Università degli Studi di Milano «La Statale»
Dipartimento di Informatica

Architettura degli elaboratori - Lez 5:

# Porte logiche e circuiti combinatori (part B)

Marco Tarini
marco.tarini@unimi.it

68

## Regole di trasformazione di espressioni

| Legge | **con OR** (duale) | **con AND** |
|---|---|---|
| Elem. neutro | 0 + A = A | 1·A = A |
| Elem. nullo | 1 + A = 1 | 0·A = 0 |
| Idempotenza | A + A = A | A·A = A |
| Inverso | A + /A = 1 | A · /A = 0 |
| Commutativa | A + B = B + A | A · B = B · A |
| Associativa | (A + B) + C = A + (B + C) | (A·B)·C = A·(B·C) |
| Distributiva | A · (B + C) = A·B + A·C | A + B·C = (A + B) · (A + C) |
| Assorbimento 1 | A + \A·B = A + B | A · (\A + B) = A · B |
| Assorbimento 2 | A + A·B = A | A·(A + B) = A |
| De Morgan | /(A + B) = /A · /B | /(A·B) = /A + /B |
| Tertium non datur | / / A = A | |



71

## Assorbimenti: explained (le spiegaziponi intuitive»

- A + \A·B = A + B
  «vale A, oppure, A non vale ma vale B» ⇔ «vale A oppure vale B»
- A · (\A + B) = A · B
  «vale A, e inoltre, vale uno fra non A e B» ⇔ «valgono sia A che B»
- A + A·B = A
  «vale A, oppure, valgono sia A che B» ⇔
  «vale A – che valga anche B oppure no» ⇔
  «vale A»
- A·(A + B) = A
  «vale A e, inoltre, vale almeno uno fra A e B» ⇔
  «vale A – non importa se vale anche B» ⇔
  «vale A»



## Esercizio: semplificare l'espressione booleana

$$\neg(x \lor (\neg x \land y) \lor \neg(x \land y)) \lor \neg(x \lor \neg y)$$

cioè

$$\overline{x + \bar{x}\, y + \overline{x\, y}} + \overline{x + \bar{y}}$$

$$\overline{x + \bar{x}\, y + \overline{x\, y}} + \overline{x + \bar{y}} =$$

$$\overline{x + \bar{x}\, y + \bar{x} + \bar{y}} + \overline{x + \bar{y}} =$$

$$\overline{\bar{x}\, y + x + \bar{x} + \bar{y}} + \overline{x + \bar{y}} =$$

$$\overline{\bar{x}\, y + 1 + \bar{y}} + \overline{x + \bar{y}} =$$

$$\bar{1} + \overline{x + \bar{y}} =$$

$$\cancel{0 +} \overline{x + \bar{y}} =$$

$$\bar{x}\ \bar{\bar{y}} =$$

$$\bar{x}\ y$$

74

## Esercizio: semplificare l'espressione booleana

$$\neg(x \lor (\neg x \land y) \lor (\neg x \land \neg y)) \lor \neg(x \land \neg y)$$

cioè

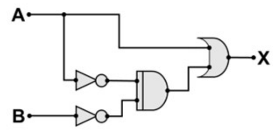
$$\overline{x + \bar{x}\, y + \bar{x}\, \bar{y}} + \overline{x + \bar{y}}$$



75

## da **Tabella di vertità** ad **Espressione booleana**

implementabile in un circuito!




Rete combinatoria

Implementazione

$$A + /A\,/B$$

Espressione booleana

SINTESI
di espressioni

| A | B | X |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

Tabella verità
(funzione booleana)



## da **Tabella di vertità**
## ad **Espressione booleana**

- Vediamo ora come, data una tabella di verità (cioè un modo per esprimere una qualsiasi funzione booleana) possiamo costruire facilmente un'espressione booleana che computa quella funzione
- Vedremo un modo che produce un'espressione detta in "prima forma canonica" (o somma di prodotti)
  - Esiste anche un'alternativa, detta "seconda forma canonica" (o prodotto di somme) che ci limiteremo ad accennare
- Questa espressione potrà essere poi implementata in un circuito combinatorio (come sappiamo già)
  - Quindi, possiamo scrivere un circuito di qualsiasi funzione
- Nota: non è detto che l'espressione prodotta in questo modo sia la "migliore possibile" che implementa la funzione data
  - Anzi, di solito, non lo è
  - Quindi, è conveniente ottimizzarla (tramite trasfromazioni di espressioni) per ottenere un circuito migliore

