Politecnico di Milano

Dipartimento di Elettronica e Informazione

prof.ssa Anna Antola **prof. Fabrizio Ferrandi**
prof.ssa Cristiana Bolchini

Reti Logiche A - Prova di mercoledì 17 novembre 2004

Matricola_______ _________________________

Cognome_______ _____________________**Nome** _______________________

Istruzioni

- Scrivere solo sui fogli distribuiti. Non separare questi fogli.
- È vietato portare all'esame libri, eserciziari, appunti e calcolatrici. Chiunque venga trovato in possesso di documentazione relativa al corso – anche se non strettamente attinente alle domande proposte – vedrà annullata la propria prova.
- Non è possibile lasciare l'aula conservando il tema della prova in corso.
- Tempo a disposizione: 2h:00m.

Valore indicativo di domande ed esercizi e voti parziali:

Esercizio 1 (2 punti) ___________________________

Esercizio 2 (2 punti) ___________________________

Esercizio 3 (2 punti) ___________________________

Esercizio 4 (2 punti) ___________________________

Esercizio 5 (3 punti) ___________________________

Esercizio 6 (3 punti) ___________________________

Esercizio 7 (2 punti) ___________________________

Esercizio 8 (punteggio non preassegnato) _______________

Con Soluzioni

Esercizio n. 1

Data la seguente espressione logica:
a’*((c’+d’)’+e)+ ab*(e+ed+cd) + (cde’+e)’ *(a’+ab)

la si semplifichi, utilizzando le proprietà dell’algebra di commutazione. Riportare per ogni passaggio la proprietà utilizzata.

Soluzione:

a’*((c’+d’)’+e)+ ab*(e+ed+cd) + (cde’+e)’ *(a’+ab)
=> Assorbimento: e+ed = e

a’*((c’+d’)’+e)+ ab*(e+cd) + (cde’+e)’ *(a’+ab)
=> De Morgan: (c’+d’)’ = (c’)’*(d’)’;

a’*((c’)’*(d’)’+e)+ ab*(e+cd) + (cde’+e)’ *(a’+ab)
=> Involuzione: (c’)’ = c

a’*(c*(d’)’+e)+ ab*(e+cd) + (cde’+e)’ *(a’+ab)
=> Involuzione: (d’)’ = d

a’*(cd+e)+ ab*(e+cd) + (cde’+e)’ *(a’+ab)
=> Semplificazione: cde’+e = cd + e;

a’*(cd+e)+ ab*(e+cd) + (cd+e)’ *(a’+ab)
=> Commutativa: e + cd = cd +e;

a’*(cd+e)+ ab(cd+e) + (cd+e)’ (a’+ab)
=> Distributiva: a’(cd+e)+ ab(cd+e) = (cd+e)(a’+ab) ;

(cd+e)*(a’+ab) + (cd+e)’*(a’+ab)
=> Distributiva: (cd+e)*(a’+ab) + (cd+e)’*(a’+ab) = (a’+ab)*((cd+e)+ (cd+e)’);

(a’+ab)*((cd+e)+ (cd+e)’)
=> Inverso: (cd+e)+ (cd+e)’ = 1;

(a’+ab)*1
=> Elemento Neutro: (a’+ab)*1 = a’+ab;

a’+ab
=> Semplificazione: a’ + ab = a’ + b;

a’+b
=> Soluzione Finale.

Esercizio n. 2

Data la seguente funzione ad una uscita, non completamente specificata:

F(a,b,c,d)= ONset(2,3,6,9,12,13,15) DCset(0,4,7)

