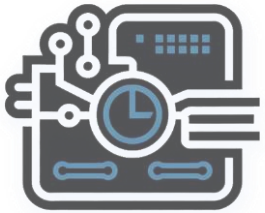


Progettazione Elettronica Digitale

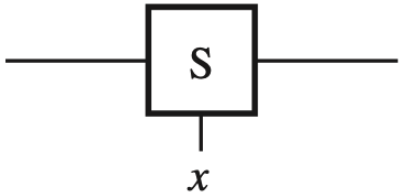
Introduzione ai circuiti logici

- ➔Segnali logici e natura elettrica
- ➔Funzioni logiche
- ➔Porte logiche
- ➔Circuiti logici
- ➔Richiami di algebra booleana

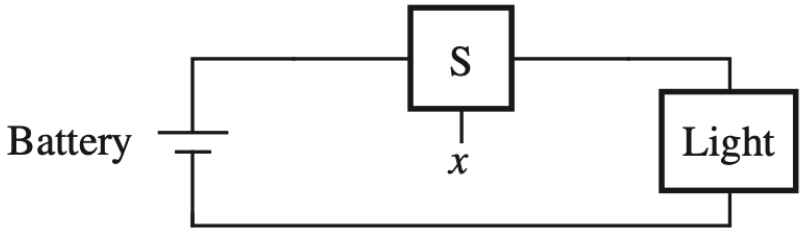
Zero (0) e Uno (1) come grandezze elettriche

- Consideriamo un interruttore (switch, S) con due stati: Aperto (0), Chiuso (1)
- Lo usiamo per accendere o spegnere una lampadina

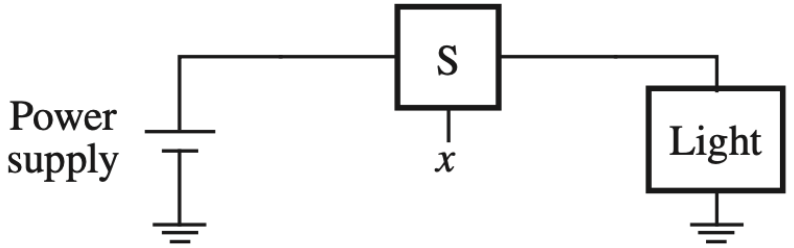
(a) Two states of a switch



(b) Symbol for a switch



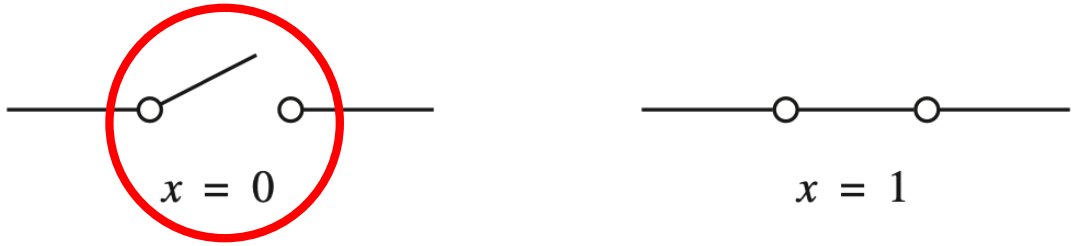
(a) Simple connection to a battery



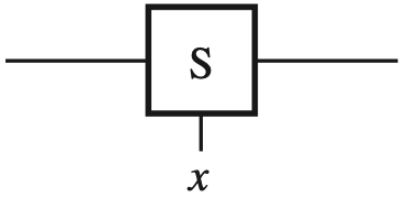
(b) Using a ground connection as the return path

Zero (0) e Uno (1) come grandezze elettriche

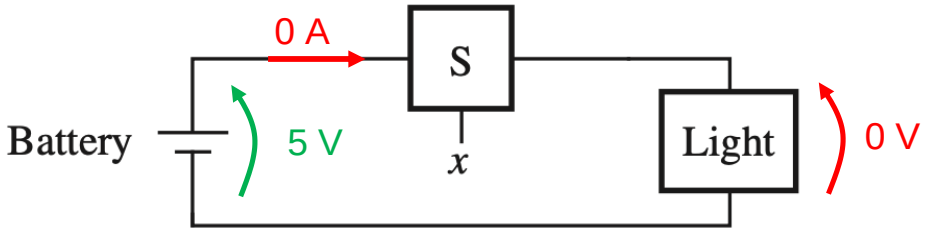
- Consideriamo un interruttore (switch, S) con due stati: Aperto (0), Chiuso (1)
- Lo usiamo per accendere o spegnere una lampadina



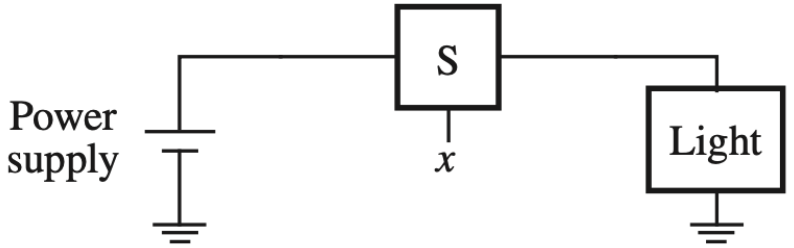
(a) Two states of a switch



(b) Symbol for a switch



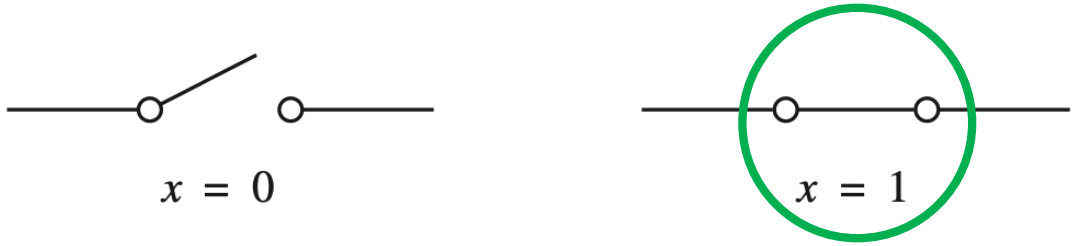
(a) Simple connection to a battery



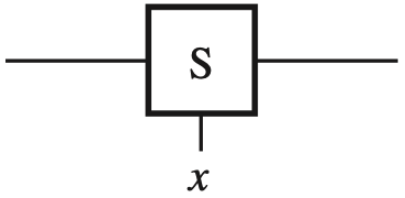
(b) Using a ground connection as the return path

Zero (0) e Uno (1) come grandezze elettriche

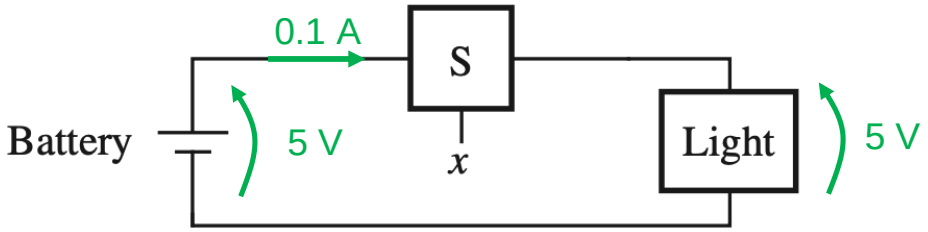
- Consideriamo un interruttore (switch, S) con due stati: Aperto (0), Chiuso (1)
- Lo usiamo per accendere o spegnere una lampadina



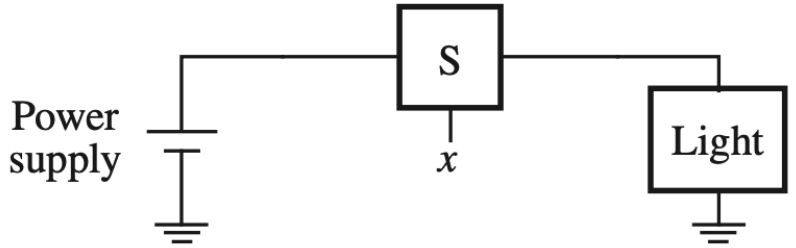
(a) Two states of a switch



(b) Symbol for a switch



(a) Simple connection to a battery



(b) Using a ground connection as the return path

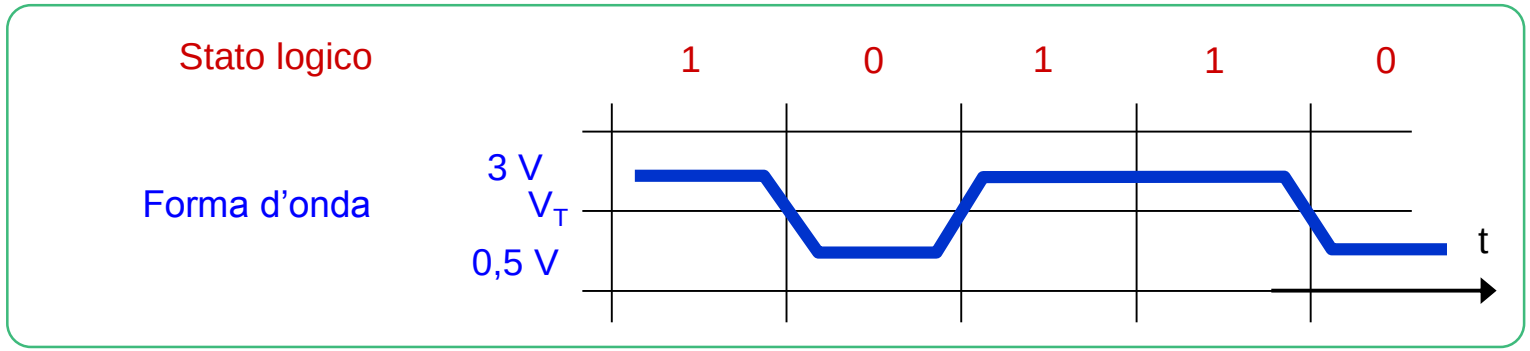
Corrispondenza tra livello di tensione e livello logico

