

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

13 Luglio 2010

**NOME:**

**COGNOME:**

**MATRICOLA:**

**ESERCIZIO 1 (7 punti)**

Si consideri una rete sequenziale avente un ingresso  $x$  e una uscita  $z$ . L'uscita  $z = 1$  quando viene riconosciuta la sottosequenza 101100. Negli altri casi l'uscita  $z = 0$ .

1. (3 punti) Disegnare il diagramma degli stati.
2. (2 punti) Codificare gli stati e scrivere la tabella di flusso. Si scriva poi la tabella delle transizioni qualora si usino flip-flop di tipo T.
3. (2 punti) Calcolare le forme minime per le variabili di eccitazione dei flip-flop e per l'uscita impiegando le mappe di Karnaugh.

**ESERCIZIO 2 (8 punti)**

1. (2 punti) Mostrare la suddivisione nei relativi campi TAG, Index, Offset di un indirizzo di memoria primaria di 1 Kbyte, disponendo di una cache da 32 byte, secondo il metodo diretto con blocchi da 4 byte.
2. (4 punti) Si consideri la sequenza di riferimenti alla memoria indicati come indirizzi di parole (in formato decimale, il primo indirizzo è 0): 389, 719, 387, 697, 306, 308, 194, 198, 699, 310. Indicare i cache *hit* e il contenuto finale della cache, nella configurazione al punto 1. E' sufficiente indicare, per ciascun blocco di cache, l'indirizzo del corrispondente blocco di primaria in esso eventualmente presente.
3. (2 punti) Indicare almeno una possibile alternativa per aumentare il numero di hit in cache, senza alterare la dimensione di memoria primaria e di cache e il metodo di indirizzamento adottato.  
(Suggerimento. Ricordare il principio di località: chiamate vicine in ordine di tempo e di spazio)

**ESERCIZIO 3 (6 punti)**

Implementare una procedura Assembly MIPS che, dati l'indirizzo iniziale di un vettore  $v$  (in \$4) e due indici  $i$  e  $j$  (rispettivamente in \$5 e in \$6), scambi  $v(i)$  con  $v(j)$  solo se  $v(i) < v(j)$  (cioè se gli elementi corrispondenti ai due indici sono diversi).

**ESERCIZIO 4 (7 punti)**

Un certo numero di periferiche sono collegate al calcolatore per mezzo di un bus sincrono. L'arbitraggio viene effettuato secondo lo schema "daisy chain".

1. (3 punti) Mostrare lo schema di collegamento delle periferiche al bus indicando le diverse linee che devono essere presenti nel bus per gestire l'arbitraggio e il trasferimento dati.
2. (2 punti) Descrivere il protocollo usato dal controller del bus per individuare la periferica che ha richiesto l'uso del bus.
3. (2 punti) Descrivere il protocollo per effettuare la lettura di un dato dalla memoria su un bus sincrono quando tale lettura è iniziata dalla periferica.

**ESERCIZIO 5 (5 punti)**

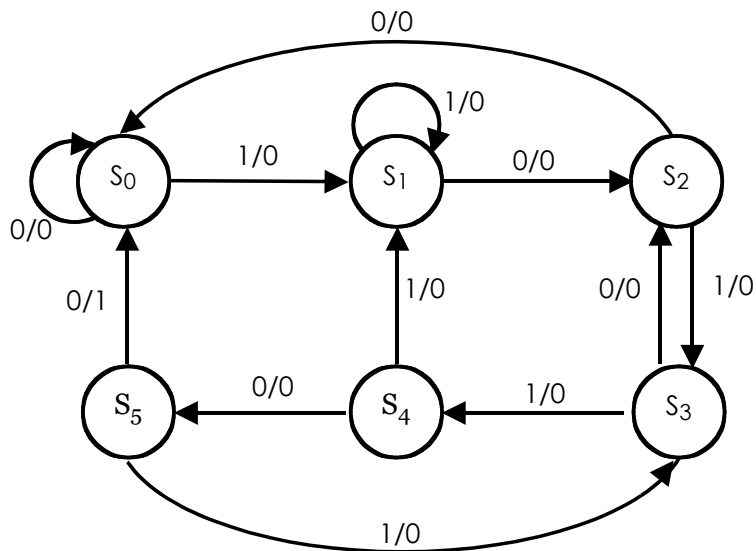
I trasferimenti di parole a/dalla memoria di un calcolatore sono codificate utilizzando il codice di Hamming. Si consideri la stringa di 12 bit 001001101110 (il bit meno significativo è a sinistra), risultata dalla codifica di una parola di  $N$  bit secondo il codice di Hamming.

