

**SOLUZIONI DELLA PROVA SCRITTA DEL CORSO DI  
CALCOLATORI ELETTRONICI**

27 Febbraio 2001

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

**ESERCIZIO 1** (VO: 7 punti - NO: 8 punti)

Si consideri la rete combinatoria caratterizzata da tre ingressi A, B, C e da due uscite le cui funzioni sono:

$$Y_1 = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$$

$$Y_2 = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

- 1) (VO: 1 punti - NO: 2 punti) Scrivere la tabella di verità di  $Y_1$  e  $Y_2$ .
- 2) (VO: 1 punti - NO: 2 punti) Calcolare le forme minime per mezzo delle mappe di Karnaugh.
- 3) (2 punti) Riscrivere le equazioni delle reti logiche risultanti da 2) usando soltanto porte NAND.
- 4) (VO: 1 punti - NO: 2 punti) Progettare, con le mappe di Karnaugh, una terza uscita  $Y_3$  che, a partire da A, B, C realizzi la funzione:

$$Y_3 = Y_1 + Y_2.$$

- 5) (solo VO: 2 punti) Quale funzionalità realizzano  $Y_1$  e  $Y_2$  ?

**Soluzione.**

- 1) Tabella di verità e mappe di Karnaugh.

Tabella di verità					
A	B	C	$Y_1$	$Y_2$	
0	0	0	0	0	
0	0	1	1	0	
0	1	0	1	0	
0	1	1	0	1	
1	0	0	1	0	
1	0	1	0	1	
1	1	0	0	1	
1	1	1	1	1	

A B					
C		00	01	11	10
0			1		1
1		1		1	

A B					
C		00	01	11	10
0				1	
1			1	1	1

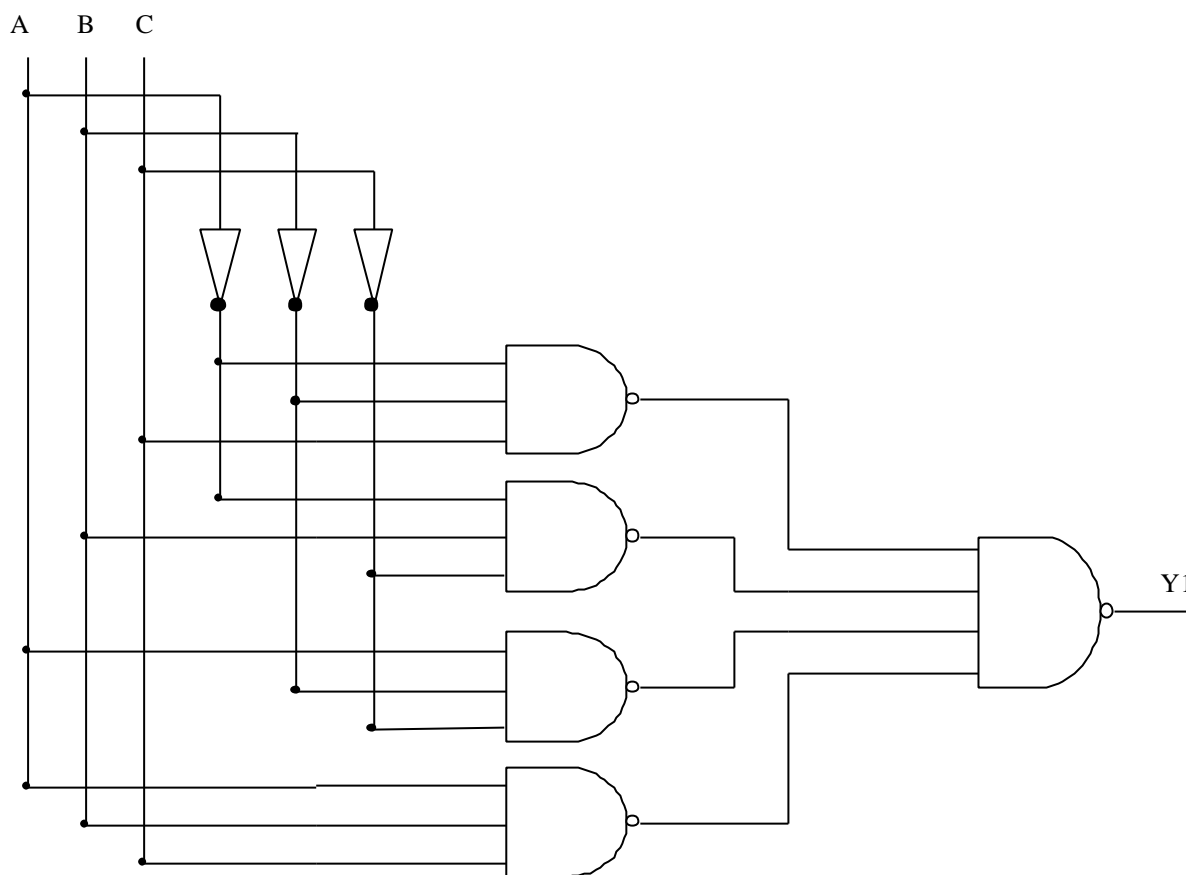
$Y_1$  è già espressa nella forma minima, mentre  $Y_2 = AB + AC + BC$ .

