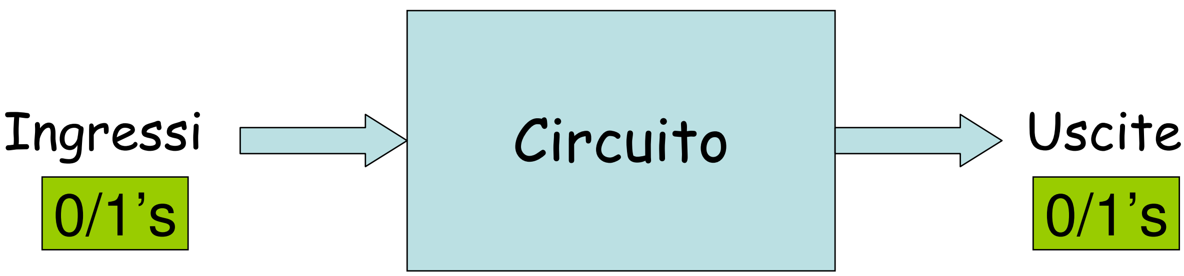
Parte III

Hardware

Circuiti Digitali



- Un circuito (digitale) puo' essere descritto
  - Mostrando i dettagli realizzativi in termini di circuiti elementari (porte)
  - Mediante una tavola di verita' che mostra i valori in uscita x tutti i possibili ingressi

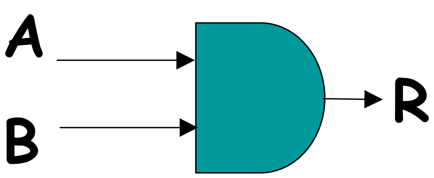
L' Hardware di un computer

I componenti di un computer sono realizzati con un **gran numero** di componenti elettroniche "molto semplici" dette *porte logiche*

Solamente 3 tipi di porte (base):

- AND ("e")
- OR ("o")
- NOT ("non")

AND



Fornisce tensione all'output **R** se e solamente se vi e\` tensione in **entrambi** gli input **A** e **B**

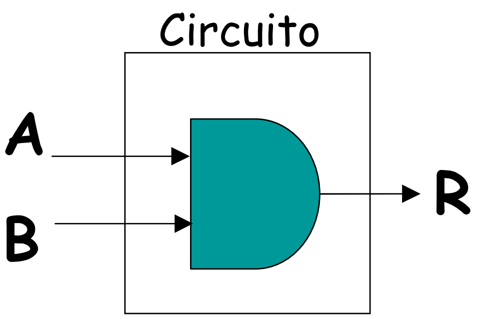
A	B	R
0	0	0
0	1	0
1	0	0
1	1	1

Tavola di verità

1 = VERO, 0 = FALSO

1 = tensione, 0 = no tensione

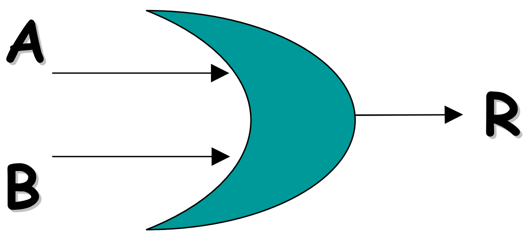
tavola di verità ⇔ circuito



A	B	R
0	0	0
0	1	0
1	0	0
1	1	1

- Dato un qualsiasi circuito e' sempre possibile definire la tavola di verita' (in un solo modo)
- Data una tavola di verita' si possono costruire in generale piu' circuiti che la realizzano

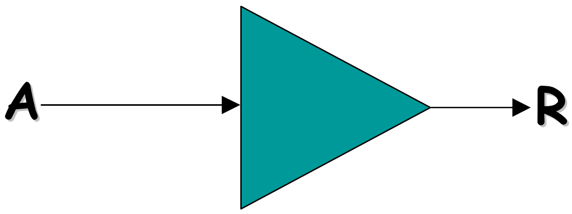
OR



A	B	R
0	0	0
0	1	1
1	0	1
1	1	1

Fornisce tensione all'output **R** se e solamente se vi e` tensione in **almeno uno** degli input **A** e **B**

NOT



A	R
0	1
1	0

Fornisce tensione all'output **R** se e solamente se **non** vi e` tensione all'input ***A***

Assemblando queste componenti possiamo costruire nuovi circuiti piu'complessi..