1. (1 punto) Calcolare  $N$ , supponendo di aver fatto uso del numero minimo di bit di controllo necessario per una stringa di 12 bit;
2. (1 punto) scrivere la parola di  $N$  bit a partire dalla stringa data;
3. (3 punti) indicare eventuali errori nella stringa codificata, specificando quale dei bit è stato alterato.

## ESERCIZIO 1

### Soluzione

Grafo degli stati



Codifica degli stati  
(3 bit -  $Y_2Y_1Y_0$ )

$S_0 \rightarrow 000$        $S_1 \rightarrow 001$   
 $S_2 \rightarrow 010$        $S_3 \rightarrow 011$   
 $S_4 \rightarrow 100$        $S_5 \rightarrow 101$

Tabella di flusso

Stato iniziale	Stato finale/uscita	
	x = 1	x = 0
$S_0$	$S_1/0$	$S_0/0$
$S_1$	$S_1/0$	$S_2/0$
$S_2$	$S_3/0$	$S_0/0$
$S_3$	$S_4/0$	$S_2/0$
$S_4$	$S_1/0$	$S_5/0$
$S_5$	$S_3/0$	$S_0/1$

Tabella delle transizioni (FF-T)  $Y'$  indica lo stato futuro

A	B	C	X	A'	$T_A$	B'	$T_B$	C'	$T_C$	Z
0	0	0	1	0	0	0	0	1	1	0
0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	1	0	0
0	0	1	0	0	0	1	1	0	1	0
0	1	0	1	0	0	1	0	1	1	0
0	1	0	0	0	0	0	1	0	0	0
0	1	1	1	1	1	0	1	0	1	0
0	1	1	0	0	0	1	0	0	1	0
1	0	0	1	0	1	0	0	1	1	0
1	0	0	0	1	0	0	0	1	1	0
1	0	1	1	0	1	1	1	1	0	0
1	0	1	0	0	1	0	0	0	1	1
1	1	0	1	d	d	d	d	d	d	d
		⋮					⋮			
1	1	1	1	d	d	d	d	d	d	d