Convenzione solitamente utilizzata

- **Tensione alta ⇔ 1 logico**
- **Tensione bassa ⇔ 0 logico**

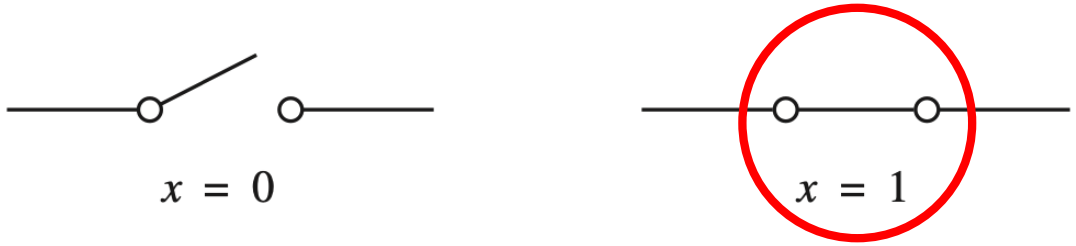
Ma quanto alta (o bassa) dev'essere una tensione per rappresentare 1 (o 0) logico?

- Concetto di **soglia (threshold) V_T**
  - **$V > V_T \Rightarrow$ 1 logico**
  - **$V < V_T \Rightarrow$ 0 logico**
- Concetto di **forma d'onda (waveform)**

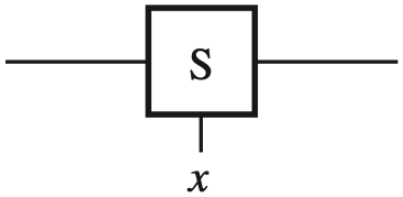


Inversione logica

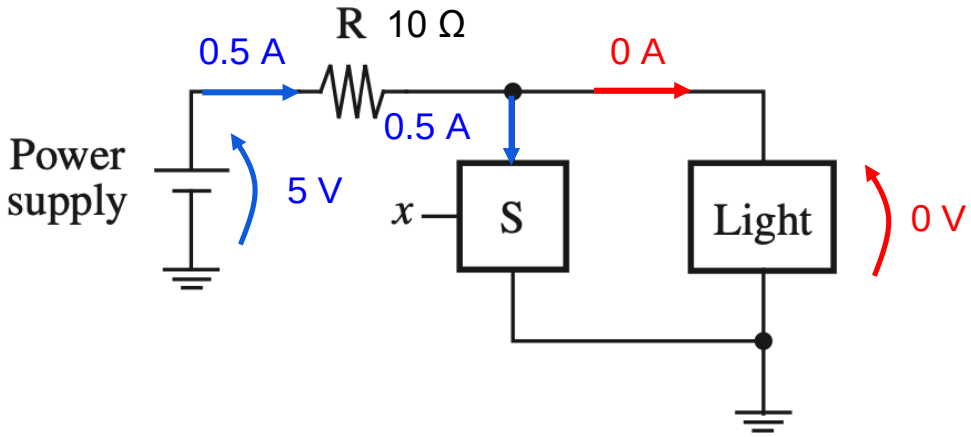
- Consideriamo un interruttore (switch, S) con due stati: Aperto (0), Chiuso (1)
- Ora accendiamo quando è aperto (0) e lo spegniamo quando è chiuso (1)



(a) Two states of a switch

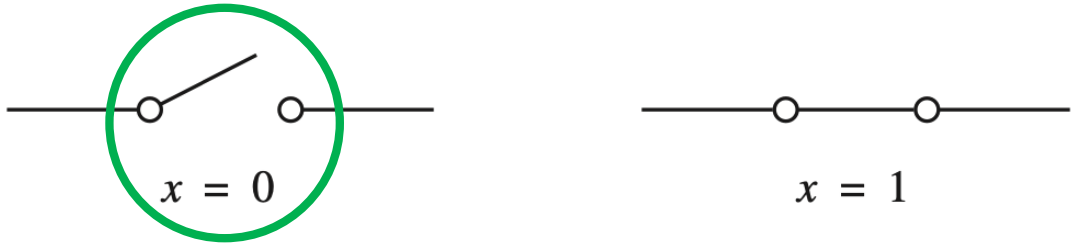


(b) Symbol for a switch

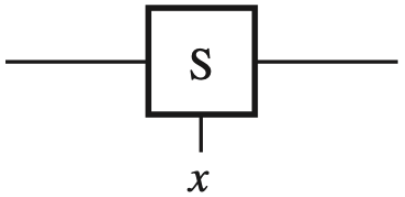


Inversione logica (NOT)

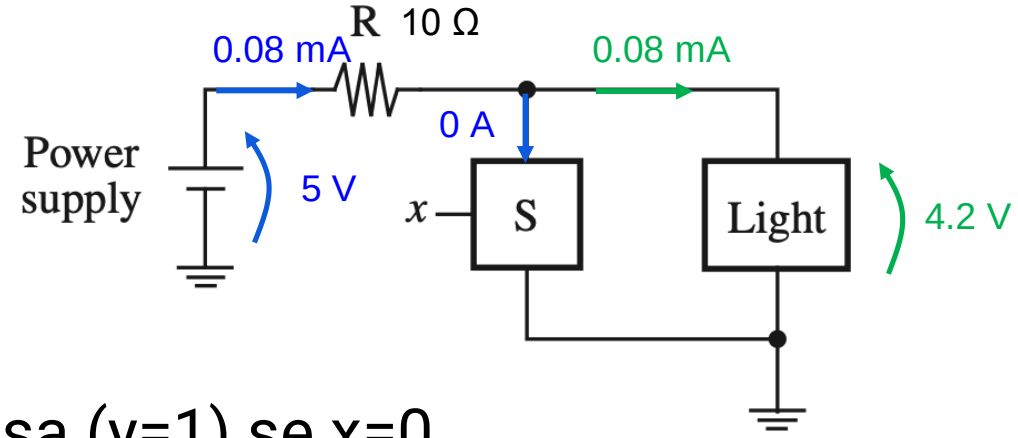
- Consideriamo un interruttore (switch, S) con due stati: Aperto (0), Chiuso (1)
- Ora accendiamo quando è aperto (0) e lo spegniamo quando è chiuso (1)



(a) Two states of a switch



(b) Symbol for a switch

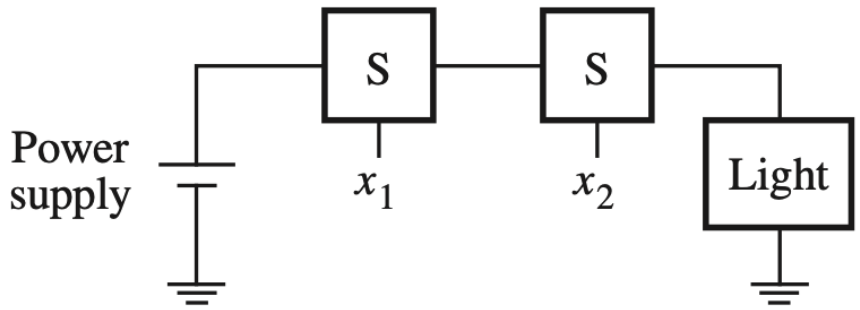


Luce accesa (y=1) se x=0

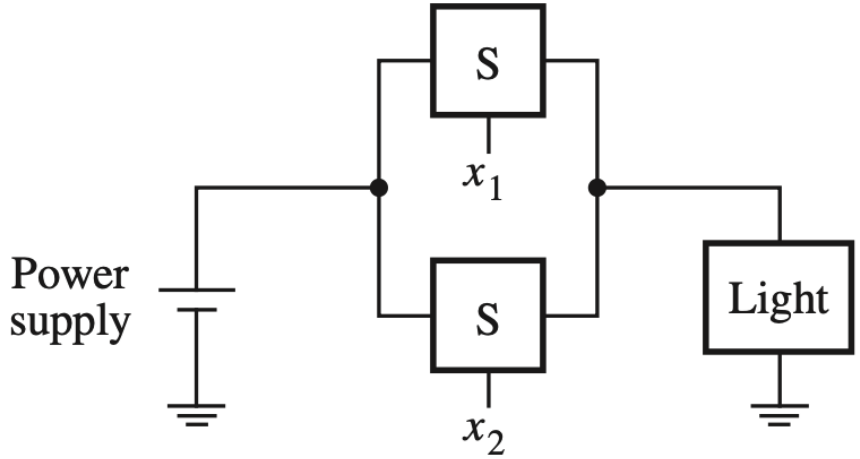
- Y = NOT x
  - Varie notazioni: NOT x = $\overline{x}$ = x' = !x = ~x = x*
- Cosa succede se R = 100 Ω?