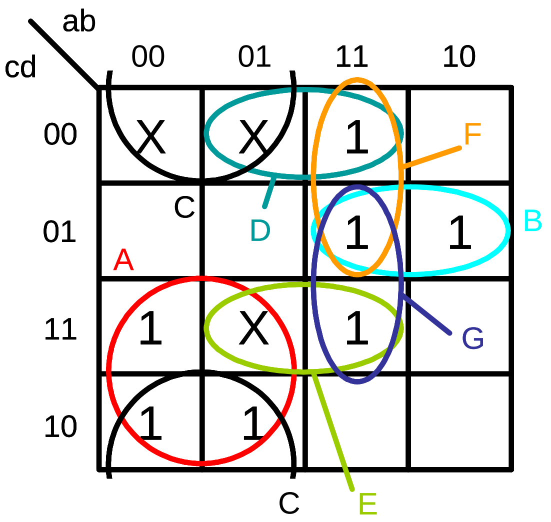
I) Sulla mappa di Karnaugh individuare gli implicanti primi **riportandone la forma algebrica** e separando gli implicanti *primi* da quelli *primi ed essenziali*.

II) Ricavare tutte le forme minime scegliendo una opportuna copertura della funzione sulla mappa, che minimizzi il numero di implicanti utilizzati ed il numero di letterali.

III) Ricavare il costo della copertura ottenuta, utilizzando come costo il numero di letterali.

Soluzione:

I)



Essenziali:
A = a'c
B = ac'd

Primi:
C = a'd'
D = bc'd'
E = bcd
F = abc'
G = abd

II)

A+B+D+E; A+B+D+G; A+B+E+F; A+B+F+G

III)

- Le soluzioni costano 2(A)+3(B)+3(D o F)+3(E o G)=11.

Esercizio n. 3

Data la seguente tabella di copertura:

	F1					F2					
	m0	m1	m2	m3	m4	m5	m6	m7	m8	m9	Costo
A					X				X		3
B						X	X	X			3
C			X						X		3
D	X	X					X	X			3
E										X	3
F						X				X	3
G								X			2
H			X	X							2
I					X						2
L	X	X									2
M				X							2

- Si trovi una copertura minima utilizzando il metodo di Quine McCluskey (m_{Xn} rappresenta un generico mintermine).
- Descrivere ogni singolo passo svolto per arrivare alla soluzione nella sequenza di applicazione

Soluzione:

F1=H+A+D

F2=F+A+D

PASSI:

1) F domina E -> E eliminato -> F essenziale per F2
2) H domina M -> M eliminato -> H essenziale per F1
3) A domina C -> C eliminato -> A essenziale per F2 -> costo A=1
4) A domina I -> I eliminato -> A essenziale per F2
5) M7 domina M6 -> M7 eliminato
6) D domina B -> B eliminato -> D essenziale per F2 -> costo D=1
7) D domina L -> L eliminato -> D essenziale per F1

Esercizio n. 4

Eseguire la generazione degli implicanti primi con il metodo di Quine McCluskey per la seguente funzione multiuscita F(F1;F2).

F1= on-set(m0, m5, m7, m12, m13)
dc-set(m4,m10,m11)

F2= on-set(m2,m3,m5,m7,m12)
dc-set(m8)

Soluzione:

Rilasso il problema trasformando il DC set in ON-Set

m0	0000	10	V
--			
m2	0010	01	V
m4	0100	10	V
m8	1000	01	V
--			
m3	0011	01	V
m5	0101	11	V
m10	1010	10	V
m12	1100	11	(A)
--			
m7	0111	11	V
m11	1011	10	V
m13	1101	10	V

m0m4	0-00	10	(B)
--			
m2m3	001-	01	(C)
m4m5	010-	10	V
m4m12	-100	10	V
m8m12	1-00	01	(D)
--			
m3m7	0-11	01	(E)
m5m7	01-1	11	(F)
m5m13	-101	10	V
m10m11	101-	10	(G)
m12m13	110-	10	V

m4m5m12m13	-10-	10	(H)

Ritorno al problema iniziale, gli implicanti primi rimangono:

A, B, C, D, E, F, H

G copre solo DC di f1 quindi non è un implicante primo per il problema iniziale.

Esercizio n. 5

- Scrivere la tabella della verità di una rete logica combinatoria che riceve in ingresso i segnali *a*, *b*, *c* e *d* che sulle uscite codifica in binario naturale il numero di ingressi con valore 0.