Flip-flop T		
Y	Y'	T
0	0	0
0	1	1
1	0	1
1	1	0

Uscita  $z = Y_2 Y_0 x$

<div> <div> <div>AB</div> <div>CX</div> </div> <table> <tr> <th></th> <th>00</th> <th>01</th> <th>11</th> <th>10</th> </tr> <tr> <th>00</th> <td></td> <td></td> <td>d</td> <td></td> </tr> <tr> <th>01</th> <td></td> <td></td> <td>d</td> <td>1</td> </tr> <tr> <th>11</th> <td></td> <td>1</td> <td>d</td> <td>1</td> </tr> <tr> <th>10</th> <td></td> <td></td> <td>d</td> <td>1</td> </tr> </table> <math display="block">T_A = BCX + AX + AC</math> </div>		00	01	11	10	00			d		01			d	1	11		1	d	1	10			d	1	<div> <div> <div>AB</div> <div>CX</div> </div> <table> <tr> <th></th> <th>00</th> <th>01</th> <th>11</th> <th>10</th> </tr> <tr> <th>00</th> <td></td> <td>1</td> <td>d</td> <td></td> </tr> <tr> <th>01</th> <td></td> <td></td> <td>d</td> <td></td> </tr> <tr> <th>11</th> <td></td> <td>1</td> <td>d</td> <td>1</td> </tr> <tr> <th>10</th> <td>1</td> <td></td> <td>d</td> <td></td> </tr> </table> <math display="block">T_B = \overline{A}B\overline{C}\overline{X} + B\overline{C}\overline{X} + BCX + ACX</math> </div>		00	01	11	10	00		1	d		01			d		11		1	d	1	10	1		d		<div> <div> <div>AB</div> <div>CX</div> </div> <table> <tr> <th></th> <th>00</th> <th>01</th> <th>11</th> <th>10</th> </tr> <tr> <th>00</th> <td></td> <td></td> <td>d</td> <td>1</td> </tr> <tr> <th>01</th> <td>1</td> <td>1</td> <td>d</td> <td>1</td> </tr> <tr> <th>11</th> <td></td> <td>1</td> <td>d</td> <td></td> </tr> <tr> <th>10</th> <td>1</td> <td>1</td> <td>d</td> <td>1</td> </tr> </table> <math display="block">T_C = \overline{A}\overline{C} + \overline{C}X + BX + C\overline{X}</math> </div>		00	01	11	10	00			d	1	01	1	1	d	1	11		1	d		10	1	1	d	1
	00	01	11	10																																																																									
00			d																																																																										
01			d	1																																																																									
11		1	d	1																																																																									
10			d	1																																																																									
	00	01	11	10																																																																									
00		1	d																																																																										
01			d																																																																										
11		1	d	1																																																																									
10	1		d																																																																										
	00	01	11	10																																																																									
00			d	1																																																																									
01	1	1	d	1																																																																									
11		1	d																																																																										
10	1	1	d	1																																																																									
	<div> <div> <div>AB</div> <div>CX</div> </div> <table> <tr> <th></th> <th>00</th> <th>01</th> <th>11</th> <th>10</th> </tr> <tr> <th>00</th> <td></td> <td></td> <td>d</td> <td></td> </tr> <tr> <th>01</th> <td></td> <td></td> <td>d</td> <td></td> </tr> <tr> <th>11</th> <td></td> <td></td> <td>d</td> <td></td> </tr> <tr> <th>10</th> <td></td> <td></td> <td>d</td> <td>1</td> </tr> </table> <math display="block">Z = A\overline{C}\overline{X}</math> </div>		00	01	11	10	00			d		01			d		11			d		10			d	1																																																			
	00	01	11	10																																																																									
00			d																																																																										
01			d																																																																										
11			d																																																																										
10			d	1																																																																									

## ESERCIZIO 2

### Soluzione

1. <TAG 5 bit> <Cache Index 3 bit> <Offset 2 bit>
2. Applicando per ciascuna chiamata le formule per il calcolo di block frame e cache index, si ricavano i seguenti valori per ciascun set:

<b>Indirizzo</b>	389	719	387	697	306	308	194	198	699	310
<b>Blocco primaria</b>	97	179	96	174	76	77	48	49	174	77
<b>Blocco cache</b>	1	3	0	6	4	5	0	1	6	5
<b>Hit</b>									X	X

Il contenuto finale della cache è dunque:

<b>Blocco Cache</b>	0	1	2	3	4	5	6	7
<b>Blocco Primaria</b>	48	49		179	76	77	174	

3. La cosa più semplice per ridurre il numero di miss in cache è accrescere la grandezza dei blocchi. Ciò si può notare dal fatto che le richieste in primaria distano spesso meno di otto parole (e.g. prima e terza chiamata, quinta, sesta e ultima chiamata). Ad es. possiamo utilizzare blocchi da 8 byte. In questo caso la cache conterrà solo 4 blocchi, con la seguente sequenza:

<b>Indirizzo</b>	389	719	387	697	306	308	194	198	699	310
<b>Blocco primaria</b>	48	89	48	87	38	38	24	24	87	38
<b>Blocco cache</b>	0	1	0	3	2	2	0	0	3	2
<b>Hit</b>			X			X		X	X	X

Come si nota il numero di hit è passato da 2 a 5.