Funzioni logiche AND e OR

- Due interruttori in serie o in parallelo
- Luce accesa (y=1) se
  - entrambi gli switch sono accesi (serie)
  - almeno uno è acceso (parallelo)
- $y = x_1$ AND x_2
  - Lo indichiamo come $y = x_1 \cdot x_2$
- $y = x_1$ OR x_2
  - Lo indichiamo come $y = x_1 + x_2$



(a) The logical AND function (series connection)



(b) The logical OR function (parallel connection)

Tavole di verità

- Tabelle che riportano il valore logico di una **funzione logica $y = f(x_1,x_2,\ldots,x_n)$** per tutte le possibili combinazioni logiche degli **ingressi** $x_1, x_2, \ldots, x_n$

NOT

x	$f = \overline{x}$
0	1
1	0

AND

x_1	x_2	$f = x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1

OR

x_1	x_2	$f = x_1 + x_2$
0	0	0
0	1	1
1	0	1
1	1	1

Tavole di verità

- Tabelle che riportano il valore logico di una **funzione logica $y = f(x_1,x_2,\ldots,x_n)$** per tutte le possibili combinazioni logiche degli **ingressi** $x_1, x_2, \ldots, x_n$

NOT

x	$f = \overline{x}$
0	1
1	0

AND

x_1	x_2	$f = x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1

OR

x_1	x_2	$f = x_1 + x_2$
0	0	0
0	1	1
1	0	1
1	1	1

NAND = NOT AND

x_1	x_2	$f = \overline{x_1 \cdot x_2}$
0	0	1
0	1	1
1	0	1
1	1	0

NOR = NOT OR

x_1	x_2	$f = \overline{x_1 + x_2}$
0	0	1
0	1	0
1	0	0
1	1	0

Tavole di verità

- Tabelle che riportano il valore logico di una **funzione logica $y = f(x_1,x_2,\ldots,x_n)$** per tutte le possibili combinazioni logiche degli **ingressi** x_1, x_2, …, x_n
- Tutte le 16 possibili funzioni di due ingressi (16 = 2^(2^2))

Ingressi

x_1	x_2	0	1	x_1	x_2	$\overline{x_1}$	$\overline{x_2}$	$x_1 \cdot x_2$	$x_1 + x_2$	$\overline{x_1 \cdot x_2}$	$\overline{x_1 + x_2}$	$x_1 \oplus x_2$	$\overline{x_1 \oplus x_2}$	$\overline{x_1} \cdot x_2$	$x_1 \cdot \overline{x_2}$	$\overline{\overline{x_1} \cdot x_2}$	$\overline{x_1 \cdot \overline{x_2}}$
0	0	0	1	0	0	1	1	0	0	1	1	0	1	0	0	1	1
0	1	0	1	0	1	1	0	0	1	1	0	1	0	1	0	0	1
1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	1	0
1	1	0	1	1	1	0	0	1	1	0	0	0	1	0	0	1	1

Costanti

XOR / XNOR

Tavole di verità

- Tabelle che riportano il valore logico di una **funzione logica $y = f(x_1,x_2,\ldots,x_n)$** per tutte le possibili combinazioni logiche degli **ingressi** x_1, x_2, …, x_n
- AND e OR di 3 ingressi
  - Provare a scrivere le tabelle di altre funzioni di 3 ingressi (es. NAND, NOR, ecc.)

AND

x_1	x_2	x_3	$x_1 \cdot x_2 \cdot x_3$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

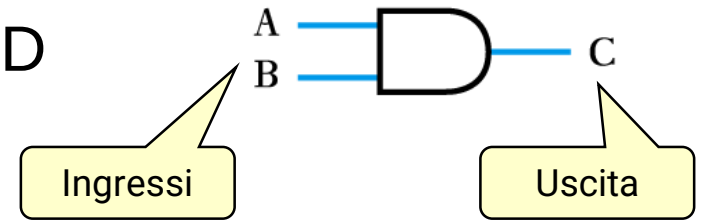
OR

x_1	x_2	x_3	$x_1 + x_2 + x_3$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Porte logiche

- Sono circuiti elettronici che contengono **transistori** opportunamente connessi per realizzare semplici funzioni logiche tra gli ingressi e l'uscita
  - I transistori si comportano da switch nelle porte logiche, lo vedremo più avanti

AND



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

$C = A \cdot B$

OR

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

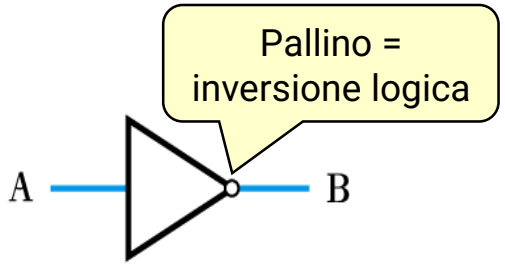
$C = A + B$

(a) Circuit symbol (b) Truth table (c) Boolean expression

Porte logiche

- Sono circuiti elettronici che contengono **transistori** opportunamente connessi per realizzare semplici funzioni logiche tra gli ingressi e l'uscita
  - I transistori si comportano da switch nelle porte logiche, lo vedremo più avanti

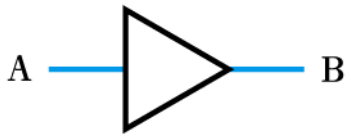
NOT
anche detto
INVERTER



A	B
0	1
1	0

$B = \overline{A}$

BUFFER
(non invertente)



(a) Circuit symbol

A	B
0	0
1	1

(b) Truth table

$B = A$

(c) Boolean expression

Porte logiche

- Sono circuiti elettronici che contengono **transistori** opportunamente connessi per realizzare semplici funzioni logiche tra gli ingressi e l'uscita
  - I transistori si comportano da switch nelle porte logiche, lo vedremo più avanti

NAND

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

$$C = \overline{A \cdot B}$$

NOR

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

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

(a) Circuit symbol (b) Truth table (c) Boolean expression

Porte logiche

- Sono circuiti elettronici che contengono **transistori** opportunamente connessi per realizzare semplici funzioni logiche tra gli ingressi e l'uscita
  - I transistori si comportano da switch nelle porte logiche, lo vedremo più avanti

