Esercitazione 03 - Soluzioni

1.Si dimostri che (A + ~B)(B + C) = AB + AC + ~BC.

Soluzione:

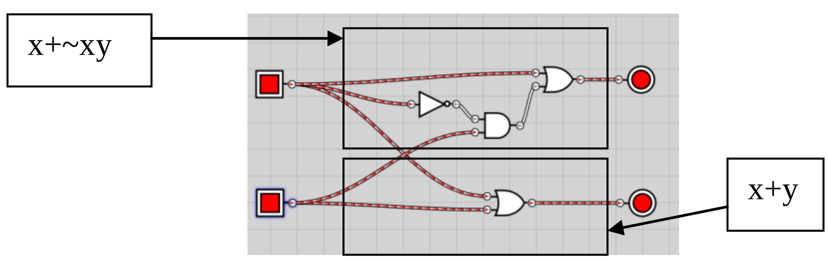
Y	**= (A + ~B)(B + C)**	
	= A(B+C) + ~B(B+C)	*(x+y)z=xz+yz*
	= AB + AC + ~BB + ~BC	*(x+y)z=xz+yz*
	= AB + AC + 0 + ~BC	*~xx=0*
	= AB + AC + ~BC	*x+0=x*

2. Si dimostri che x + ~xy = *x* + *y*. Si implementino in gatesim i due circuiti corrispondenti a x + ~xy e *x* + *y* e si verifichi la correttezza del risultato.

Soluzione:

Y	**= x + ~xy**	
	= x + xy + ~xy	*x=x+xz*
	= x + (x + ~x)y	*xy+xz=x(y+z)*
	= x + 1y	*x+~x=1*
	= x + y	*1x =x*

Il circuito Gatesim:



Il secondo circuito è migliore del primo sia rispetto al cammino critico (2 per il primo, 1 per il secondo) sia rispetto al costo circuitale (2 per il primo, 1 per il secondo).

Massimo Marchi – marchi@dsi.unimi.it

3. **Si ricavi la tabella della verità delle seguenti funzioni: A+B+C, A+B+C+D (or a 3 e 4 ingressi). Si implementi il modulo corrispondente in Gatesim e lo si salvi. Si faccia lo stesso per le funzioni AND a 3 e 4 ingressi.**

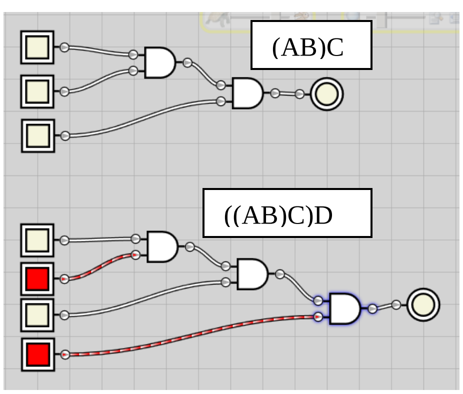
Soluzione:

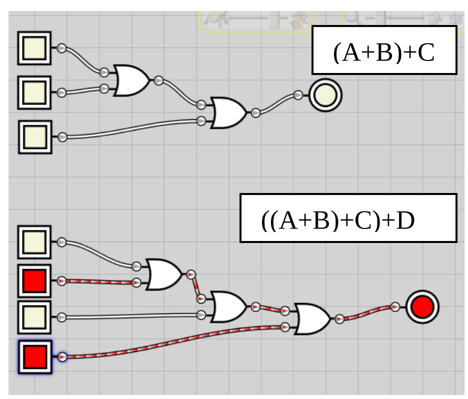
Indicando esplicitamente la precedenza tra gli operatori, le formule diventano:

(A+B)+C , ((A+B)+C)+D) , (AB)C , ((AB)C)D)

Di seguito sono riportate raggruppate per brevità tutte e quattro le tabelle:

A	B	C	D	A+B+C	A+B+C+D	ABC	ABCD
0	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0
0	1	0	0	1	1	0	0
1	1	0	0	1	1	0	0
0	0	1	0	1	1	0	0
1	0	1	0	1	1	0	0
0	1	1	0	1	1	0	0
1	1	1	0	1	1	1	0
0	0	0	1		1		0
1	0	0	1		1		0
0	1	0	1		1		0
1	1	0	1		1		0
0	0	1	1		1		0
1	0	1	1		1		0
0	1	1	1		1		0
1	1	1	1		1		1

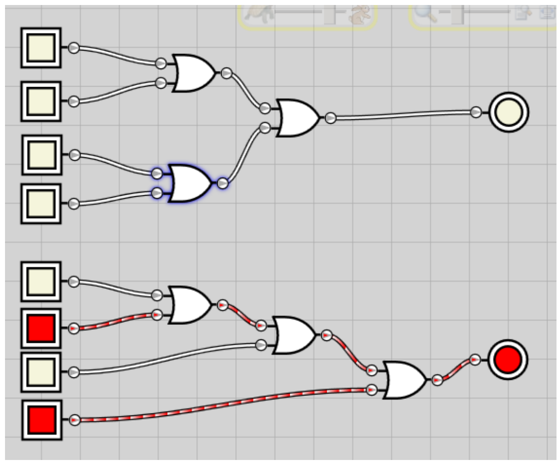




4. **Si confrontino (se necessario) le tabelle della verità di A+B+C+D e (A+B)+(C+D). Si confrontino i due circuiti equivalenti. Quale circuito risulta essere più vantaggioso da implementare è perchè? Si faccia lo stesso confrontando ABCD e (AB)(CD). Si rivedano i circuiti salvati al punto 3 di conseguenza.**

Soluzione:

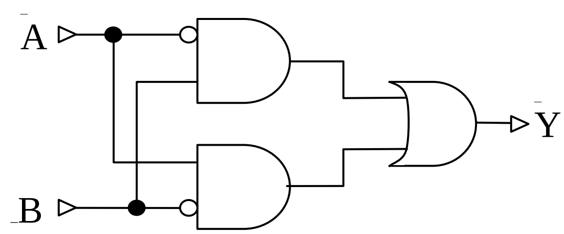
Il circuito Gatesim:



Si nota che il cammino critico di (A+B)+(C+D) è 2 mentre il cammino critico di ((A+B)+C)+D è 3. In entrambi i casi il costo è 3. Questo comportamento è valido in generale, per ogni numero di ingressi: è possibile ottenere un cammino critico minore rispetto all'implementazione in cascata riorganizzando le porte, pur mantenendo lo stesso costo.

Rappresentazioni possibili per una funzione logica:

- **circuito logico:**



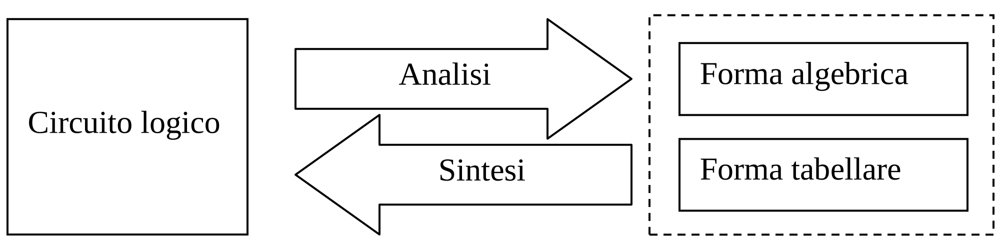
- **forma tabellare** (tabella lookup):

A	**B**	**Y**
0	0	0
0	1	1
1	0	1
1	1	0

- **formula algebrica:**

Y= (NOT A)B OR NOT(B) A

Sintesi e analisi di circuiti logici:



Precedenza degli operatori logici:

In assenza di parentesi, AND ha la priorità sull'OR ed il NOT su entrambi:

NOT > AND > OR

Principio di dualità:

Il duale di una funzione si ottiene sostituendo:
AND con OR,OR con AND, 0 con 1 ed 1 con 0.