## Sintesi come **somma di prodotti** (SdP) (cioè in «**prima forma canonica**»)

- Input: la tabella delle verità della funzione da sintetizzare,
- Output: una somma di prodotti, cioè un'espressione booleana del tipo
  XXXX + YYYY + ZZZZ + ….
  dove ciascuno degli addendi della somma (XXXX, YYYY … )
  è il prodotto di tutte le variabili (negata o no), detta **mintermine**.



## Sintesi come **somma di prodotti**: spiegazione intuitiva (con un esempio)

- F(A, B) =
  /A /B
  + /A B
  + A B

| A | B | OUTPUT |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

- F=1 se e solo se *uno qualsiasi dei casi 1 si verifica*, cioè quando…
  - si verifica il primo:
    A vale 0 e B vale 0

…*oppure*…

  - si verifica il secondo:
    A vale 0 e B vale 1

…*oppure*…

  - si verifica il terzo:
    A vale 1 e B vale 1

## Esempio: funzione maggioranza

- Si chiede di sintetizzare una funzione combinatoria dotata di 3 ingressi A, B e C, e di un'uscita,
  definita (a parole) come segue:
  - Se la maggioranza degli ingressi vale 0, l'uscita valga 0
  - Se la maggioranza degli ingressi vale 1, l'uscita valga 1



## Esempio: funzione maggioranza
## Tabella delle verità

- Primo passo,
  scriviamo la tabella delle verità
- E' quella mostrata a lato
- L'uscita vale 1 se e solo se 2 o tutti e 3 gli ingressi valgono 1 (cioè se e solo se il valore 1 è in maggioranza)

| A | B | C | OUTPUT |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

## Esempio: funzione maggioranza
## Sintesi espressione (in 1ma forma canonica)

- F(A, B, C) =
  /A B C +
  A /B C +
  A B /C +
  A B C

| # riga | A | B | C | F |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 1 | 1 |
| 4 | 1 | 0 | 0 | 0 |
| 5 | 1 | 0 | 1 | 1 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 1 |

- E' una somma di prodotti
- Il passaggio successivo è quello di **implementare** questa espressione in un circuito
- Vediamo un modo semplice di farlo (per le Somme di Prodotti)