XOR

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

$C = A \oplus B$

XNOR

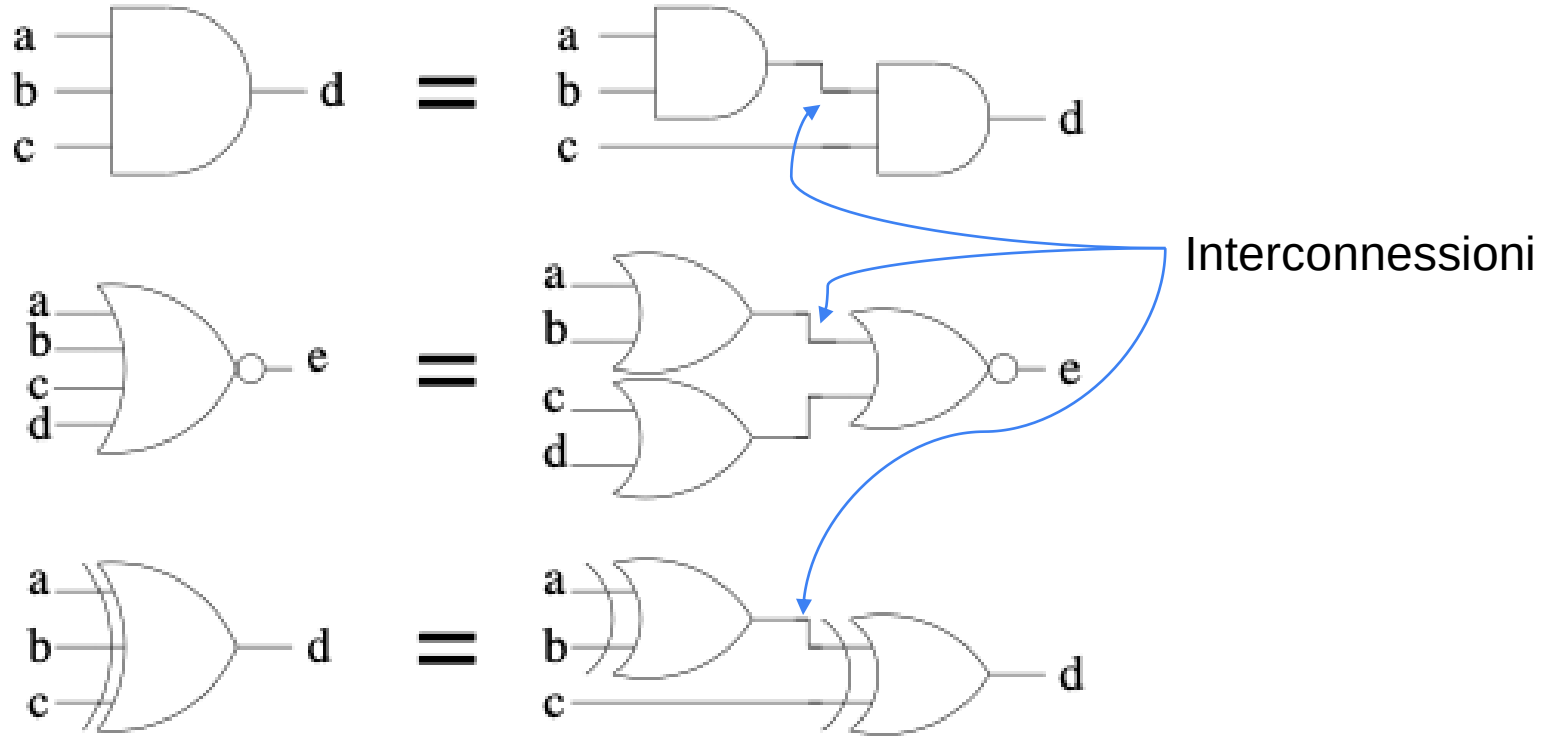
A	B	C
0	0	1
0	1	0
1	0	0
1	1	1

$C = \overline{A \oplus B}$

(a) Circuit symbol (b) Truth table (c) Boolean expression

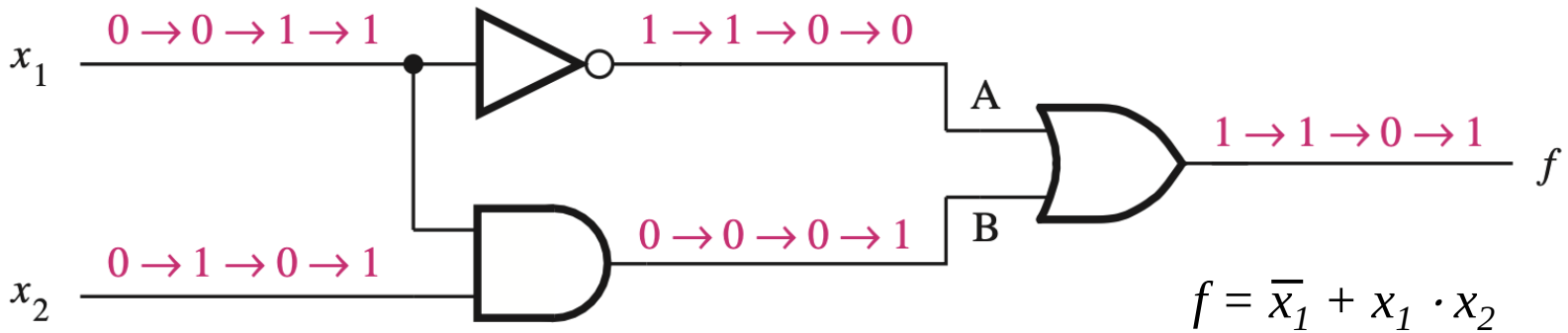
Dalle porte alle Reti Logiche (o Circuiti Logici)

- Esistono porte logiche anche a più di due ingressi, ad esempio tre o quattro
- Di solito non si usano porte con troppi ingressi (es. non più di 4/5 ingressi)
- Combinazioni di porte semplici **opportunamente collegate** possono essere usate per realizzare funzioni logiche più complesse
- Chiamiamo una rete di porte logiche interconnesse **Rete Logica** o **Circuito Logico**



Analisi di un circuito logico

- Dato un circuito logico, possiamo **analizzarne** il comportamento per determinare la funzione logica che realizza (o le funzioni, se ha più uscite)
  - Operazione inversa: **sintesi logica** (dal funzionamento al circuito), molto più complessa
- Per circuiti semplici basta considerare tutte le combinazioni degli ingressi
  - Per circuiti complessi non è possibile, es: 32 ingressi => 2^32 = 4.294.967.296 combinazioni!

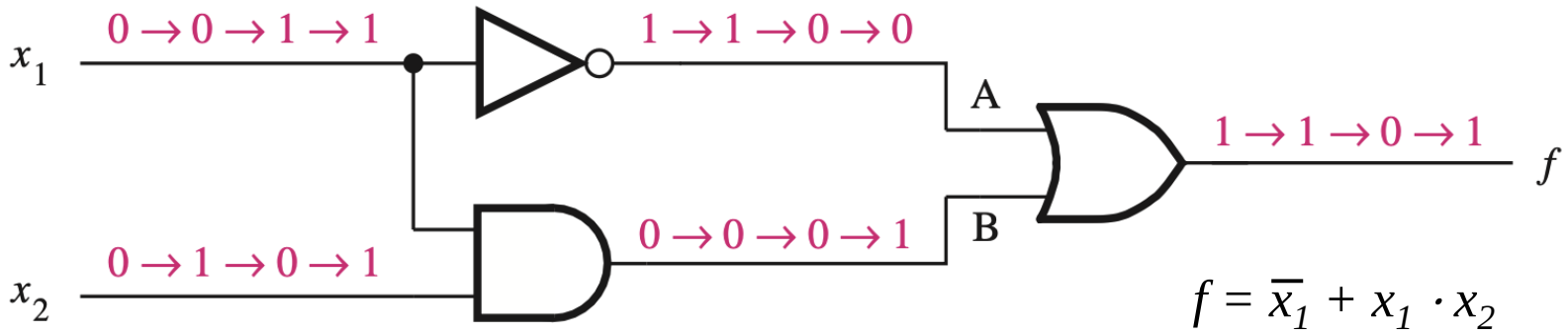


x_1	x_2	$f(x_1, x_2)$	A	B
0	0	1	1	0
0	1	1	1	0
1	0	0	0	0
1	1	1	0	1

- Possiamo inoltre usare l'algebra booleana per determinare la funzione

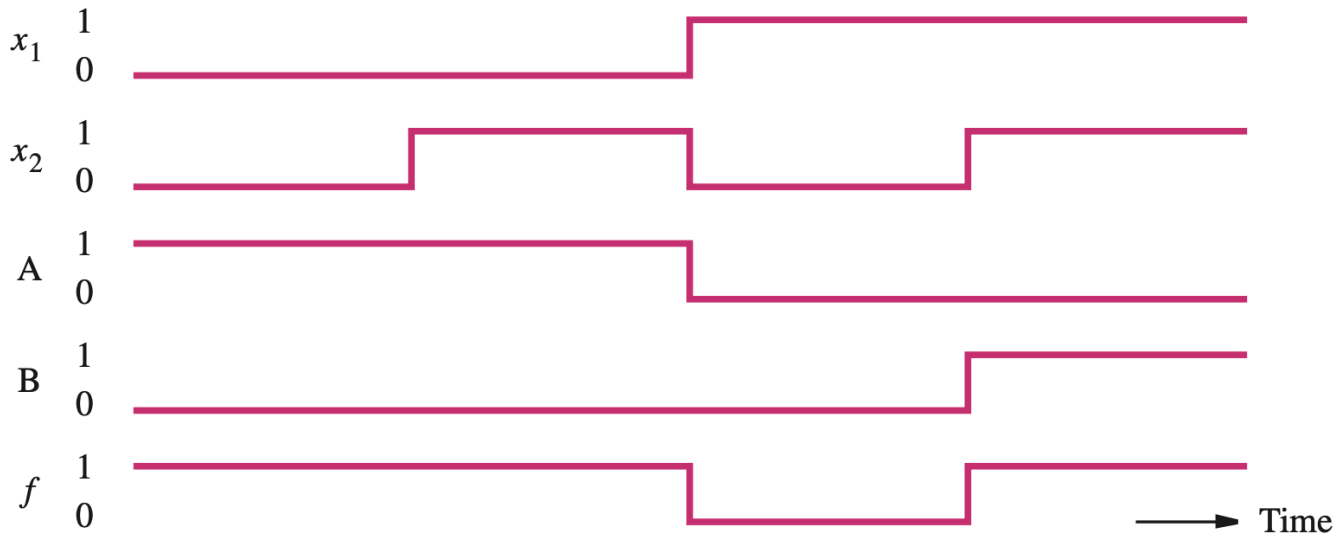
Analisi di un circuito logico

- In un circuito logico di solito gli ingressi variano nel tempo: **segnali logici**
- **Diagramma temporale** con forme d'onda di ingressi, nodi interni e uscita