IMPLICAZIONE LOGICA

se *A* allora B $A \Rightarrow B$

ogni volta che *A* e' VERO anche B deve essere VERO

Tavola di verita'

$A \Rightarrow B$

A	B	$A \Rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

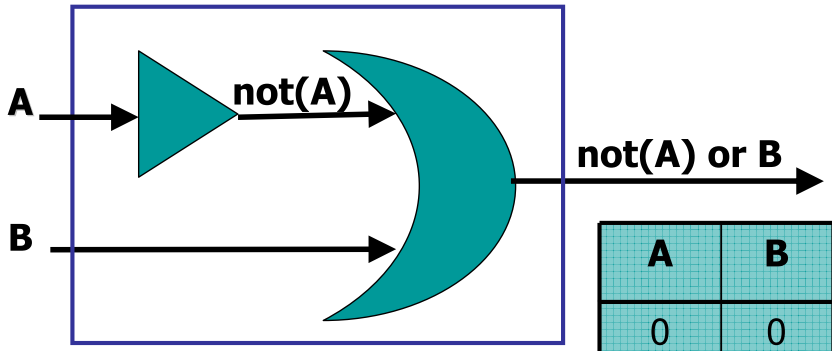
Come costruire un circuito che realizza questa tavola di verita`?

A	Ris
0	0/1
1	0/1

A	B	Ris
0	0	0/1
0	1	0/1
1	0	0/1
1	1	0/1

Ci sono 4 = $2^{(2^1)}$ possibili tabelle di verita` ad una entrata e 16= $2^{(2^2)}$ possibili tabelle di verita` a due entrate.
In generale, ci sono $2^{(2^N)}$ tabelle di verita` con N entrate.
I circuiti corrispondenti **si possono tutti realizzare** componendo i circuiti elementari AND, OR e NOT .

Il seguente circuito realizza l'implicazione logica $A \Rightarrow B$



A	B	not(A)	not(A) or B
0	0	1	1
0	1	1	1
1	0	0	0
1	1	0	1

Infatti vale che

$A \Rightarrow B \cong$ not(A) or B

Si puo' inoltre dimostrare che vale il seguente fatto:
ogni tavola di verita` si puo' realizzare componendo un solo circuito elementare tra i due seguenti:
NAND, cioe` **not (A and B)**
NOR, cioe` **not(A or B)**

Dimostrazione per nand:
Si puo` dimostrare che

- A and B $\cong$ (A nand B) nand (A nand B)
- A or B $\cong$ (B nand B) nand (A nand A)
- not(A) $\cong$ A nand A

Esercizio: Verificare che le tabelle di verita' coincidono nei tre casi sopra.

Quindi, una CPU si puo` realizzare stampando su silicio una griglia di milioni di porte logiche tutte uguali: NAND o NOR.

Le moderne CPU sono realizzate in questo modo. Ogni porta logica di una CPU restituisce il risultato in un ciclo di clock.

Equivalenza Logica

A ⇔B Ogni volta che A è VERO anche B è VERO e viceversa.

Ossia, A e' VERO se e solo se B e' VERO

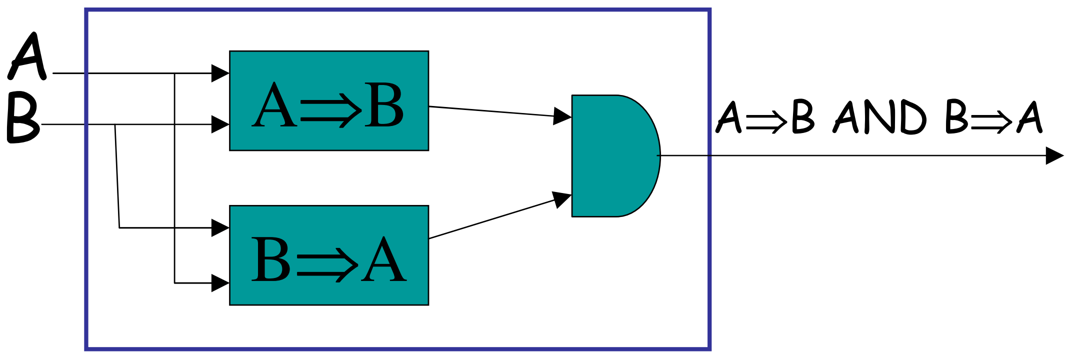
A	B	A⇔B
0	0	1
0	1	0
1	0	0
1	1	1

Notiamo che l'equivalenza è una doppia implicazione:

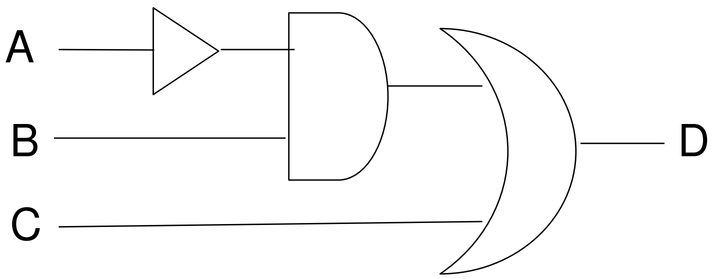
$$A \Leftrightarrow B \quad \cong \quad (A \Rightarrow B) \quad \text{AND} \quad (B \Rightarrow A)$$

A	B	A⇒B	B⇒A	A⇒B AND B⇒A
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	1	1	1

Possiamo quindi realizzare l'equivalenza logica tramite il seguente circuito



Come ottenere una tabella di verita' dato un circuito?



A	B	C	D
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

D = ………………………

Due Modi

- Modo 1) Si calcola la formula logica corrispondente al circuito e si calcolano le tabelle di verita' partendo dalle formule intermedie piu' semplici fino alla formula data

- Modo 2) Si calcola, per ogni possibile configurazione degli ingressi, l'uscita delle porte fino alle uscite del circuito

Come si costruisce un circuito che implementi una tavola di verita'?

Il modo + semplice

Passo 1) Per ogni uscita nella tabella uguale a 1, si crea un 'circuito riconoscitore' della corrispondente configurazione di ingressi, ovvero una AND tra gli ingressi, negati quando uguali a zero

Passo 2) OR tra le uscite delle AND costruite al passo precedente

Chiaramente il circuito cosi' costruito non necessariamente e' il piu' semplice (ovvero con il minor numero possibile di porte logiche utilizzate)

Riassumendo: Si possono comporre circuiti semplici per ottenenere circuiti che realizzano nuove operazioni piu` complesse

Ma cosa si riesce a fare con le porte logiche?

Somma tra numeri binari (Ripasso)

La somma di due numeri interi in rappresentazione binaria si effettua in modo analogo alla somma di interi in rappresentazione decimale: si sommano a due a due le cifre corrispondenti a partire da destra e se la somma e` maggiore o uguale alla base (2 per la rappresentazione binaria, 10 per la rappresentazione decimale, ecc.) si ha un **riporto** che si deve sommare alle due cifre successive.


```
1    11      riporti
 1010011 +
 1100011 =
----------
10110110
```

Proviamo a costruire un circuito per sommare due numeri interi

```
10000110    riporti
 1010011 +
 1100011 =
----------
10110110
```

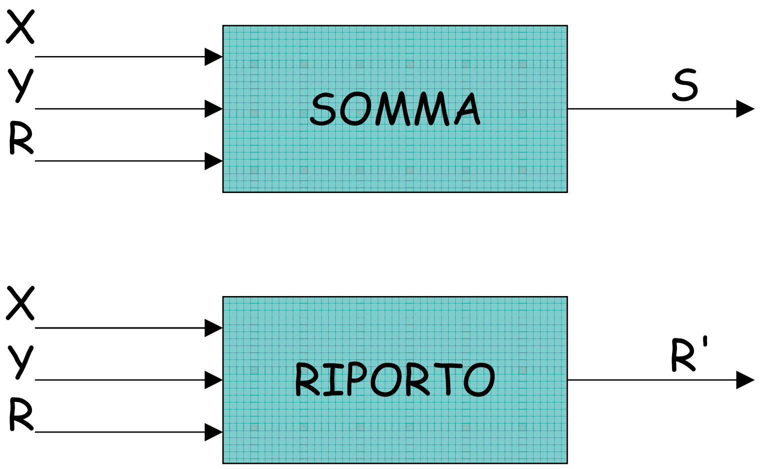
Iniziamo con un circuito che faccia la somma su una colonna

Abbiamo tre cifre binarie X, Y, R in input mentre in output vogliamo ottenere la somma S ed il riporto R'

