

**PROVA SCRITTA DEL CORSO DI**  
**CALCOLATORI ELETTRONICI**  
**NUOVO ORDINAMENTO DIDATTICO (7 CFU)**  
21 Giugno 2011

**NOME:**

**COGNOME:**

**MATRICOLA:**

**ESERCIZIO 1 (7 punti)**

1. (4 punti) Progettare un flip flop JK a partire da un flip flop D. Disegnare il circuito finale indicando chiaramente la funzione di transizione dello stato, il numero di stati, la funzione di uscita.
2. (3 punti) Spiegare in modo chiaro e sintetico la differenza tra una rete sequenziale sincrona e una asincrona.

**ESERCIZIO 2 (6 punti)**

I trasferimenti di parole  $a$ /dalla memoria di un calcolatore sono codificate utilizzando il codice di Hamming. Si consideri la stringa di 8 bit 11010111 (il bit meno significativo è a sinistra). Spiegando bene ogni passo del ragionamento:

1. (1 punto) calcolare il minimo numero di bit di controllo necessari per la codifica della parola;
2. (3 punti) codificare la stringa data;
3. (2 punti) imporre un errore nel terzo bit della stringa informazione e spiegare come l'errore viene rivelato e corretto per mezzo della codifica di Hamming.

**ESERCIZIO 3 (9 punti)**

Si scriva il codice Assembly 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 (6 punti)**

Si consideri una memoria primaria costituita da 8 parole indirizzabili e da una cache formata da quattro parole.

1. (3 punti) Spiegare come vengono interpretati gli indirizzi logici per recuperare l'informazione contenuta nella cache nel caso venga usata la modalità di indirizzamento:
  - a. Diretto, con blocchi di due parole
  - b. "associativo su insiemi", con insiemi di due blocchi di una parola ciascuno.
2. (3 punti) Ipotizzando la cache vuota all'istante iniziale, indicare nei casi (a-b) lo stato finale della cache e il numero di hit nel caso vengano inoltrate le seguenti chiamate (indirizzi espressi in decimale, primo indirizzo 0): 4, 6, 6, 5, 1, 3, 7, 6, 3, 6. Nel caso (b) si ipotizzi una politica di sostituzione dei blocchi FIFO.

**ESERCIZIO 5 (6 punti)**

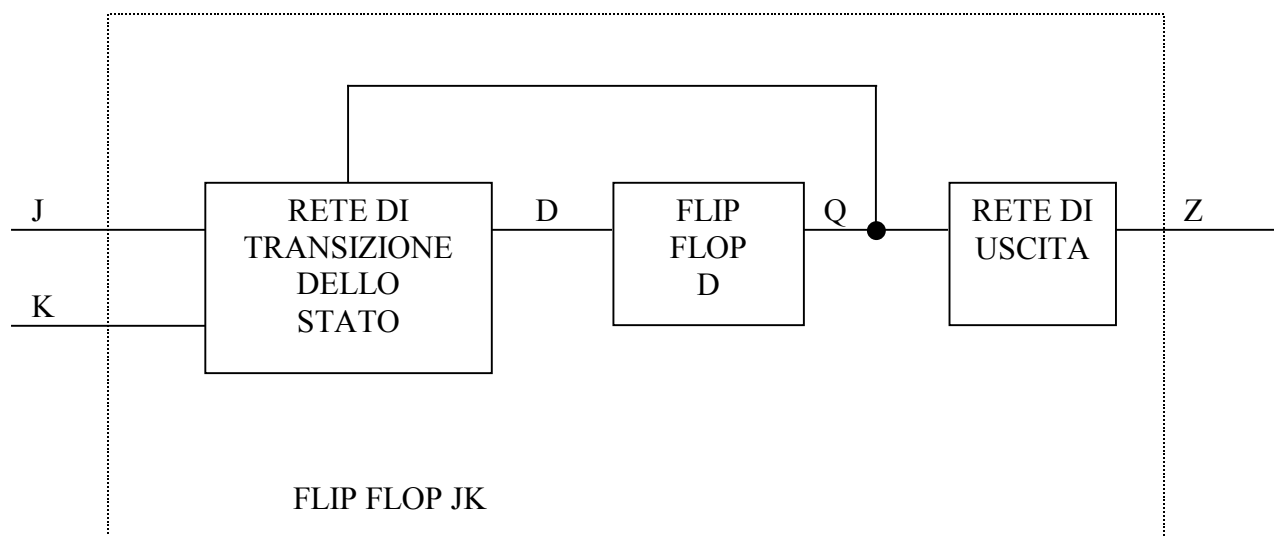
Sia data la seguente lista di processi (si supponga che l'istante iniziale sia 0):