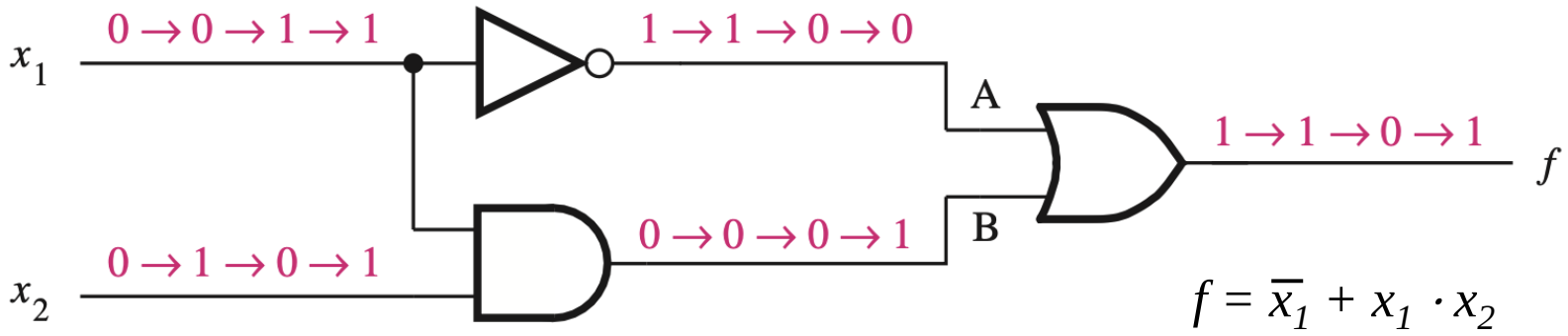
$$f = \overline{x_1} + x_1 \cdot x_2$$

x_1	x_2	$f(x_1, x_2)$	A	B
0	0	1	1	0
0	1	1	1	0
1	0	0	0	0
1	1	1	0	1

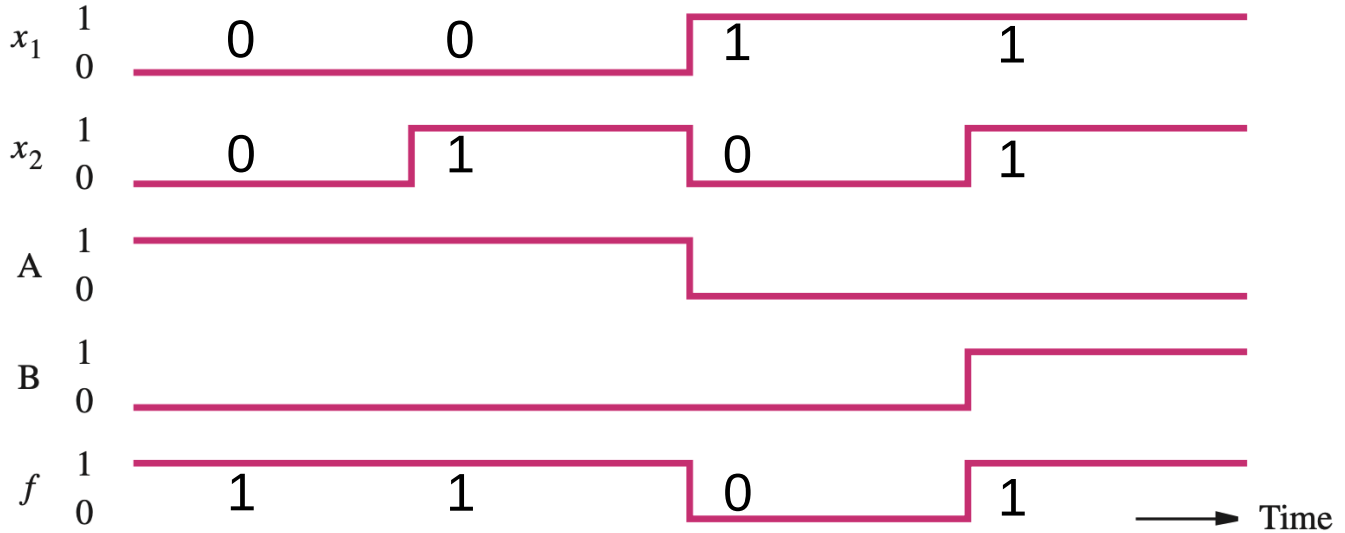


Analisi di un circuito logico

- In un circuito logico di solito gli ingressi variano nel tempo: **segnali logici**
- **Diagramma temporale** con forme d'onda di ingressi, nodi interni e uscita

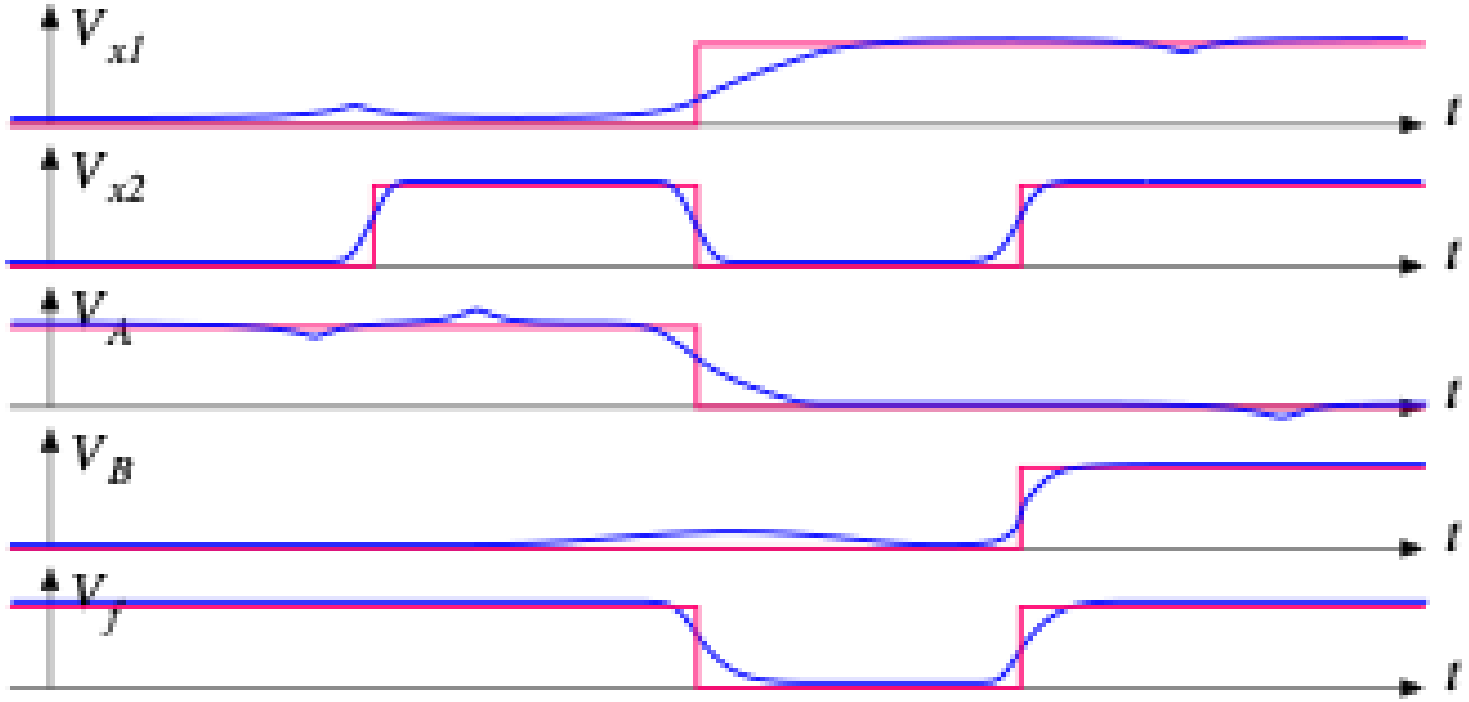


x_1	x_2	$f(x_1, x_2)$	A	B
0	0	1	1	0
0	1	1	1	0
1	0	0	0	0
1	1	1	0	1



Segnali logici vs segnali elettrici

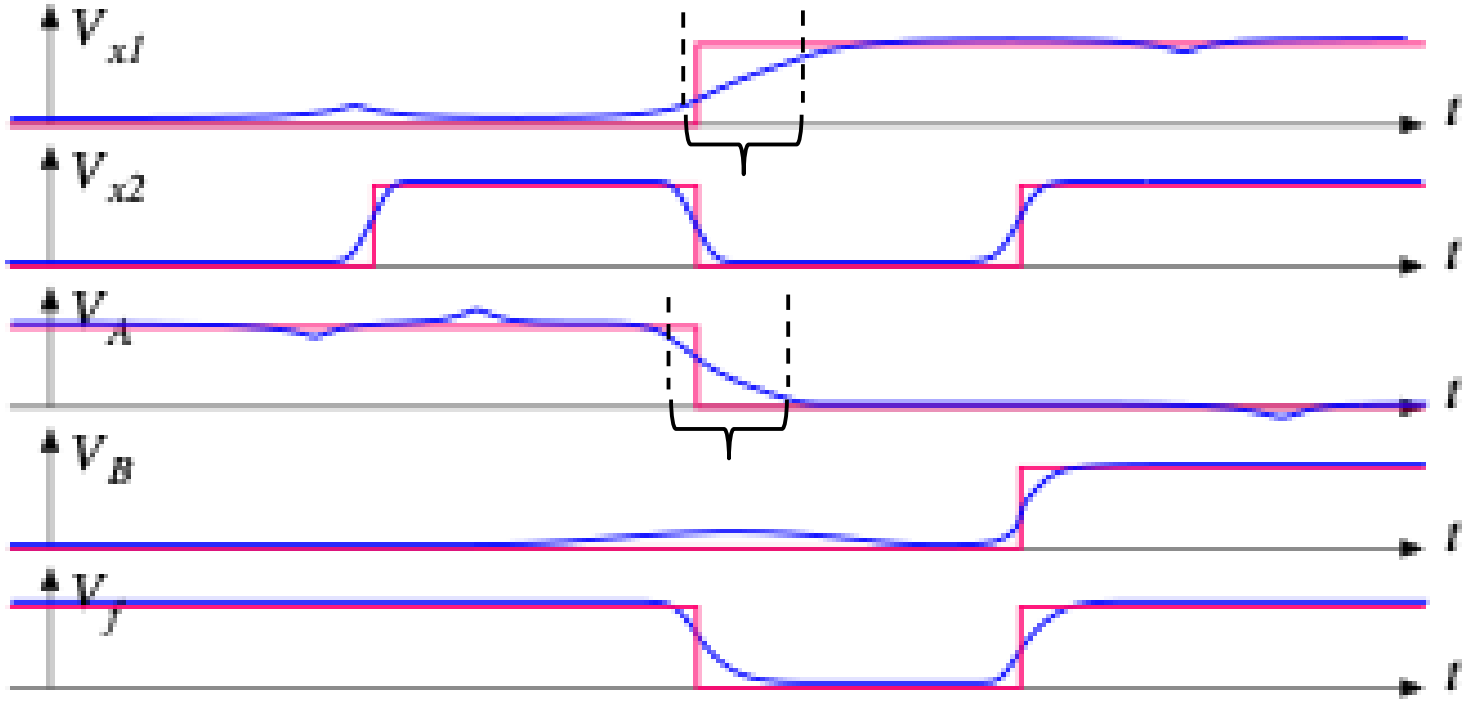
- I segnali logici sono un'astrazione di segnali elettrici caratterizzati da
  - **Livelli logici** => tensioni corrispondenti a 0 e 1
  - Tempi di **transizione** => **tempo di salita, tempo di discesa**
  - Eventuale presenza di **rumore**