- 2) Rete logica con sole porte NAND.

Per brevità riportiamo la rete logica relativa a  $Y_1$ . La rete logica di  $Y_2$  presenta la stessa struttura (cambiano le connessioni di ingresso e il numero di ingressi supportati da ciascuna NAND del primo livello di logica).

$$Y_1 = \overline{\overline{A}\overline{B}C} \cdot \overline{\overline{A}B\overline{C}} \cdot \overline{A\overline{B}\overline{C}} \cdot \overline{ABC} = (\overline{A} \uparrow \overline{B} \uparrow C) \uparrow (\overline{A} \uparrow B \uparrow \overline{C}) \uparrow (A \uparrow \overline{B} \uparrow \overline{C}) \uparrow (A \uparrow B \uparrow C)$$

Riportiamo il disegno di questa rete.



3) OR di Y1 e Y2 a partire da A, B, C con le mappe di Karnaugh.

Dal momento che viene richiesto un OR, è sufficiente sovrapporre le due mappe precedenti ottenendo:

A B					
C		00	01	11	10
	0		1	1	1
	1	1	1	1	1

Il risultato è dunque:

$$Y3 = A + B + C.$$

4) Funzione realizzata da Y1 e Y2.

Dall'analisi della tabella di verità si ottiene immediatamente  $Y1 = A \oplus B \oplus C$ . Y1 è il bit di parità in corrispondenza degli ingressi, ovvero il bit somma degli ingressi.

Per quanto riguarda Y2, una possibile linea di ragionamento è la seguente. Si nota subito dalla tabella di verità che, quando  $A = 0$ , Y2 è il riporto della somma di B con C, ovvero della somma degli ingressi. Lo stesso si può ricavare, anche dalla espressione algebrica di Y2, nei casi  $B = 0$  (riporto di  $A \oplus C$ ) e  $C = 0$  (riporto di  $A \oplus B$ ). Quanto esposto viene confermato anche nel caso  $A = B = C = 1$ .

La rete combinatoria realizza dunque la somma di A, B e C con l'uscita Y1 e il relativo riporto con l'uscita Y2.

**ESERCIZIO 2 (VO: 7 punti - NO: 8 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 101100100111 (il bit meno significativo è a sinistra), risultata della codifica di una parola di N bit secondo il codice di Hamming. **Spiegando bene ogni passo del ragionamento:**

- 1) (2 punti) calcolare N, supponendo di aver fatto uso del numero minimo di bit di controllo necessario per una stringa di 12 bit;
- 2) (VO: 2 – NO: 3 punti) 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.

**Soluzione.**

- 1) Deve venire 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:

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

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 10010111.

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

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

$$c'_1 = b_0 \oplus b_2 \oplus b_3 \oplus b_5 \oplus b_6 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 1 = 0$$

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

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

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):

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

$$e_1 = c_1 \oplus c'_1 = 0$$

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

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

Poiché il vettore risultante 1100 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 dodicesimo ( $b_7$ ), e la parola corretta è 10010110.