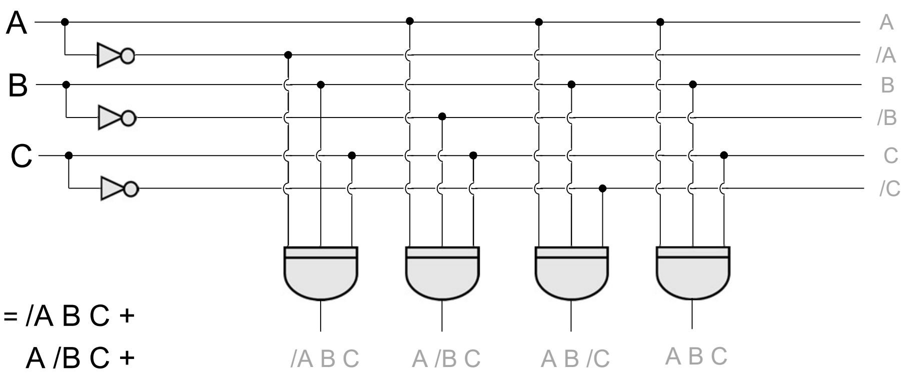




## Modo pratico di disegnare un circuito per una somma di prodotti (step 1)


X = /A B C +
A /B C +
A B /C +
A B C

## Modo pratico di disegnare un circuito per una somma di prodotti (step 2)

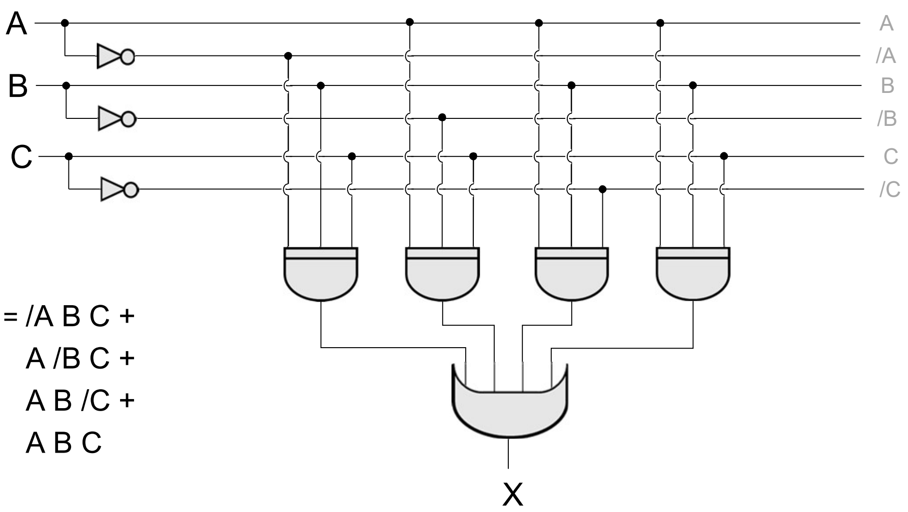



X = /A B C +
A /B C +
A B /C +
A B C





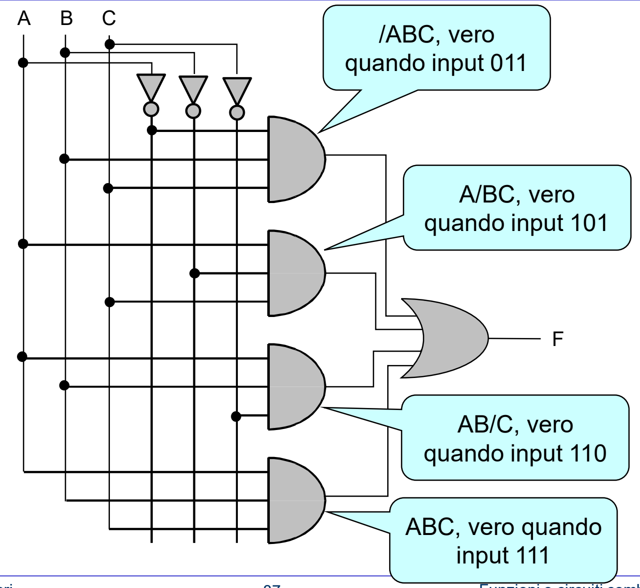
## Modo pratico di disegnare un circuito per una somma di prodotti (step 3)




X = /A B C +
A /B C +
A B /C +
A B C

## Modo pratico di disegnare un circuito per una somma di prodotti: una variante

F(A, B, C) =
/A B C +
A /B C +
A B /C +
A B C






87

## OPZIONALE: 2nda forma canonica. Cioè prodotto di somme. Un esempio

- F(A, B) =
  (A+B)
  (A+/B)
  (/A+B)

| A | B | F |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

- F=1 se e solo se *nessuno dei casi 0 si verifica*, cioè quando…
  - non si verifica il primo: A non 0 oppure B non 0