- **Soglia V_T**
- **Fondamentale per passare dal segnale elettrico al segnale logico**

Segnali logici vs segnali elettrici

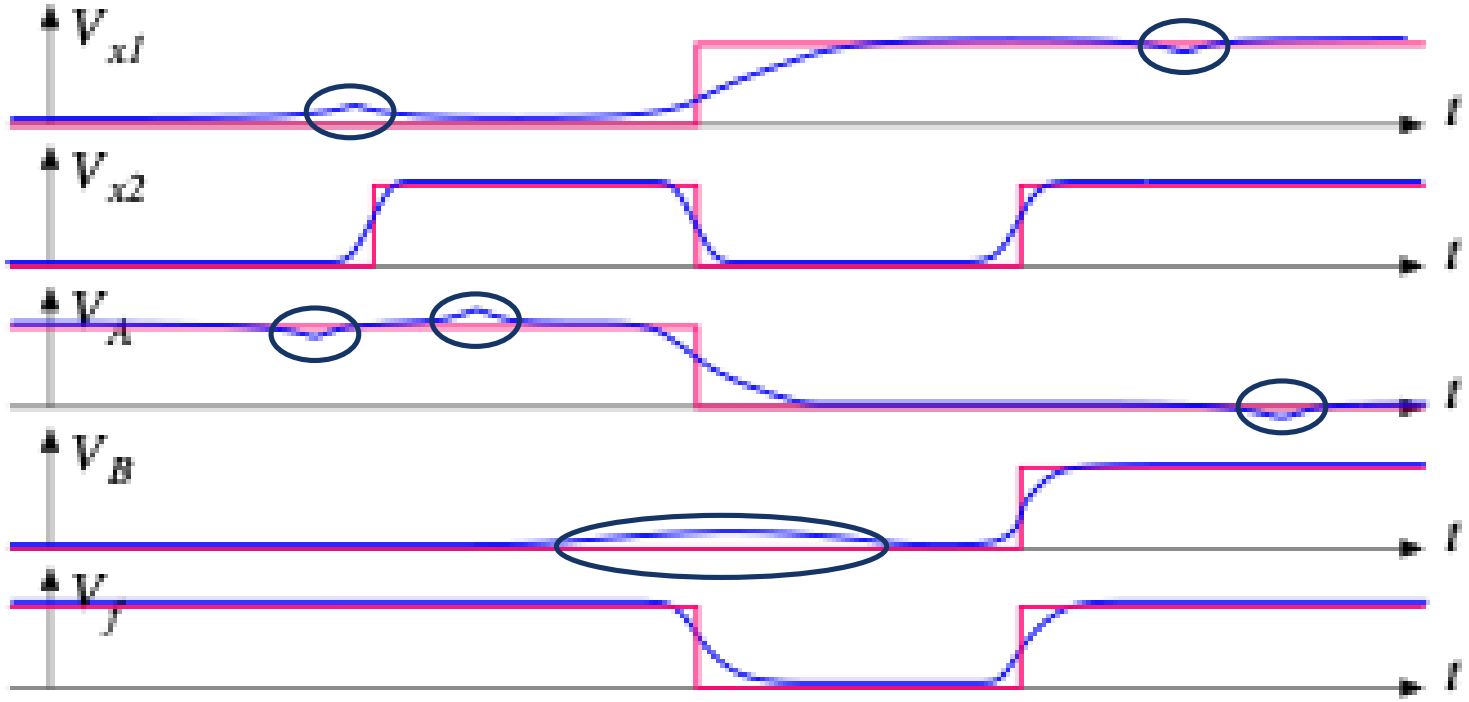
- I segnali logici sono un'astrazione di segnali elettrici caratterizzati da
  - **Livelli logici** => tensioni corrispondenti a 0 e 1
  - Tempi di **transizione** => **tempo di salita, tempo di discesa**
  - Eventuale presenza di **rumore**



- **Soglie 10% e 90% dell'escursione di tensione**
- **Necessarie per definire tempi di salita e di discesa**

Segnali logici vs segnali elettrici

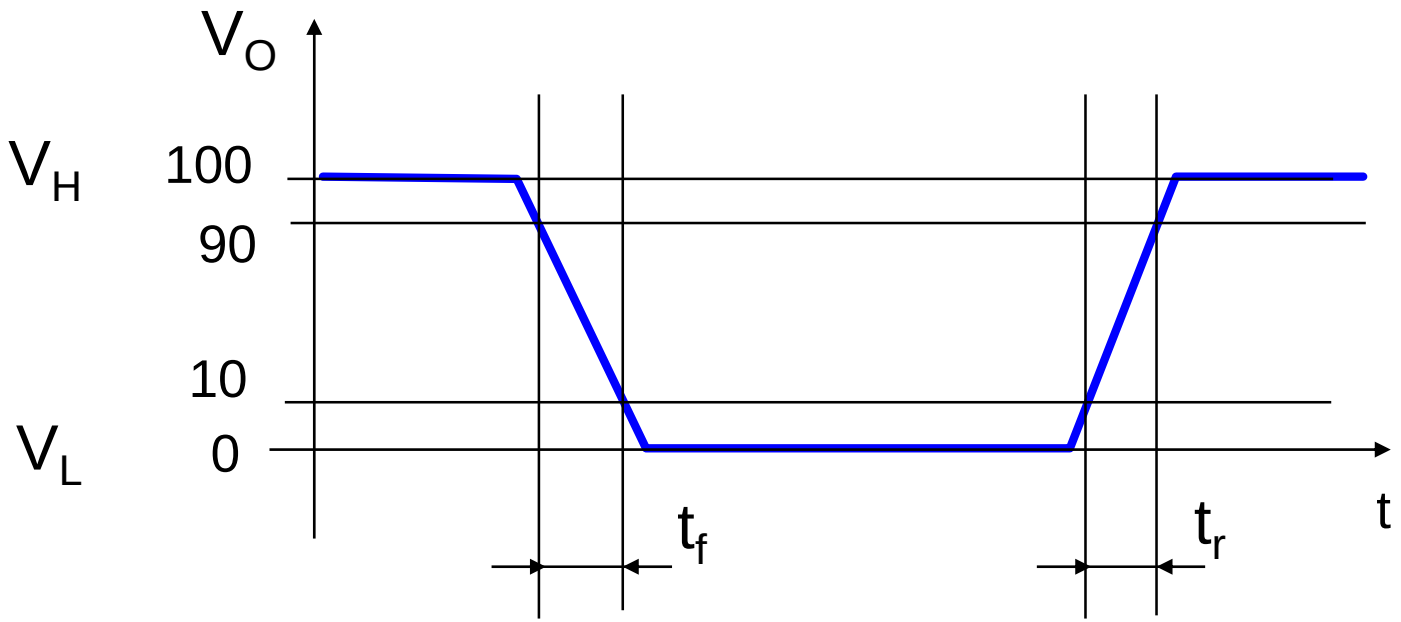
- I segnali logici sono un'astrazione di segnali elettrici caratterizzati da
  - **Livelli logici** => tensioni corrispondenti a 0 e 1
  - Tempi di **transizione** => **tempo di salita, tempo di discesa**
  - Eventuale presenza di **rumore**



- **Fondamentale che rumore non causi superamento della soglia V_T**
- **Altrimenti errore logico**

Tempi di transizione

- Tempi di transizione: salita (rise) e discesa (fall)
  - definiti tramite soglie poste al 10% e al 90% della variazione del segnale
  - Variazione tra **livello logico basso V_L** e **livello logico alto V_H** e viceversa