### ESERCIZIO 3

#### Soluzione

$\$9 \leftarrow v[i]; \$10 \leftarrow v[j]$

```
swap:      addi $29, $29, -16
           sw $9, 0($29)
           sw $10, 4($29)
           sw $5, 8($29)
           sw $6, 12($29)

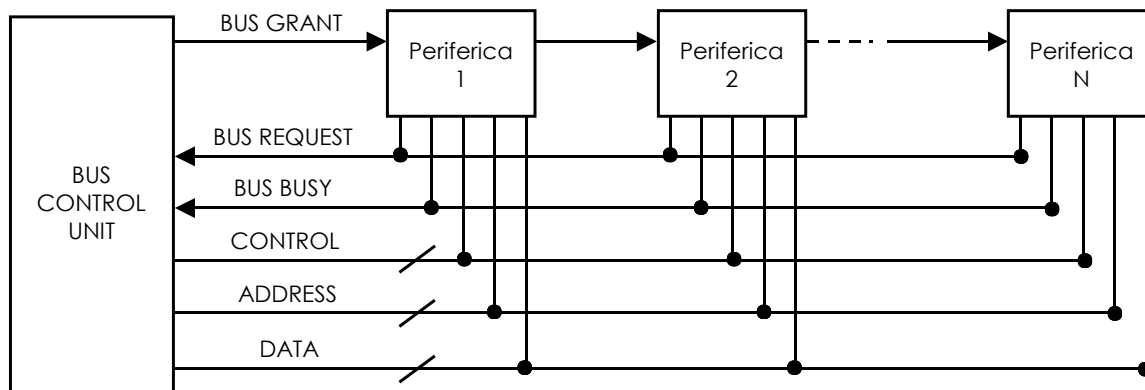
           muli $5, $5, 4
           add $5, $5, $4      #indirizzo di v[i] in $5
           muli $6, $6, 4
           add $6, $6, $4      #indirizzo di v[j] in $6
           lw $9, 0($5)
           lw $10, 0($6)
           beq $9, $10, exit   #if v[i]==v[j] exit
           sw $9, 0($6)
           sw $10, 0($5)

exit:      lw $9, 0($29)
           lw $10, 4($29)
           lw $5, 8($29)
           lw $6, 12($29)
           addi $29, $29, 16

           jr $31
```

## ESERCIZIO 4

### Soluzione.



1. Le linee necessarie per il bus sono indicate in figura. Le linee di bus request, grant e busy, sono utilizzate per effettuare l'arbitraggio, mentre le linee control, address e data vengono usate per trasferire i dati. N.B. le linee di control, address e data sono in realtà "gruppi" di linee. La motivazione è contenuta nella risposta 3.
2. Arbitraggio in daisy chain: La periferica invia il segnale di richiesta del bus sulla linea bus request, se la linea bus busy indica che il bus è libero. Il controllore del bus invia in risposta il segnale di bus grant sull'apposita linea. Questo segnale viene propagato serialmente fra tutte le periferiche, fino a raggiungere la periferica che ha fatto richiesta, che blocca la propagazione del segnale di bus grant e attiva il segnale di bus busy. (in caso di richieste contemporanee la propagazione del bus grant viene bloccata dalla prima periferica che ha fatto richiesta). A questo punto la periferica può trasferire i dati.
3. Sulle linee di controllo viene inviato il segnale sul tipo di trasferimento, in questo caso lettura (ci sono più linee, ciascuna per un tipo di trasferimento come lettura, scrittura, trasferimento di blocchi, ecc.), e contemporaneamente l'indirizzo di memoria che deve essere letto sulla linea indirizzi (address). Trascorso il tempo di ciclo della memoria, il dato viene inviato sulla linea data che può consentire il trasferimento contemporaneo di un certo numero di byte.

## ESERCIZIO 5

### 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  $N + K = 12$ , si evince dalla (1) che il numero minimo di bit di controllo richiesto è 4. Da cui  $N = 8$ .

2. 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	1	0	0	1	1	0	1	1	1	0

Dove  $c_0 \dots c_3$  sono i quattro bit costituenti il vettore di controllo, e  $b_0 \dots b_7$  gli otto bit trasmessi. La sequenza ricevuta è 10111110.

3. Per verificare la presenza di un errore, dobbiamo ricalcolare il vettore di controllo a partire dalla sequenza ricevuta. Si ha:

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

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

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

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

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):

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

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

$$e_2 = c_2 \oplus c'_2 = 0$$

$$e_3 = c_3 \oplus c'_3 = 1$$

Poiché il vettore risultante 1010 non è nullo, vi è un errore nella stringa di 12 bit data e precisamente nella posizione indicata dal vettore di errore tradotto in notazione decimale. Il bit sbagliato è quindi il decimo ( $b_5$ ), e la parola corretta è 10111010.