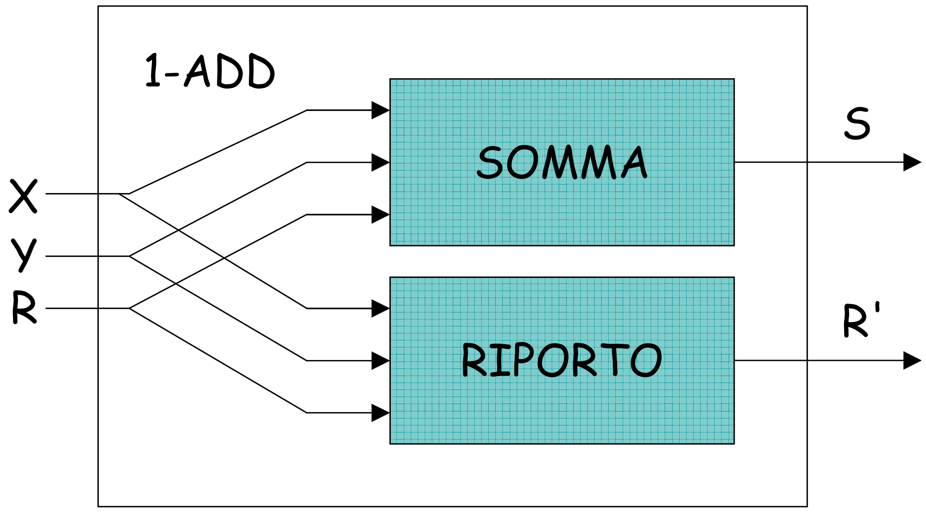
Tabella di verità

X	Y	R	S	R'
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

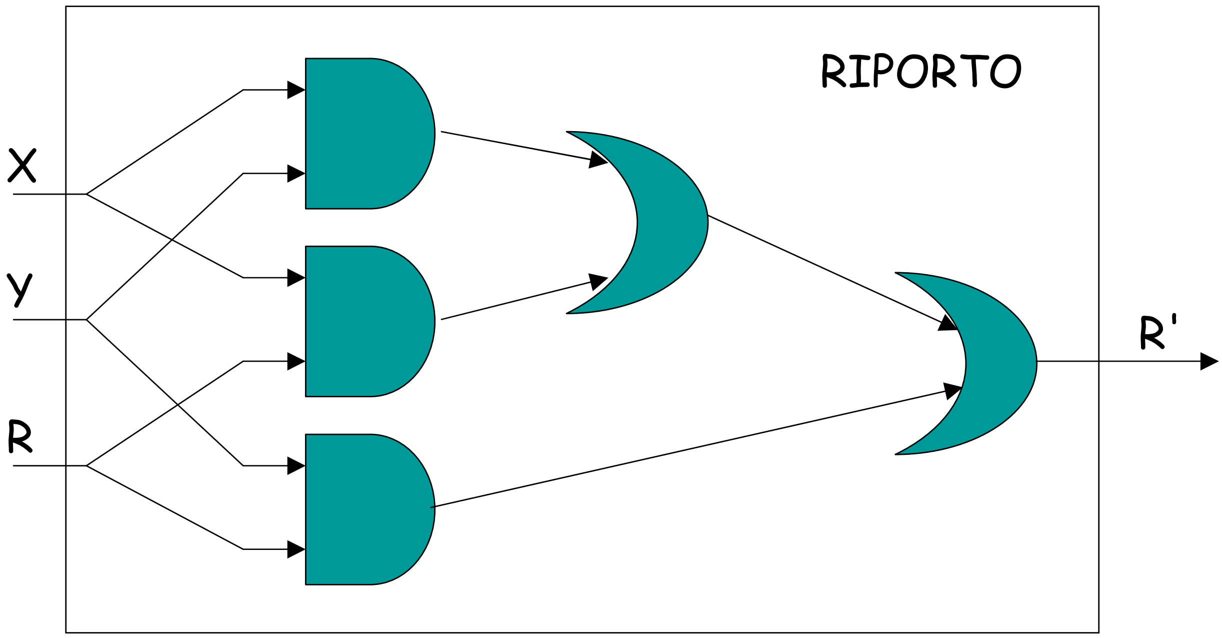
Supponiamo di avere i circuiti che calcolano somma e riporto



Possiamo allora combinare i circuiti SOMMA e RIPORTO per ottenere il seguente circuito 1-ADD