…*e inoltre*…

  - non si verifica il secondo: A non 0 oppure B non 1

…*e inoltre*…

  - non si verifica il terzo: A non 1 oppure B non 0



89

## OPZIONALE: 2nda forma canonica. Prodotto di somme. Esempio a tre ingressi

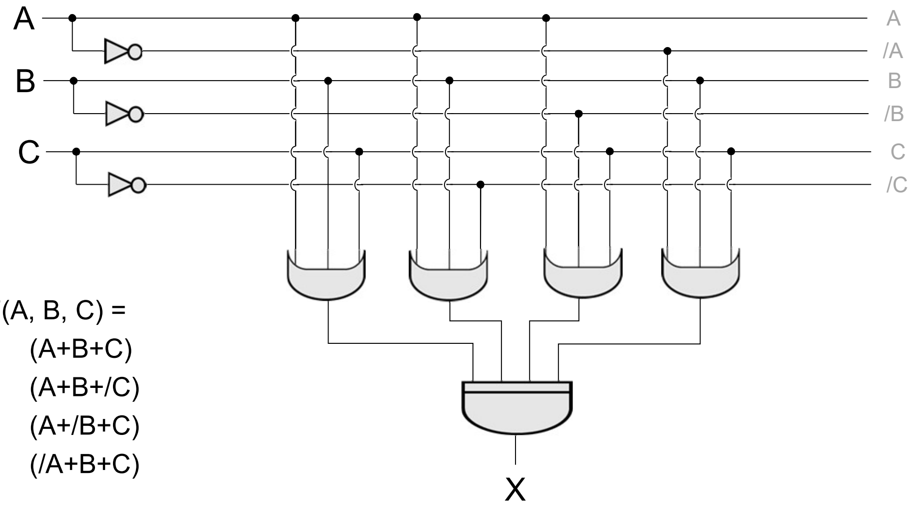
- F(A, B, C) =
  (A+B+C)
  (A+B+/C)
  (A+/B+C)
  (/A+B+C)

- Nota bene: è un prodotto di somme

| # riga | A | B | C | F |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 1 | 1 |
| 4 | 1 | 0 | 0 | 0 |
| 5 | 1 | 0 | 1 | 1 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 1 |



## OPZIONALE: 2nda forma canonica. Come disegnarlo




F(A, B, C) =
(A+B+C)
(A+B+/C)
(A+/B+C)
(/A+B+C)

## Quale metodo conviene?

- PoS o SoP?
- In genere
  - se ci sono pochi 1 conviene SoP
  - se ci sono pochi 0 conviene PoS

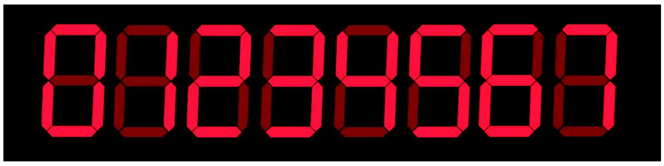


## Costo di realizzazione di un circuito digitale

- Il costo di un circuito digitale si valuta in vari modi (criteri di costo):
  - Numero totale di porte utilizzate
  - Numero di tipi diversi di porte utilizzate
  - Altri criteri:
    - Numero di tipi di porte *diverse* usate (NAND o NOR)
    - Numero totale di transistor (per comporre tutte le porte necessarie)
      ES:
      ogni AND oppure OR : 4 transistor
      ogni NOT : 2 transistor
    - Lunghezza e complessità delle interconnessioni (wires)
    - ecc ...
- Dal punto di vista economico, subentrano molti altri fattori
  - Scala della produzione (produzione di massa = meno $$$)
  - Tecnologia utilizzata

## Esempio: il “7 segment display” per cifre in base 8

- costruiamo un piccolo circuito combinatorio che calcola lo stato (acceso, spento) di uno dei 7 led del familiare display
- dobbiamo far apparire sul display una cifra in base 8 (da 0 a 7) codificata da tre bit ABC in input  (da $000_2$ a $111_2$)
- proviamo prima con uno dei sette led (quello in basso a sinistra)



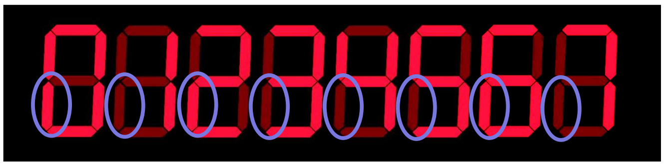
000 ABC   001 ABC   010 ABC   011 ABC   100 ABC   101 ABC   110 ABC   111 ABC