Proprietà generali dell'Algebra Booleana:

0.	**Doppia Inversione**	$\sim(\sim x) = x$	
1.	**Identità**:	$1\,x = x$	$0 + x = x$
2.	**Elemento nullo**:	$0\,x = 0$	$1 + x = 1$
3.	**Idempotenza**:	$x\,x = x$	$x + x = x$
4.	**Inverso:**	$x \sim x = 0$	$x + \sim x = 1$
5.	**Commutativa**:	$x\,y = y\,x$	$x + y = y + x$
6.	**Associativa:**	$(x\,y)\,z = x\,(y\,z)$	$(x + y) + z = x + (y + z)$
7.	**Distributiva:**	$x\,(y + z) = x\,y + x\,z$	$x + y\,z = (x + y)\,(x + z)$
8.	**Assorbimento:**	$x\,(x + y) = x$	$x + x\,y = x$
9.	**De Morgan**:	$\sim(x\,y) = \sim x + \sim y$	$\sim(x + y) = \sim x \;\sim y$

Forma Canonica SOP *(Sum Of Product)* :

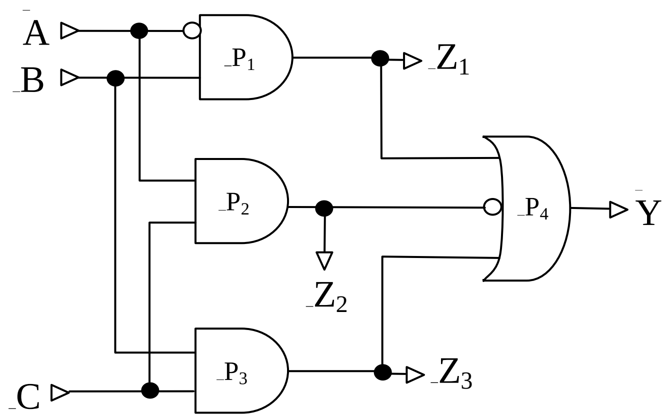
Implicante → Prodotto di variabili, semplici o negate, che vale 1 per combinazioni di valori per cui la funzione data vale 1 (*se implicante vale 1 allora la funzione vale 1, il contrario può non valere*)

Mintermine → Implicante contenente tutte le variabili della funzione data

Forma **SOP** di F → $F = \sum_{i=1}^{Q} m_j$ dove m_j è il j-esimo mintermine della funzione F

Ex: **A** XOR **B** = $\overline{\mathbf{A}}\,\mathbf{B} + \mathbf{A}\,\overline{\mathbf{B}}$

7 Ricavare la forma tabellare , la prima forma canonica e la forma algebrica del seguente circuito semplificando dove possibile.



Nel ricavare la forma tabellare dal circuito logico conviene procedere dagli ingressi verso le uscite,individuando ogni punto di calcolo intermedio e l'ordine in cui questi calcoli saranno disponibili e quindi utilizzabili

Costruite una tabella con una riga per ogni possibile combinazione degli ingressi ed una colonna con i risultati intermedi per l'uscita di ogni porta logica, ev. aggiungendo ulteriori colonne per il risultato negato (come $\sim Z_2$ nell'esempio).

AND — OR

A	B	C	~A	Z_1=~AB	Z_2=AC	Z_3=BC	~Z_2	**Y**=Z_1+~Z_2+Z_3
0	0	0	1	0	0	0	1	1
0	0	1	1	0	0	0	1	1
0	1	0	1	1	0	0	1	1
0	1	1	1	1	0	1	1	1
1	0	0	0	0	0	0	1	1
1	0	1	0	0	1	0	0	0
1	1	0	0	0	0	0	1	1
1	1	1	0	0	1	1	0	1

Nota: *una porta AND dà come risultato 0 quando uno dei suoi ingressi è 0. Per calcolare una colonna risultato di una AND allora è comodo procedere nel seguente modo: per ogni termine si identificano le celle a 0 e si pone a 0 la cella risultato corrispondente. Alla fine le celle ancora vuote si pongono ad 1. Analogamente il metodo duale può essere applicato alle porte OR: prima si identificano le celle risultato a 1 corrispondenti ai termini posti uguale a 1 quindi si completano le celle ancora vuote con 0.*

Es.: Si consideri il calcolo di Z_1. Il termine ~A va a zero per le ultime quattro configurazioni. Ne segue che Z_1 può essere posto a zero per le corrispondenti celle. Analogamente B è uguale a zero per le prime due configurazioni e per la 5^{ta} e la 6^{ta} . Ne segue che Z_1 può essere messo a 0 anche per le prime due celle. Le restanti celle saranno obbligatoriamente uguali ad 1.