- Scrivere quindi la tabella della verità modificata nel caso in cui i segnali *a*, *b*, *c* e *d* siano stati codificati in modo tale che il numero di 1 complessivamente presente in ogni configurazione sia sempre dispari.

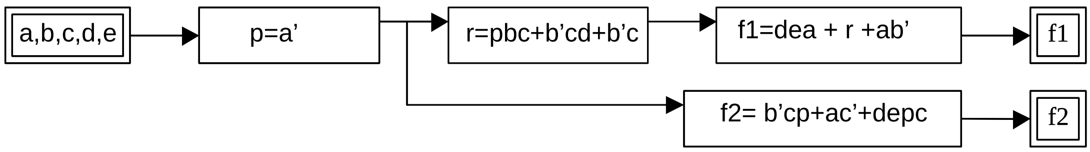
a	b	c	d	$f0$	$f1$	$f2$
0	0	0	0	1	0	0
0	0	0	1	0	1	1
0	0	1	0	0	1	1
0	0	1	1	0	1	0
0	1	0	0	0	1	1
0	1	0	1	0	1	0
0	1	1	0	0	1	0
0	1	1	1	0	0	1
1	0	0	0	0	1	1
1	0	0	1	0	1	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	0	1	0
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	0	0

a	b	c	d	$f1$	$f2$
0	0	0	0	—	—
0	0	0	1	1	1
0	0	1	0	1	1
0	0	1	1	—	—
0	1	0	0	1	1
0	1	0	1	—	—
0	1	1	0	—	—
0	1	1	1	0	1
1	0	0	0	1	1
1	0	0	1	—	—
1	0	1	0	—	—
1	0	1	1	0	1
1	1	0	0	—	—
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	—	—

Nel secondo caso, sono sufficienti due uscite visto che da specifica non è possibile che si presenti l'ingresso con i quattro segnali a 0.

D'altra parte anche l'uscita f2 potrebbe essere ritenuta poco significativa in quanto assume sempre valore 1.

Esercizio n. 6



Data la rete multilivello sopra riportata, applicare in sequenza le trasformazioni sotto indicate e rispondere alle domande dove richiesto. Disegnare anche il modello della rete finale.

Nota Bene: per ogni trasformazione **è obbligatorio** riportare il **risultato della trasformazione** e **mostrare chiaramente tutti i passaggi** effettuati per ottenere il risultato stesso.

1. **COST( ):** Calcolo del numero di letterali. La funzione COST() calcola il costo in letterali indipendentemente dalla forma (SOP o Multilivello) delle espressioni algebriche dei nodi.

2. **SWEEP:** Eliminazione dei nodi costituiti da un solo letterale.
   2a. Mostrare <u>formalmente</u> che il costo della rete ottenuta applicando tale trasformazione è non peggiorativo.

3. **SIMPLIFY(r):** Minimizzazione a due livelli di r.
   3a. Mostrare <u>formalmente</u> che il costo della rete ottenuta applicando tale trasformazione al nodo r è non peggiorativo.

4. **ELIMINATE(r,-2):** Eliminazione vincolata del nodo r. Il parametro –2 indica la soglia di incremento di area per accettare o meno la trasformazione.

5. **FACTOR(f1):** Fattorizzazione del nodo f1.

6. **[s] = EXTRACT(f1,f2):** Estrazione di un fattore comune a f1 e f2. Il nodo s derivato dall’estrazione può essere un nuovo nodo o un nodo già presente nella rete.

7. **COST():** Calcolo del numero di letterali.

Soluzione

1. **COST( ):** 24 letterali

2. **SWEEP:** Viene eliminato il solo nodo p. Quindi:

r = a’bc+b’cd + b’c
f2= b’ca’ + ac’ + dea’c

2a. L’espressione (n*l – n – l) fornisce l’incremento di area in letterali di una rete a seguito dell’eliminazione di un nodo (l è il numero di letterali del nodo eliminato e n è il numero di nodi che lo assorbono). Nel caso di nodi eliminati costituiti da un solo letterale l’incremento di area è sempre pari a –1, qualunque sia il numero di nodi che assorbono.