Job	Tempo di Arrivo	Tempo di CPU richiesto
1	0.0	2.0
2	1.2	1.1
3	2.0	0.7
4	2.5	1.2

1. (3 punti) Mostrare la sequenza di esecuzione dei job usando un grafico (tempo, job), qualora si impieghi la politica di scheduling SJF monoprogrammata.
2. (2 punti) Calcolare il tempo di *turnaround* medio e il tempo di *turnaround* pesato medio.

**ESERCIZIO 1****Soluzione**

1. Come in ogni rete logica sequenziale, lo schema del dispositivo è il seguente:

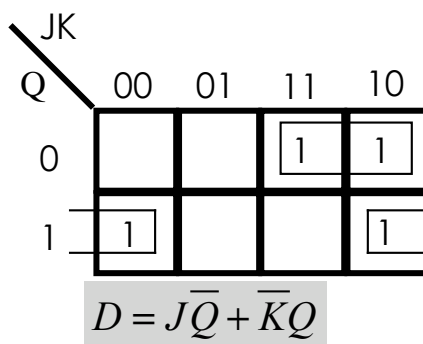


L'uscita di un flip flop JK corrisponde ai valori dello stato (due), per cui si ha  $Z=Q$ .

Rimane da definire solo la funzione di transizione dello stato, che si ottiene dalla tabella di transizione di un flip flop JK e dalla tabella di eccitazione del flip flop D:

J	K	Q	Q'	D
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

Sintetizzando con le mappe di Karnaugh:



2. Vedi dispense del corso.

## ESERCIZIO 2

### Soluzione

1) Deve essere rispettata la condizione:

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

dove K è il numero di bit di controllo inseriti. Essendo N=8, il numero minimo di bit di controllo richiesto è 4.

2) Nella codifica di Hamming, la sequenza in ingresso presenta la seguente struttura:

c0	c1	b0	c2	b1	b2	b3	c3	b4	b5	b6	b7
		1		1	0	1		0	1	1	1

Dove c<sub>0</sub>...c<sub>3</sub> sono i quattro bit costituenti il vettore di controllo, e b<sub>0</sub>...b<sub>7</sub> gli otto bit trasmessi. Tali bit si ottengono con le seguenti operazioni

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

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

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

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

La stringa codificata è 001110110111.

3) Nell'ipotesi di un errore sul terzo bit della stringa iniziale, la stringa ricevuta risulta: 001111100111. Per rivelare questo errore, bisogna ricalcolare i bit di controllo:

$$c'_0 = b_0 \oplus b_1 \oplus b_3 \oplus b_4 \oplus b_6 = 1 \oplus 1 \oplus 1 \oplus 0 \oplus 1 = 0$$

$$c'_1 = b_0 \oplus b_2 \oplus b_3 \oplus b_5 \oplus b_6 = 1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 = 1$$

$$c'_2 = b_1 \oplus b_2 \oplus b_3 \oplus b_7 = 1 \oplus 1 \oplus 1 \oplus 1 = 0$$

$$c'_3 = b_4 \oplus b_5 \oplus b_6 \oplus b_7 = 0 \oplus 1 \oplus 1 \oplus 1 = 1$$

Il passo successivo è calcolare il vettore di errore dato dalla differenza dei vettori di controllo c e c' (ricordiamo che somma e differenza tra bit producono lo stesso risultato):

$$e_0 = c_0 \oplus c'_0 = 0$$

$$e_1 = c_1 \oplus c'_1 = 1$$

$$e_2 = c_2 \oplus c'_2 = 1$$

$$e_3 = c_3 \oplus c'_3 = 0$$

Poiché il vettore risultante 0110 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 sesto (b<sub>2</sub>), che può venire dunque corretto.