**ESERCIZIO 3 (VO: 8 punti – NO: 9 punti)**

Si consideri un calcolatore in cui la CPU esegue  $10^5$  istruzioni/s. L'esecuzione di una istruzione richiede 5 cicli di clock, 3 dei quali tengono occupato il bus di sistema. Si ipotizzi che il 85% dell'Instruction Rate sia usato dalla CPU per eseguire programmi che non contengono trasferimenti di I/O. L'ampiezza della linea dati del bus è pari a 32 bit.

Si consideri il caso in cui il trasferimento dei dati avvenga mediante IO da programma, con le seguenti 4 istruzioni:

- 1) LOAD *parola* dalla periferica al registro CPU
  - 2) STORE *parola* da registro CPU a memoria
  - 3) generazione indirizzo di memoria successivo
  - 4) conteggio dati da trasferire.
- a) (3 punti) Calcolare la massima frequenza di trasferimento dati ottenibile (espressa in kB/s) fra una periferica collegata al bus di sistema e la memoria principale.
- b) (4 punti) Calcolare la massima frequenza di trasferimento dati ottenibile (espressa in kB/s) nel caso in cui si usi la modalità "transparent" DMA. Si ipotizzi che una operazione di lettura/scrittura della memoria richieda un ciclo di clock.
- c) (VO: 1 – NO: 2) Spiegare quali 'passi' sostituiscono le istruzioni nel caso DMA.
- 

**Soluzione**

- a) Nel caso di trasferimento mediante I/O da programma, per trasferire una parola occorrono 4 istruzioni. La CPU è impegnata per l'85% del tempo a eseguire istruzioni che non coinvolgono l'I/O, dunque può usare solo il 15% del tempo per eseguire istruzioni di trasferimento dati con periferiche. In termini di istr./sec questo tempo è pari a  $0.15 \cdot 10^5 \text{ istr./s} = 1.5 \cdot 10^4 \text{ istr./s}$ . Dal momento che per trasferire una parola servono due istruzioni, la velocità di trasferimento è pari a:
- $$1.5 \cdot 10^4 \text{ istr./s} / (4 \text{ istr./parola}) = 3750 \text{ parole/s.}$$
- La dimensione di una parola è pari a 32 bit (4 byte), da cui si ricava la velocità di trasferimento di **14.65 kB/s**,
- b) Nel caso di 'transparent DMA' posso trasferire i dati tutte le volte che il bus di sistema è libero. Nel caso in esame questo tempo è pari alla somma del 15% del tempo lasciato libero dall'esecuzione di istruzioni che non coinvolgono I/O, più i due cicli/istruzione in cui il bus è libero. Pertanto durante l'85% del tempo posso trasferire due parole/istr.:
- $$0.85 \cdot 2 \text{ parole/istr} \cdot 10^5 \text{ istr./s} = 1.7 \cdot 10^5 \text{ parole/s}$$
- Nel restante 15% del tempo posso trasferire 5 parole/istr.:
- $$0.15 \cdot 5 \text{ parole/istr.} \cdot 10^5 \text{ istr./s} = 0.75 \cdot 10^5 \text{ parole/s}$$
- In totale, nel caso di trasferimento con DMA la velocità totale di trasferimento è pari a:  $(1.7 + 0.75) \cdot 10^5 \text{ parole/s} = 2.45 \cdot 10^5 \text{ parole/s} = 239 \text{ kB/s}$
- c) Il 'controller' DMA esegue le operazioni di generazione indirizzi e conteggio dati trasferiti usando registri interni al controller oltre, ovviamente, a trasferire i dati dalla periferica alla memoria.

**ESERCIZIO 4 (VO: 6 punti – NO: 8 punti)**

Si consideri un sistema operativo con gestione della memoria paginata. Ciascun processo ha una dimensione massima di 128 pagine. Le pagine hanno dimensione 1KB. Si abbia ad esempio la seguente situazione:

Tabella delle Pagine Processo 0			
Pagina Virtuale	Pagina Fisica	Bit di Validità	Solo lettura
0000000	0000011	0	0
0000001	0010110	0	0
0000010	1001001	1	1
0000011	1001010	1	1
0000100	1010101	1	1
0000101	0011101	1	0
0000110	0111111	1	0
0000111	1011101	1	0
0001000	1010011	1	0
0001001	0001111	1	0
0001010	0011011	1	0
0001011	0100010	1	0

