

PROVA SCRITTA DEL MODULO DI
CALCOLATORI ELETTRONICI
NUOVO E VECCHIO ORDINAMENTO DIDATTICO (5-6 CFU)
13 gennaio 2016

NOME:

COGNOME:

MATRICOLA:

CFU:

ESERCIZIO 1 (8 punti)

Progettare una rete logica sequenziale che riconosca la sequenza 10010. Si dispone di un FF-T, da utilizzare per il bit di stato meno significativo, e di FF-D da usare per gli altri bit di stato. Indicare anche la rete combinatoria di uscita.

ESERCIZIO 2 (8 punti)

E' data una gerarchia di memorie cache-primaria. La memoria primaria è di 1 KB mentre la cache è di 256 B. E' possibile indirizzare il singolo byte, e la memoria primaria è suddivisa in blocchi di 32 B.

1. (2 punti) Indicare, specificando l'ampiezza e la funzione dei diversi campi, come vengono interpretati gli indirizzi di memoria primaria secondo il metodo associativo su insiemi a N vie, valutando tutte le possibili configurazioni del metodo.
2. (6 punti) Ipotizzando la cache vuota, si consideri la sequenza di chiamate seguenti: dalla parola 0 alla 63, dalla 256 alla 319, dalla 512 alla 575, dalla 768 alla 831, in quest'ordine, per dieci volte consecutive. Indicare quale sia il valore di N che massimizza H_c motivando ogni passaggio del ragionamento, e scrivere a quanto ammonti il corrispondente valore di H_c . Mostrare infine lo stato finale della cache.

ESERCIZIO 3 (9 punti)

(7 punti) Si scriva una funzione Assembly MIPS $salta(k)$ che, ricevendo come ingresso un valore k , memorizzato in \$4, intero compreso tra 0 e 3, salti a quattro aree di memoria contenenti un numero imprecisato di istruzioni. Ciascuna area parte dagli indirizzi 2^{16} , 2^{17} , 2^{18} , 2^{19} , ai quali la funzione deve saltare per k pari a 0, 1, 2, 3, rispettivamente. Nello scrivere la funzione, si faccia uso di una funzione $p(k)$ che, ricevendo in ingresso un valore intero k in \$4, **compreso tra 0 e 15**, restituisce il valore 2^k in \$5.

(2 punti) Si indichi con quale istruzione debba terminare ciascuna delle quattro sequenze di istruzioni alla quale la funzione deve saltare.

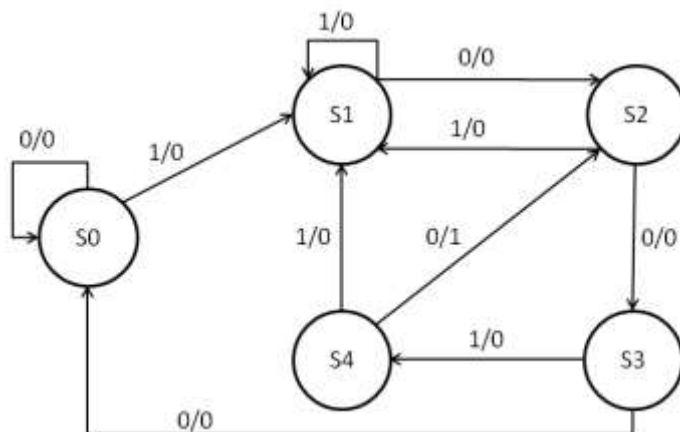
ESERCIZIO 4 (8 punti)

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. (4 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 1

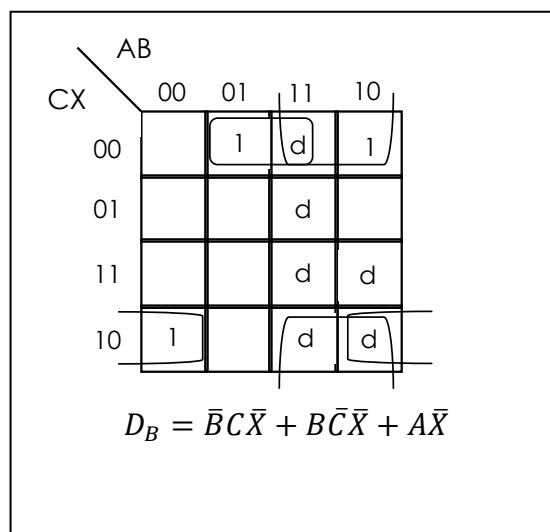
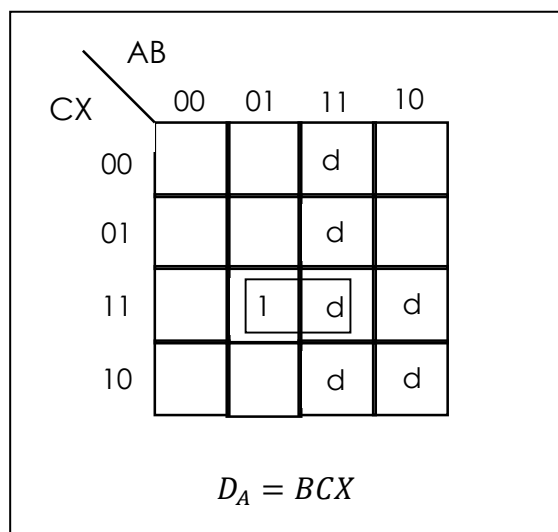
Il grafo degli stati è il seguente:



Rispettando i vincoli forniti dal problema, utilizziamo il FF-T per memorizzare il bit di stato meno significativo rispetto ai tre necessari per codificare ciascuno stato, mentre per gli altri utilizziamo FF-D, ottenendo:

A	B	C	X	A' (DA)	B' (DB)	C'	TC	Z
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	1	0
0	0	1	0	0	1	0	1	0
0	0	1	1	0	0	1	0	0
0	1	0	0	0	1	1	1	0
0	1	0	1	0	0	1	1	0
0	1	1	0	0	0	0	1	0
0	1	1	1	1	0	0	1	0
1	0	0	0	0	1	0	0	1
1	0	0	1	0	0	1	1	0
1	0	1	0	D	D	D	D	0
1	0	1	1	D	D	D	D	0
1	1	0	0	D	D	D	D	0
1	1	0	1	D	D	D	D	0
1	1	1	0	D	D	D	D	0
1	1	1	1	D	D	D	D	0

Occorre semplificare le tre mappe di Karnaugh per ciascuna delle uscite:



		AB			
CX		00	01	11	10
	00		1	d	
	01	1	1	d	1
	11		1	d	d
	10	1	1	d	d

$$T_c = \bar{C}X + C\bar{X} + B$$

Per quanto riguarda l'uscita: $Z = A\bar{B}\bar{C}\bar{X}$.

ESERCIZIO 2

1. Memoria indirizzabile 2^{10} B \rightarrow 10 bit di indirizzamento, di cui 5 per l'offset (i blocchi sono di 2^5 B). Il campo restante da 5 bit va distribuito in funzione del metodo di indirizzamento. Nel caso del metodo set-associativo a N vie, possiamo avere quattro possibilità:
- 1) Un insieme, otto vie \rightarrow metodo completamente associativo
 - 2) Due insiemi, quattro vie
 - 3) Quattro insiemi, due vie
 - 4) Otto insiemi, una via \rightarrow metodo diretto (un insieme corrisponde ad una linea)

Tralasciando i casi 1 e 4, si hanno dunque due configurazioni:

< TAG 4 bit > < C.I. 1 bit > < Offset 5 bit >, con N=4

< TAG 3 bit > < C.I. 2 bit > < Offset 5 bit >, con N=2

2. Il primo blocco di chiamate 0-63 corrisponde ad i primi due blocchi di primaria, 0-31 e 32-63. Essi presentano cache index 0 e 1 rispettivamente. Secondo, terzo e quarto blocco di chiamate sono ancora costituiti da due blocchi di primaria con stessi cache index, 0 e 1. E' evidente che per memorizzarli tutti senza rischi di dover sovrascriverli, dobbiamo avere almeno due insiemi in cache per ciascuno dei quali avere almeno quattro vie. In questo modo potremo anche massimizzare Hc, visto che la sequenza è richiamata 10 volte. Delle due configurazioni al punto precedente, l'unica percorribile è dunque la prima, con due insiemi, ognuno dei quali a quattro vie.

La cache a questo punto è saturata ma tutte le parole chiamate sono memorizzate. Il numero di hit è dato da $31 \text{ (hit per blocco)} * 8 \text{ (blocchi chiamati)} = 248$ alla prima invocazione. La sequenza di blocchi viene chiamata per altre 9 volte: tutte le parole sono presenti in cache, determinando $32*8 = 256$ hit in totale per ciascuna delle iterazioni. $Hc = (248+9*256)/10*256 = 0.996875$.

Stato finale della cache:

Insieme 0				Insieme 1			
L0	L1	L2	L3	L4	L5	L6	L7
0-31	256-287	512-543	768-799	32-63	288-319	544-575	800-831

ESERCIZIO 3

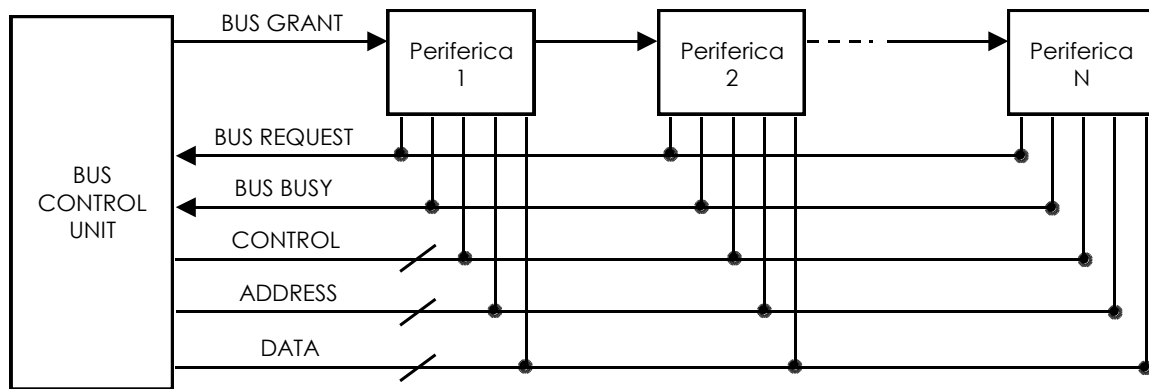
Utilizzando i registri col seguente significato:

\$8 \rightarrow indirizzo a cui saltare in funzione di k

```
salta: addi $29, $29, -12
      sw $8, 0($29)
      sw $5, 4($29)           #non sappiamo se $5 è in uso dal chiamante
      sw $31, 8($29)          #prevediamo chiamata alla funzione p
      move $8, $0
      lui $8, 1                #dopo questa operazione  $2^{16} \rightarrow \$8$ 
      jal p
      mul $8, $8, $5           #l'indirizzo a cui saltare è dato da  $2^{16} * 2^k$ 
      jr $8
exit:  lw $8, 0($29)
      lw $5, 4($29)
      lw $31, 8($29)
      addi $29, $29, 12
      jr $31
```

L'istruzione finale di ciascun'area di memoria dev'essere j exit.

ESERCIZIO 4



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.