## Esempio dell’intero processo 1/5: scriviamo la tabella di verità

Acceso   Spento   Acceso   Spento   Spento   Spento   Acceso   Spento



000 ABC   001 ABC   010 ABC   011 ABC   100 ABC   101 ABC   110 ABC   111 ABC

tabella di veritá

| A | B | C | X |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

## Esempio dell'intero processo 2/5: espressione in forma normale

tabella
di veritá

| A | B | C | X |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

1ma forma normale (Somma di Prodotti):

$$\overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,B\,\overline{C} + A\,B\,\overline{C}$$

(3 uno, 3 mintermini)

Avremmo anche potuto usare la 2da forma, ma in questo caso viene più complessa:

$$(A+B+\overline{C})(A+\overline{B}+\overline{C})(\overline{A}+B+\overline{C})(\overline{A}+\overline{B}+C)(\overline{A}+\overline{B}+\overline{C})$$

(5 zero, 5 mintermini)



## Esempio dell'intero processo 3/5: riscrittura dell'espressione

$$\overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,B\,\overline{C} + A\,B\,\overline{C}$$

$=$ (distributiva)

$$\overline{A}\,\overline{B}\,\overline{C} + (\overline{A} + A)\,B\,\overline{C}$$

$=$ (inverso)

$$\overline{A}\,\overline{B}\,\overline{C} + 1\,B\,\overline{C}$$

$=$ (el. neutro)

$$\overline{A}\,\overline{B}\,\overline{C} + B\,\overline{C}$$

$=$ (distributiva)

$$(\overline{A}\,\overline{B} + B)\,\overline{C}$$

$=$ (assorbimento)

$$(\overline{A} + B)\,\overline{C}$$

## Esempio dell'intero processo 4/5
## Verifica: l'espressione ottenuta computa ancora stessa funzione?

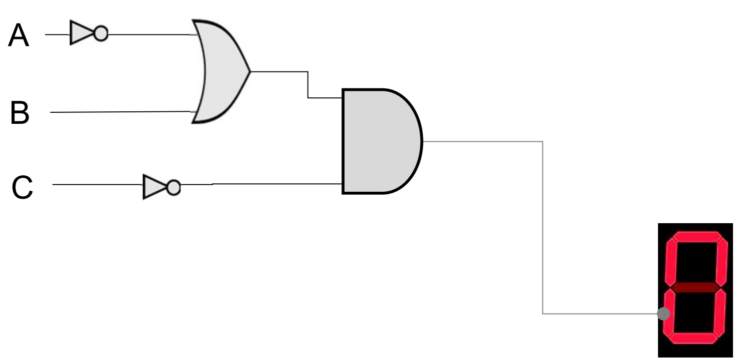
| A | B | C | \A | \A+B | \C | (\A+B) x \C |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 |

Sì!



## Esempio dell'intero processo 5/5: sintesi della rete

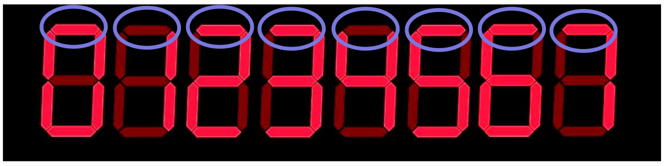
- Rete combinatoria per: (\A+B) x \C
-



## Esempio dell'intero processo 1/4

- Risolviamo per il led in alto

acceso spento acceso acceso spento acceso acceso acceso



000 ABC | 001 ABC | 010 ABC | 011 ABC | 100 ABC | 101 ABC | 110 ABC | 111 ABC

tabella di veritá

| A | B | C | X |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |





## Esempio dell'intero processo 2/4

| A | B | C | X |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

- Come SOMMA DI PRODOTTI (1ma forma canonica)

\A \B \C + \A B \C + \A B C + A \B C + A B \C + A B C

- Come PRODOTTO DI SOMME (2da forma canonica)

( A + B + \C ) ( \A + B + C)

(usiamo questa! meno 0 che 1)

## Esempio dell'intero processo 3/4

- Ottimizzazione:

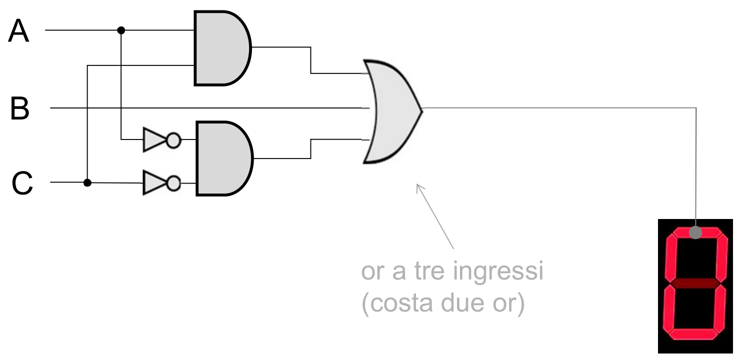
```
                              // start: 5 operatori binari, 2 not
( A + B + \C  ) ( \A + B + C)
            =                 // distributiva:  ( X + Y ) ( X + Z ) = X + YZ
B + ( A + \C  ) ( \A + C)
            =                 // distr (2 volte):  X( Z + K ) = XZ+XK
B + A \A + AC + \C \A + C \C
            =                 // inverso (2 volte):     X \X = 0
B +  0   + AC + \C \A + 0
            =                 // elem neutro
B + AC + \A \C
                              // end: 4 opeatori binari, 2 not
```





## Esempio dell'intero processo 4/4

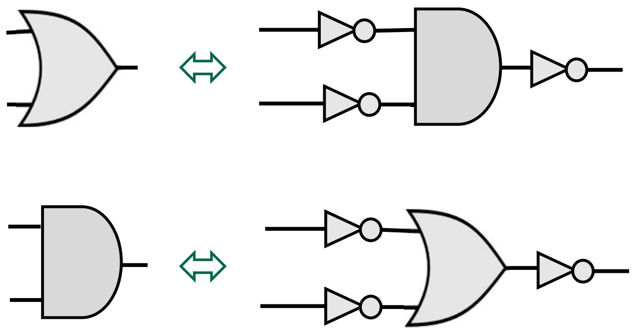
- Rete combinatoria per: B + AC + \A\C



## De Morgan: una conseguenza

- DeMorgan ci mostra che possiamo, se lo vogliamo, fare a meno di porte **OR** .
  - potremmo sempre sostituirle con porte **AND** (più alcuni **NOT**)
- e viceversa!

ecco le leggi
di De Morgan…
…a circuito:

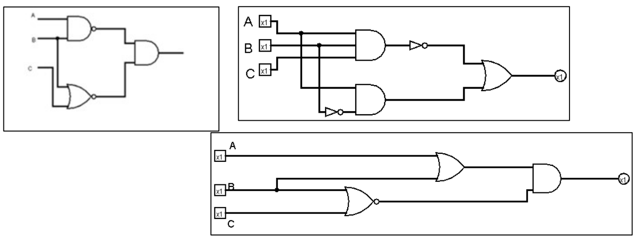




111

## Esercizi di riepilogo 1/2

- 1) Analisi: definire la funzione booleana corrispondente ai circuiti:




- Sintesi: Costruire il circuito non ottimizzato che permette di determinare se un numero espresso su 3 bit in ingresso è minore o uguale a 3
  Poi ottimizza l'espressione (verificando anche che sia ancora la stessa funzione),
  e ripeti il diesgno del circuito
- Ripeti l'esercizio dei led un led a scelta dei sette LED
- Provare ora con il led già visto, ma pasando alla base 10?
  (nota: serviranno 4 bit in input!)

112

## Esercizi di riepilogo 2/2

Semplificare le espressioni booleane:

- (i) $\overline{A + \overline{A}B + \overline{AB}} + \overline{A + \overline{B}}$
- (ii) $\overline{AC} + \overline{A}BC + \overline{B}C$

Costruire la tabella di verita' per la seguente funzione ed esprimere la funzione come somma di prodotti:

$(XY + Z)(Y + XZ)$