti) Un disco presenta le seguenti caratteristiche: 7200 giri/min, 100 settori per traccia, tempo di posizionamento medio 4 ms, 32 B per settore. Calcolare il tempo medio di lettura di un blocco di 1 KB da disco, nell'ipotesi che la testina si trovi su un settore e in una traccia qualunque all'istante iniziale e che i settori del blocco siano situati ciascuno in tracce diverse.

ESERCIZIO 3 (9 punti)

Si scriva il codice Assembler MIPS di una funzione che, dati tre vettori di N interi u , v , w , scriva nella posizione $w(i)$ il valore $u(i)+v(i)$, se $u(i)<v(i)$, e il valore $u(i)-v(i)$, altrimenti. Si consideri che gli indirizzi iniziali dei vettori u , v , w siano memorizzati in $\$4$, $\$5$, $\$6$, rispettivamente, e che N sia memorizzato in $\$7$.

In altri termini, il codice MIPS può implementare la seguente funzione C:

```
void elabora(int *u, int *v, int *w, int N)
{
    int i;
    for(i=0; i<N; i++)
        if(u(i)<v(i))
            w(i)=u(i)+v(i);
        else
            w(i)=u(i)-v(i);
}
```

ESERCIZIO 4 (7 punti)

1. (5 punti) Un calcolatore ha un sistema di memoria virtuale a tre livelli costituita da: cache, memoria primaria e disco. La lettura di una parola che si trova già memorizzata nella cache richiede 15 ns. La lettura di una parola dalla memoria primaria e il suo trasferimento in cache richiedono complessivamente 40 ns. La lettura di una parola dal disco e il suo trasferimento in memoria primaria richiedono complessivamente 10 ms. La probabilità che una parola si trovi già in cache è pari a 0.95, mentre la probabilità che una parola si trovi in memoria primaria dato che non è presente nella cache è pari a 0.6. Calcolare il tempo medio di accesso al sistema di memoria.
2. (2 punti) Enunciare e spiegare in modo chiaro e sintetico il principio di località.

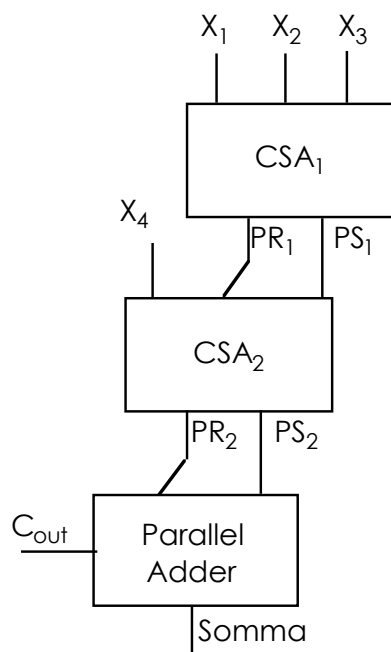
ESERCIZIO 1

Soluzione

1. Vedi dispense del corso.

2.

Schema:



L'addizionatore "Carry Save Adder" fornisce in uscita due parole a 8 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 $PS + 2 PR$. 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 $X_1 + X_2 + X_3$:

$$PS_1 = 11100100, PR_1 = 00101010; 2 PR_1 = 01010100$$

Somma di $X_4 + PS_1 + 2 PR_1$:

$$PS_2 = 10100010, PR_2 = 01010100; 2 PR_2 = 10101000$$

$$\text{Somma finale} = PS_2 + 2 PR_2 = 01001010, C_{out} = 1$$

Vantaggio del carry save adder rispetto ad addizionatore parallelo: per effettuare la somma di 4 addendi usando solo parallel adder sarebbero necessari 3 parallel adder, dunque il vantaggio non è nel numero di componenti. Il vantaggio risiede nella maggior velocità dell'operazione: gli addizionatori del tipo carry save non hanno i ritardi dovuti alla propagazione del riporto, dunque il risultato si ottiene dopo un ritardo corrispondente a una rete a due o tre livelli, a seconda della rete usata per calcolare somma e riporto. L'unico ritardo dovuto alla propagazione del riporto è presente solo nell'ultimo stadio. Se indichiamo con d il ritardo per il calcolo di somma e riporto per un full-adder, un parallel adder a 8 bit produce il risultato con un ritardo pari a $8d$, mentre un carry save adder produce il risultato in termini di PS e PR con un ritardo pari a d . Nel caso in esame (somma di 4 addendi con 2 addizionatori carry-save e un addizionatore parallelo) avremo un ritardo pari a $d + d + 8d = 10d$, mentre usando 3 parallel adder il ritardo nella produzione del risultato sarebbe pari a $3 \cdot 8d = 24d$.