**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.

Memoria indirizzabile: 8 parole =  $2^3$  parole  $\rightarrow$  3 bit di indirizzamento

(a) &lt;TAG 1 bit&gt; &lt;Cache Index 1 bit&gt; &lt;Offset 1 bit&gt;

(b) &lt;TAG 2 bit&gt; &lt;Cache Index 1 bit&gt;

2. Nel seguito, è disegnata la cache di quattro parole, e il suo stato finale nei casi a-b. Con linee tratteggiate indichiamo la separazione tra un blocco e il successivo, con linee continue la separazione tra due insiemi.

(a)

0
1
6
7

(b)

4
6
3
7

Tale stato finale è dovuto alle seguenti dinamiche. La prima riga in tabella riporta la parola chiamata, l'ultima il verificarsi di un hit.

Metodo Diretto

4	6	6	5	1	3	7	6	3	6
<b>4</b>	4	4	4	0	0	0	0	0	0
5	5	5	<b>5</b>	<b>1</b>	1	1	1	1	1
	<b>6</b>	<b>6</b>	6	6	2	6	<b>6</b>	2	<b>6</b>
	7	7	7	7	<b>3</b>	<b>7</b>	7	<b>3</b>	7
			h	h			h		

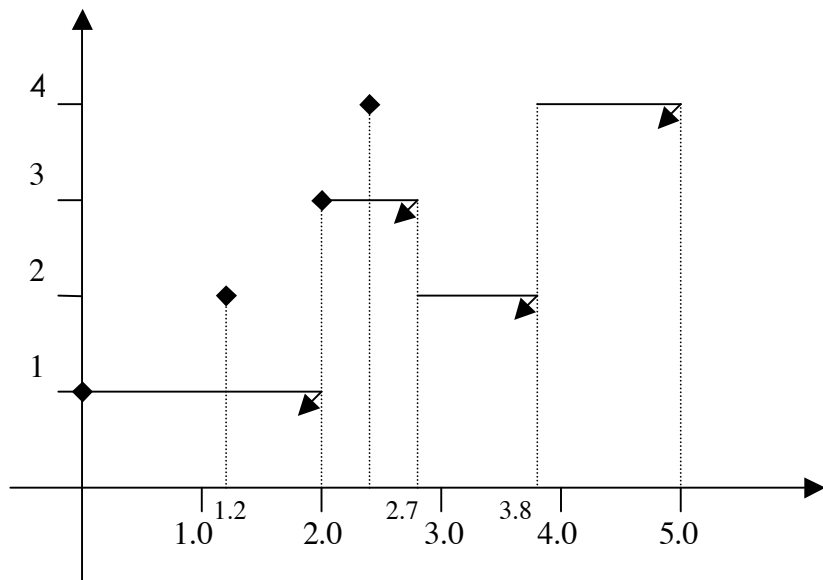
Metodo associativo su insiemi

4	6	6	5	1	3	7	6	3	6
<b>4</b>	4	4	4	4	4	4	4	4	4
	<b>6</b>	<b>6</b>	6	6	6	6	<b>6</b>	6	<b>6</b>
			<b>5</b>	5	<b>3</b>	3	3	<b>3</b>	3
			<b>1</b>	1	1	<b>7</b>	7	7	7
							h	h	h

Metodo diretto: 3/10 hit

Metodo associativo su insiemi: 4/10 hit

3. Vedi dispense del corso.

**ESERCIZIO 4****Soluzione**

- 1.
- 2.

Job	Arrivo	Start	Finish	CPU time	Turnaround	W.turnaround
1	0.00	0.00	2.00	2.00	2.00	1.00
2	1.20	2.70	3.80	1.10	2.60	2.36
3	2.00	2.00	2.70	0.70	0.70	1.00
4	2.50	3.80	5.00	1.20	2.50	2.08
<b>Average</b>					1.95	1.61