Il circuito RIPORTO puo` essere realizzato nel seguente modo



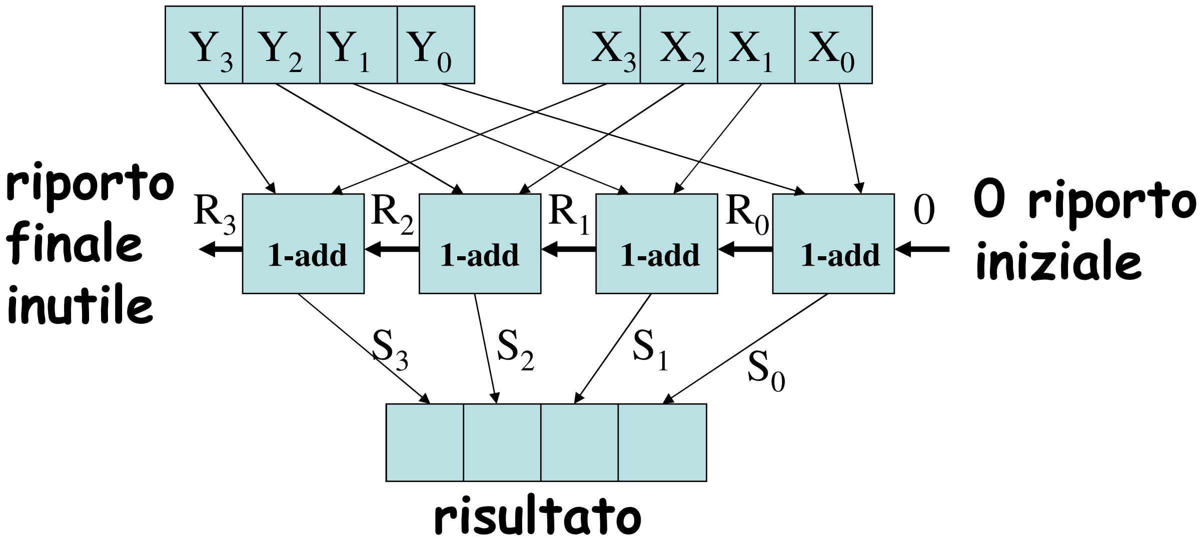
Basta infatti verificare la corrispondente tabella di verita: OTTIMO ESERCIZIO PER CASA!

Anche il circuito SOMMA essere realizzato (vedi dispensa).

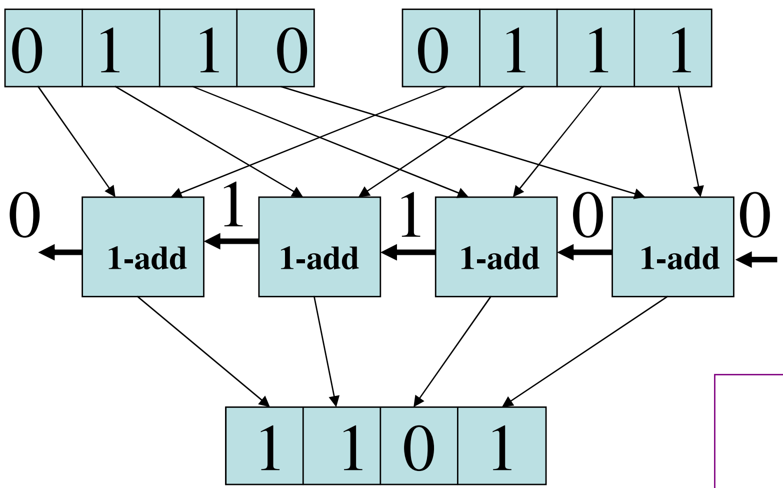
A questo punto componendo K circuiti 1-ADD e` possibile realizzare un circuito K-ADD che somma due numeri binari di K cifre.

Vediamo l'esempio della somma di due numeri binari di 4 cifre.

Somma di numeri di 4 bit



esempio




```
0111 +
0110 =
------
1101
```

Attenzione

Si e` trascurato il problema del cosiddetto **overflow**, ad esempio:

```
  0111 +
  1110 =
 ------
 10101
```


Ancora Esercizi

Definire la tabella di verità ed un circuito per questa operazione

OR esclusivo: XOR

A XOR B è vera quando esattamente uno tra A e B è vero

Suggerimento: possiamo osservare che la tabella di verita` e` "duale" alla tabella di verita` dell'equivalenza.

$$(A \Rightarrow (B \text{ and } C)) \overset{?}{\Rightarrow} (B \Rightarrow A)$$

Basta costruire la tabella di verità della formula e vedere se è una tautologia, ovvero se e' una formula sempre vera

A	B	C	$A \Rightarrow$ (B and C)	$B \Rightarrow A$	Valore
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	1	0	0
0	1	1	1	0	0
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	0	1	1
1	1	1	1	1	1

QUINDI:

$B \Rightarrow A$ non è conseguenza

$$(A \Rightarrow (B \text{ and } C)) \overset{?}{\Rightarrow} (A \Rightarrow B)$$

$$(A \Rightarrow B) \overset{?}{\Rightarrow} (\text{not } B \Rightarrow \text{not } A)$$

ESERCIZIO

Determinare la tavola di verità del seguente circuito. E` una tavola nota?