ESERCIZIO 2

Soluzione

1.

a. Deve venire rispettata la condizione:

$$2^K \geq N + K + 1 \quad (1),$$

dove K è il numero di bit di controllo inseriti e N è il numero di bit della stringa di origine. Essendo $N+K=12$, si ha $K=4$ e $N=8$.

b. Nella codifica di Hamming, la sequenza in ingresso presenta la seguente struttura:

c_0	c_1	b_0	c_2	b_1	b_2	b_3	c_3	b_4	b_5	b_6	b_7
0	0	0	1	1	0	1	0	1	1	1	1

Dove $c_0...c_3$ sono i quattro bit costituenti il vettore di controllo, e $b_0...b_7$ gli otto bit trasmessi. I corrispondenti bit di errore si ottengono così:

$$e_0 = c_0 \oplus b_0 \oplus b_1 \oplus b_3 \oplus b_4 \oplus b_6 = 0 \oplus 0 \oplus 1 \oplus 1 \oplus 1 \oplus 1 = 0$$

$$e_1 = c_1 \oplus b_0 \oplus b_2 \oplus b_3 \oplus b_5 \oplus b_6 = 0 \oplus 0 \oplus 0 \oplus 1 \oplus 1 \oplus 1 = 1$$

$$e_2 = c_2 \oplus b_1 \oplus b_2 \oplus b_3 \oplus b_7 = 1 \oplus 1 \oplus 0 \oplus 1 \oplus 1 = 0$$

$$e_3 = c_3 \oplus b_4 \oplus b_5 \oplus b_6 \oplus b_7 = 0 \oplus 1 \oplus 1 \oplus 1 \oplus 1 = 0$$

Poiché il vettore risultante 0010 non è nullo, vi è un errore nella stringa di 12 bit e precisamente nella posizione indicata dal vettore di errore tradotto in notazione decimale. Il bit sbagliato nella stringa codificata è quindi il secondo (c_1), che essendo però un bit di controllo non influisce sulla stringa di origine, che può essere dunque correttamente estratta, a partire dal bit meno significativo: 01011111.

2.

$$TROT = 60 / 7200 = 0.0083 \text{ secondi}$$

$$TLAT = TROT / 2 = 0.00415 \text{ secondi (tempo di latenza)}$$

$$Tlett = TROT / 100 = 0.0833 \text{ ms (tempo di lettura di un settore)}$$

$$Tpos = 4 \text{ ms.}$$

$$\text{Numero di settori richiesti per il blocco da 1 KB: } 1024B/32B = 32.$$

Tempo di lettura del blocco da 1KB:

$$= 32 * (TLAT + TPOS + Tlett) = 32 * (4.15 + 4.00 + 0.0833) = 263.47 \text{ msec}$$

ESERCIZIO 3

Soluzione

$\$8 \leftarrow i; \$9 \leftarrow u(i) < v(i)$

$\$10 \leftarrow u(i); \$11 \leftarrow v(i); \leftarrow \$12 \leftarrow w(i)$

```
elabora:    addi $29, $29, -20
            sw $8, 0($29)
            sw $9, 4($29)
            sw $10, 8($29)
            sw $11, 12($29)
            sw $12, 16($29)
            move $8, $0
```

```
for:        beq $8, $7, exit
            lw $10, 0($4)
            lw $11, 0($5)
            slt $9, $10, $11
            bne $9, $0, sum
            sub $12, $10, $11
```

```
continue:   sw $12, 0($6)
            addi $8, $8, 1
            addi $4, $4, 4
            addi $5, $5, 4
            addi $6, $6, 4
            j for
```

```
exit:       lw $8, 0($29)
            lw $9, 4($29)
            lw $10, 8($29)
            lw $11, 12($29)
            lw $12, 16($29)
            addi $29, $29, 20
            jr $31
```

```
sum:        add $12, $10, $11
            j continue
```

ESERCIZIO 4

Soluzione

1. Si devono considerare tre possibili situazioni:

Posizione della parola richiesta	Probabilità	Tempo di accesso totale
In cache	0.95	15 ns
Non in cache ma in memoria primaria	$(1 - 0.95) * 0.6 = (0.05) * 0.6 = 0.03$	$40\text{ns} + 15\text{ ns} = 55\text{ ns}$
Non in cache né in memoria primaria	$(1 - 0.95) * (1 - 0.6) = 0.05 * 0.4 = 0.02$	$10\text{ ms} + 40\text{ ns} + 15\text{ ns} = 10.000.055\text{ ns}$

Pertanto il tempo medio di accesso alla gerarchia di memoria è dato da:

$$T_m = 0.9 * 15 + 0.03 * 55 + 0.02 * 10.000.055 = 200.017\text{ ns}$$

2. Vedi dispense del corso.