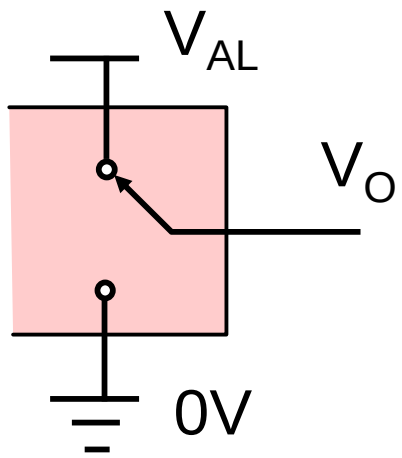


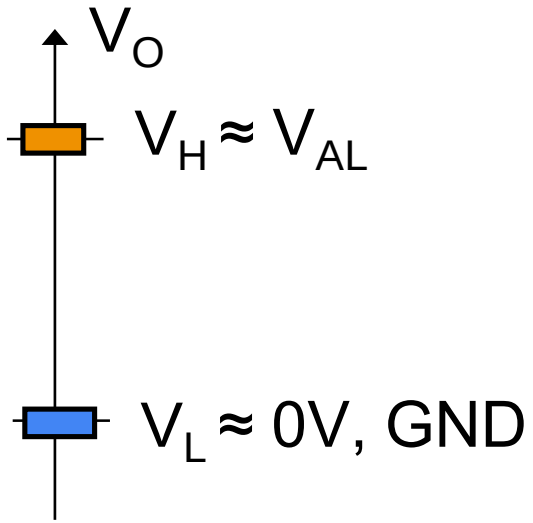
Livelli logici e tensioni di alimentazione

- Stati logici (0/1, L/H) → grandezze elettriche (V_L/V_H)
- V_H e V_L sono ricavate risp. dalla tensione di alimentazione V_{AL} e dallo 0V (GND)
- V_O tensione di uscita di una porta logica

stato 1, H :
$V_O = V_H$

stato 0, L :
$V_O = V_L$

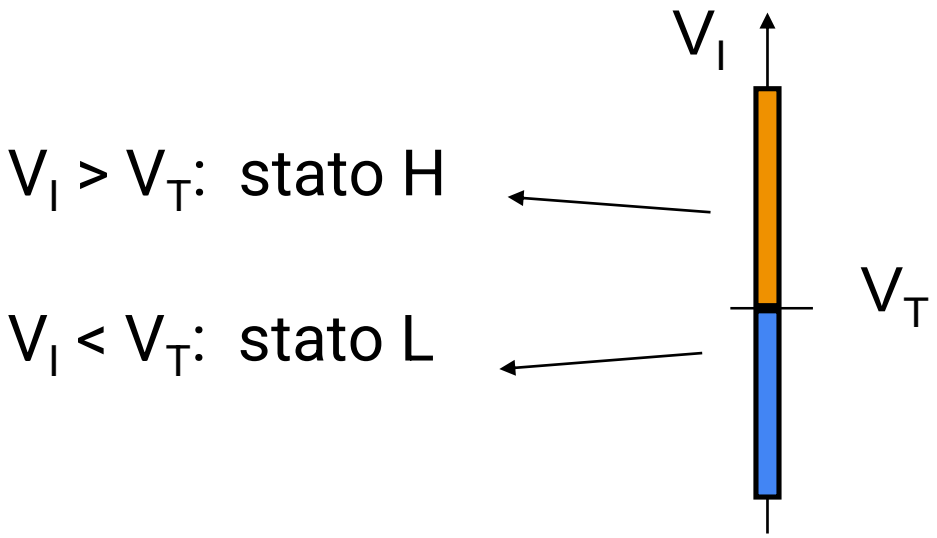


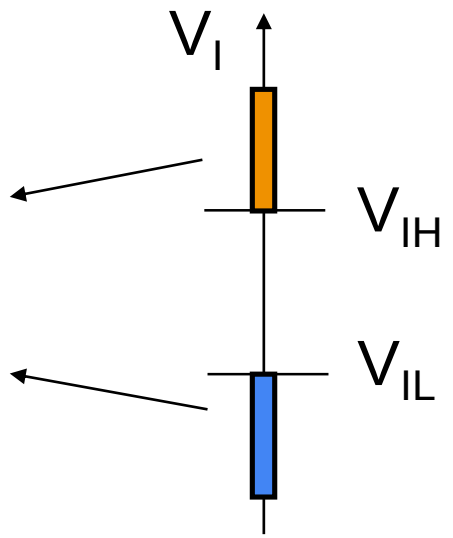


In prima approssimazione: $V_H = V_{AL}$ e $V_L = 0V$ (GND)

Ingressi di porte logiche e soglia V_T

- Una porta riconosce lo stato logico degli ingressi confrontando la tensione di ingresso V_I con la soglia V_T

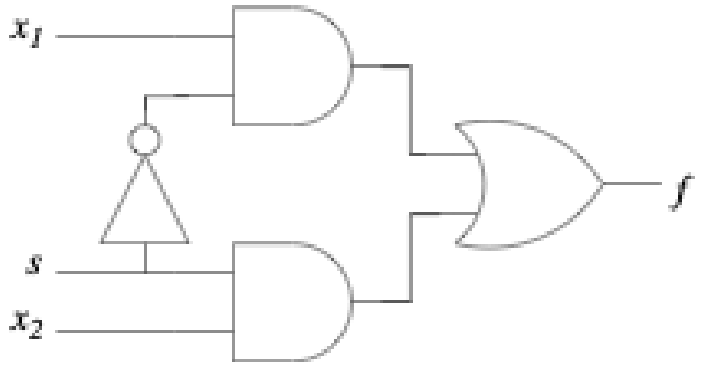




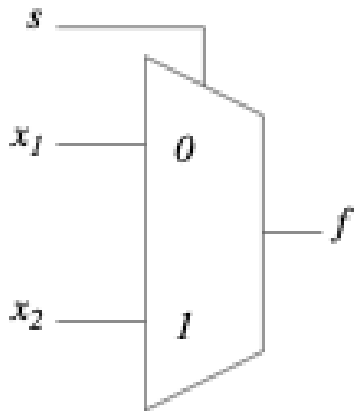
- La soglia V_T non è definita con precisione: definiamo un range
- Soglia massima V_{IH} e soglia minima V_{IL}
  - Tensione $V_{IL} < V_I < V_{IH}$ non definita: ammessa solo in transitorio
  - Al livello alto (basso) tolleriamo tanto più rumore quanto più V_I è maggiore (minore) di V_{IH} (V_{IL})

Esempio: multiplexer 2:1

- Un MUX 2:1 seleziona uno tra due ingressi e lo manda in uscita: $f = x_1 \cdot \overline{s} + x_2 \cdot s$



(a) Circuit



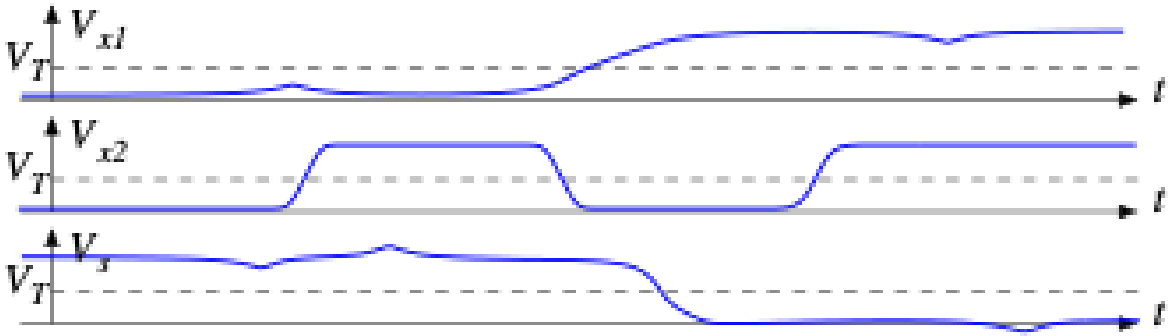
(b) Graphical symbol

s	x_1	x_2	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

(c) Truth table

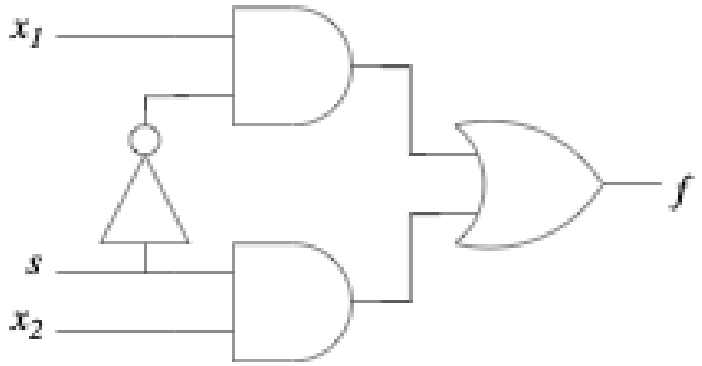
s	f
0	x_1
1	x_2

(d) Compact truth table

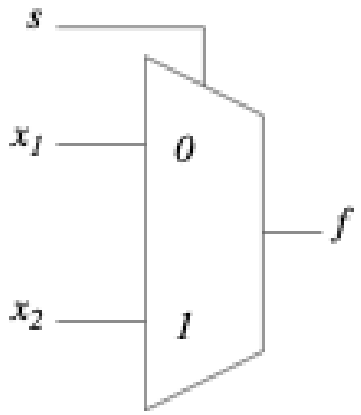


Esempio: multiplexer 2:1

- Un MUX 2:1 seleziona uno tra due ingressi e lo manda in uscita: $f = x_1 \cdot \overline{s} + x_2 \cdot s$