Tabella delle Pagine Processo 1			
Pagina Virtuale	Pagina Fisica	Bit di Validità	Solo lettura
0000000	0001010	1	0
0000001	0010001	0	0
0000010	1011001	1	0
0000011	1011010	1	0
0000100	0011101	1	0
0000101	0101111	0	0
0000110	0110111	1	0
0000111	0100000	1	0
0001000	0100100	1	0
0001001	0001001	1	0

Tabella delle Pagine Processo 2			
Pagina Virtuale	Pagina Fisica	Bit di Validità	Solo lettura
0000000	0100001	1	1
0000001	0101110	1	1
0000010	0110110	0	1
0000011	0000110	1	1
0000100	1100011	0	1
0000101	1000011	1	1
0000110	0010101	1	1

Il bit di validità indica se la pagina virtuale richiesta è presente (1) o no (0) nella memoria principale.

- (VO: 2 – NO: 4 punti) Calcolare la dimensione in bit degli indirizzi virtuali, spiegando come vengono interpretati dal sistema operativo.
  - (VO: 2 – NO: 4 punti) Si considerino le seguenti richieste e mostrare: a quale indirizzo fisico corrispondono gli indirizzi virtuali, se l'operazione può essere conclusa con successo o se viene generato un errore (pagina non valida, violazione di protezione) o un page fault:
    - (Processo 0) Read 00010110101001101
    - (Processo 2) Write 00001010101011001
    - (Processo 1) Read 00010011001110011
    - (Processo 2) Read 00001001100100110
    - (Processo 0) Write 00000110001100001
  - (solo VO: 2 punti) calcolare la dimensione massima di una PMT
-

**Soluzione:**

- a) L'indirizzo virtuale ha dimensione pari almeno a 17 bit e viene interpretato a partire dai bit più significativi nel modo seguente:  
 7 bit: pagina virtuale  
 10 bit: posizione all'interno della pagina.
- b) A partire dalla interpretazione del punto 1. Si ottiene:
- 1) Indirizzo fisico valido: 01000100101001101
  - 2) Errore: scrittura su indirizzo fisico 10000110101011001 protetto in scrittura
  - 3) Indirizzo fisico valido: 00010011001110011
  - 4) "Page fault all'indirizzo fisico 00111011100100110
  - 5) Errore: scrittura su indirizzo fisico 10010100001100001 protetto in scrittura.
- c) La PMT contiene al massimo 128 'righe', una per ciascuna pagina virtuale. Ciascuna riga contiene 7 bit per la pagina virtuale, 7 bit per la pagina fisica, 2 bit di controllo, per un totale di 16 bit. La dimensione massima di una PMT è dunque pari a  $128 \times 16$  bit = 256 byte.

**ESERCIZIO 5 (VO: 5 punti)**

- a) (3 punti) In una macchina RISC, descrivere quali problemi sono determinati dalle istruzioni di salto e in che modo vengono risolti.
- b) (2 punti) Si consideri poi la seguente sequenza di istruzioni, individuando un'eventuale bolla di ritardo e adattando il flusso delle istruzioni secondo uno dei criteri esposti in precedenza. Le lettere indicano altrettante locazioni di memoria.

INDIRIZZO	ISTRUZIONE
100	MOV A,P
101	ADD B,1
102	JMP 104
103	MOV P,Z
104	ADD A,C
105	MOV B,X

**Soluzione.**

- a) V. dispense del corso.
- b) Vi è una bolla di ritardo generata dalla istruzione all'indirizzo 102, che causa un salto all'indirizzo 104. Possiamo introdurre in punti opportuni delle istruzioni NOP (No Operation) oppure riadattare la sequenza secondo il criterio "delayed branch".

INDIRIZZO	NORMAL BRANCH	BRANCH CON NOP	OPTIMIZED "DELAYED BRANCH"
100	MOV A,P	MOV A,P	MOV A,P
101	ADD B,1	ADD B,1	JMP 104
102	JMP 104	JMP 105	ADD B,1
103	MOV P,Z	NOP	MOV P,Z
104	ADD A,C	MOV P,Z	ADD A,C
105	MOV B,X	ADD A,C	MOV B,X
106		MOV B,X	