Nota: Il numero di mintermini nella forma canonica SOP è pari al numero di 1 nella colonna risultato. Viceversa il numero di maxtermini presenti nella seconda forma canonica POS è uguale al numero di 0 presenti. In questo caso quindi sarebbe più conveniente in termini di compattezza di descrizione usare la seconda forma canonica POS al posto della forma SOP.

Massimo Marchi – marchi@dsi.unimi.it

Calcoliamo una forma algebrica semplificata partendo dal circuito logico:

Nel ricavare la formula dal circuito logico è comodo ricostruire la formula partendo dalle uscite e risalendo poi il circuito fino agli ingressi.

NOTA: Per ogni passaggio di semplificazione è indicato il punto di lavoro con un evidenziazione gialla e il risultato ottenuto con una zona sottolineata nella riga successiva.

Y = ← *Sviluppo la porta P_4*

= (Z_1 + ~Z_2 + Z_3) ← *Sviluppo le porte P_1, P_2 e P_3*

= (~AB + ~(AC) + BC) ← *9a De Morgan:* ~(**xy**) = ~**x** + ~**y**

= (~AB + ~A + ~C + BC) ← *8b Assorbimento:* **xy** + **x**= ~**x** (**x**=~*A*,**y**=*B)*

= (~A + ~C + BC)

Dim: **x** + ~**xy** = (**x** + **xy**) + ~**xy** ←*8b*
= ***x*** + (***xy*** + ~***xy***) ←*6b*
= ***x*** + ***y(x*** + ~***x)*** ←*7a*
= ***x*** + ***y1*** ←*4b*
= ***x*** + ***y*** ←*1a*

Vale la seguente proprietà: **x** + ~**xy** = **x** + **y**

Questo *non è altro che l'unico maxtermine della seconda forma canonica.*

= (~A + ~C + BC) ← *Applico:* **x** + ~**xy** = **x** + **y**

= (~A + ~C + B) ← *8b De Morgan*

= ~ (A~BC)

Questo *non è altro che l'unico mintermine della prima forma canonica della funzione ottenuta negando quella data.*

Calcoliamo lo stesso risultato partendo dalla tabella la forma SOP e semplificando:

La prima forma canonica si ottiene sommando i mintermini della funzione risultato. Ad ogni combinazione degli ingressi per cui la funzione vale 1 corrisponde un mintermine.

Y = ~A~B~C + ~A~BC + ~AB~C + ~ABC + A~B~C + AB~C + ABC

Semplifichiamo applicando varie volte le regole 7a e 4b:

x y + ~***x y*** = (***x*** + ~***x***) ***y*** = ***1 y*** = ***y***

Questo corrisponde ad individuare di volta in volta degli *implicanti* (**y**) sempre più piccoli della funzione data

Y = ~A~B~C + ~A~BC + ~AB~C + ~ABC + A~B~C + AB~C + ABC

Y = ~A~B + ~AB~C + ~ABC + A~B~C + AB~C + ABC

Y = ~A~B + ~AB + A~B~C + AB~C + ABC

Y = ~A~B + ~AB + A~C + ABC

Y = ~A + A~C + ABC ← *6b Associativa:* **A**(**w**)+**A**(**z**) = **A**[(**w**) +(**z**)]

Y = ~A + A(~C + BC) ← **x** + ~**xy** = **x** + **y**: **x**=~*C* , ~**x**=~*(~C)=C*

Y = ~A + A(~C + B) ← **x** + ~**xy** = **x** + **y**: **x**=~*A* , ***y****=(~C+B)*

Y = ~A + ~C + B = ~(A~BC) ← *Vedi sopra*

Massimo Marchi – marchi@dsi.unimi.it