3. **SIMPLIFY(r):** Minimizzazione a due livelli di r.

Tramite mappe di Karnaugh o manipolazione algebrica ottima, il risultato della minimizzazione dell’espressione r = a’bc+b’cd + b’c è

r = a’c+b’c

3a. L’espressione da minimizzare a due livelli è già una forma SOP, quindi la sua ottimizzazione non può essere peggiorativa (da SOP a SOP minima).

4. **ELIMINATE(r,-2):** Eliminazione vincolata del nodo r. Il parametro –2 indica la soglia di incremento di area per accettare o meno la trasformazione.

Applicando ancora l'espressione per il calcolo di incremento di area n*l – n – l (con l=5, numero di letterali di r e n=1, un solo nodo –f1- assorbe r), l'incremento risulta = -1. E' quindi al di sopra del valore –2 della soglia di accettazione. La trasformazione non viene accettata e le espressioni dei nodi restano quelle del passo precedente.

Lo stesso risultato si poteva ottenere eliminando il nodo e calcolando il nuovo costo della rete.

5. **FACTOR(f1):** Fattorizzazione del nodo f1.

L'algoritmo visto a lezione porta alla fattorizzazione

f1=a(de+b') + r

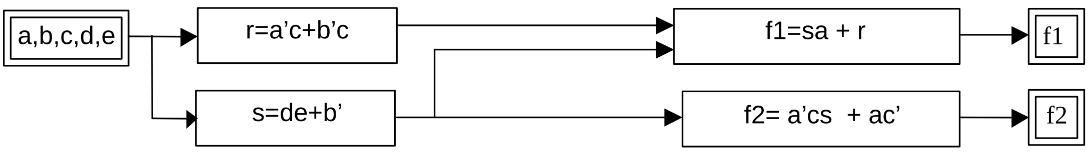
6. **[s] = EXTRACT(f1,f2):** estrazione di un fattore comune a f1 e f2. Il nodo s derivato dall'estrazione può essere un nuovo nodo o un nodo già presente nella rete.

s= de + b'
f1= as + r
f2= a'cs + ac'

7. **COST():** 15 letterali

La rete è infatti composta dai seguenti nodi

r = a'c+b'c
s= de + b'
f1= as + r
f2= a'cs + ac'



Esercizio n. 7

Dati due numeri decimali A=3.546875 e B=0.162109375. Fornire la codifica completa in virgola mobile a singola precisione di A e B.

Effettuare la somma A+B indicando tutti i passaggi relativi sia alla codifica che alla somma.

Soluzione

$3{,}546875_{10}$ = 0 10000000 11000110000000000000000

$0{,}162109375_{10}$ = 0 01111100 01001100000000000000000

Denormalizzo e sommo:

1.11000110000000000000000 +
0.00010100110000000000000 =
1.11011010110000000000000

La codifica normalizzata e'
3.708984375_{10} = 0 10000000 11011010110000000000000

Esercizio n. 8

Data la seguente descrizione VHDL disegnare il circuito logico corrispondente.

```
library IEEE;
use ieee.std_logic_1164.all;

entity l1block is
  port(X: in std_logic_vector(3 DOWNTO 0);
       Z,OK: out std_logic);
end l1block;

architecture mix of l1block is

  component mux42
   port (I0, I1, I2, I3: in std_logic;
         SEL: in std_logic_vector(1 DOWNTO 0);
              O: out std_logic);
  end component;

  signal ZINT, A, B, C, D, E, F:  std_logic;

begin

  u1:   mux42   port   map(I0=>E,I1=>F,I2=>C,I3=>D,SEL=>X(1   DOWNTO   0),
O=>ZINT);

  A <= NOT X(3) AND X(2);
  B <= X(3) AND NOT X(2);
  C <= NOT X(3);
  D <= '1';
  F <= NOT E;
  E <= A OR B;
  Z <= ZINT;
  OK <= NOT (X(3) AND X(2) AND X(1) AND X(0));

end mix;
```