(a) Circuit



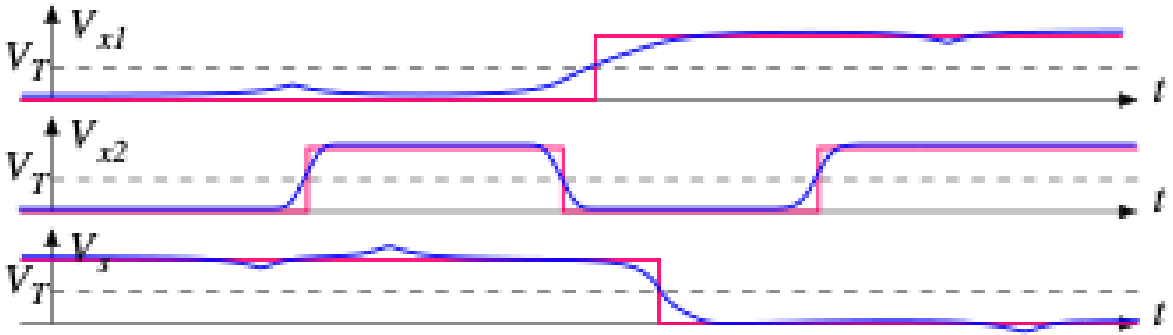
(b) Graphical symbol

s	x_1	x_2	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

(c) Truth table

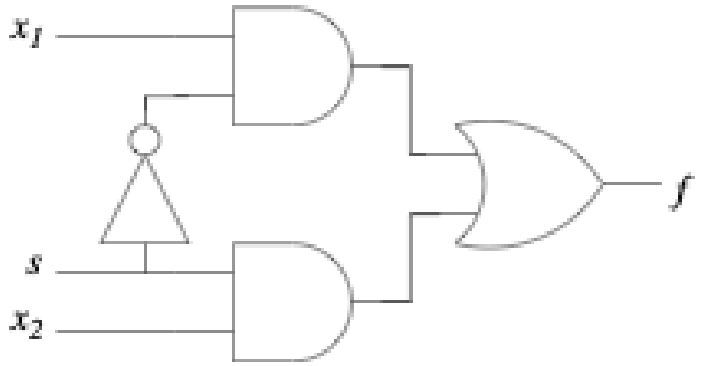
s	f
0	x_1
1	x_2

(d) Compact truth table

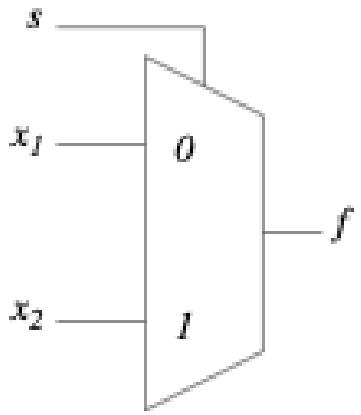


Esempio: multiplexer 2:1

- Un MUX 2:1 seleziona uno tra due ingressi e lo manda in uscita: $f = x_1 \cdot \overline{s} + x_2 \cdot s$



(a) Circuit



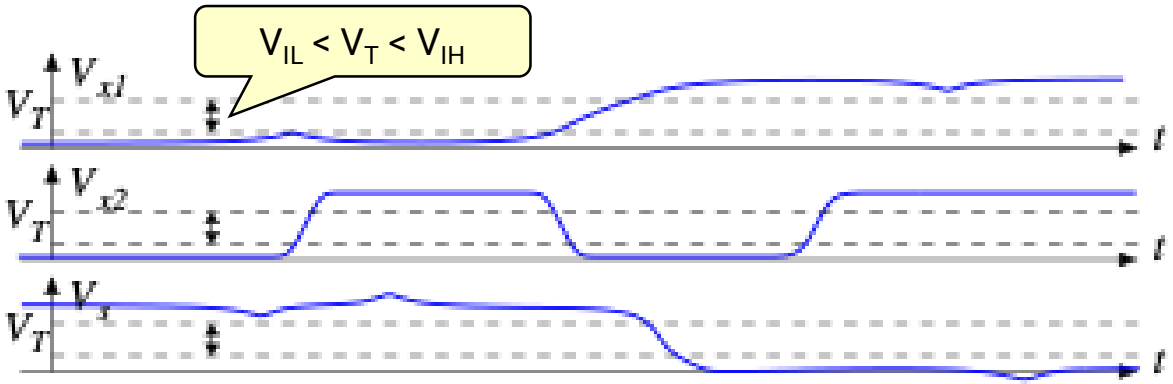
(b) Graphical symbol

s	x_1	x_2	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

(c) Truth table

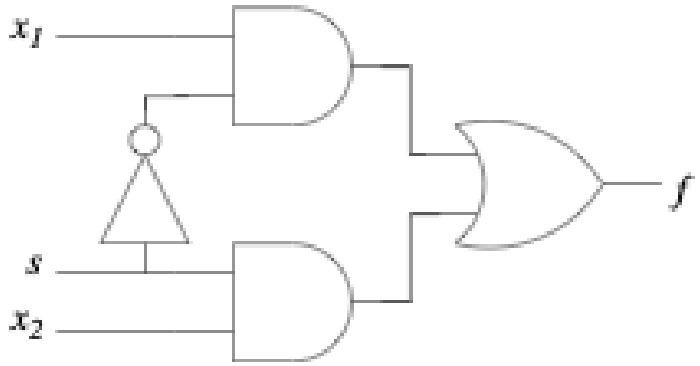
s	f
0	x_1
1	x_2

(d) Compact truth table

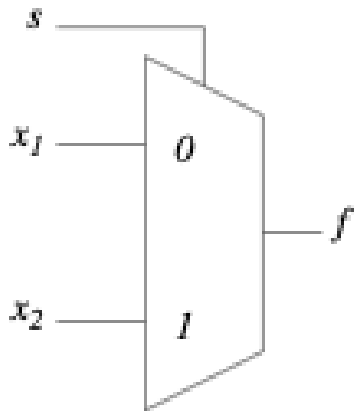


Esempio: multiplexer 2:1

- Un MUX 2:1 seleziona uno tra due ingressi e lo manda in uscita: $f = x_1 \cdot \overline{s} + x_2 \cdot s$



(a) Circuit



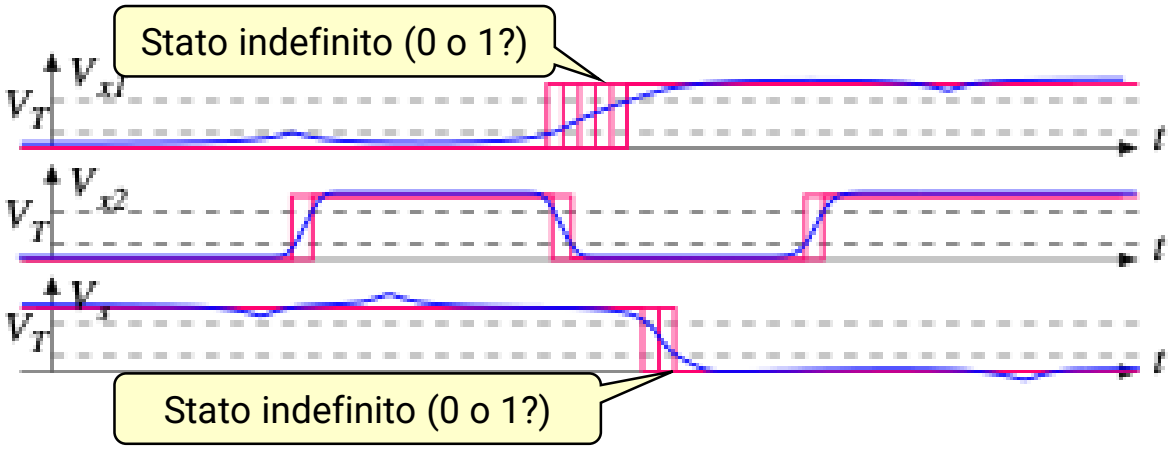
(b) Graphical symbol

s	x_1	x_2	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

(c) Truth table

s	f
0	x_1
1	x_2

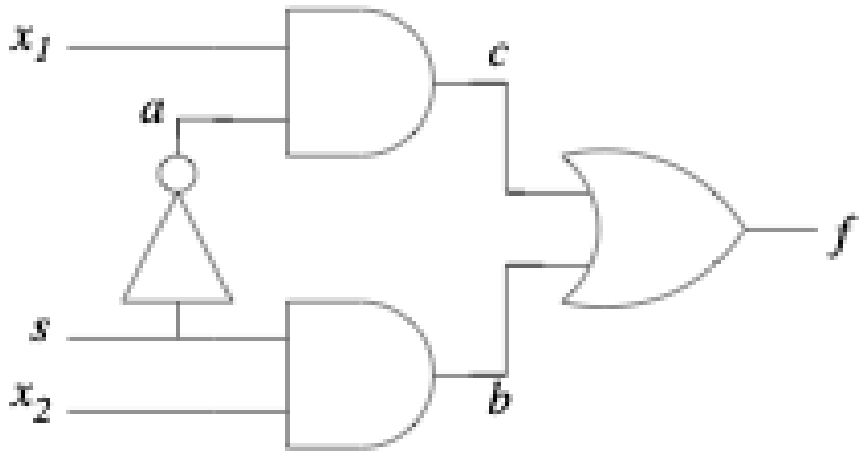
(d) Compact truth table



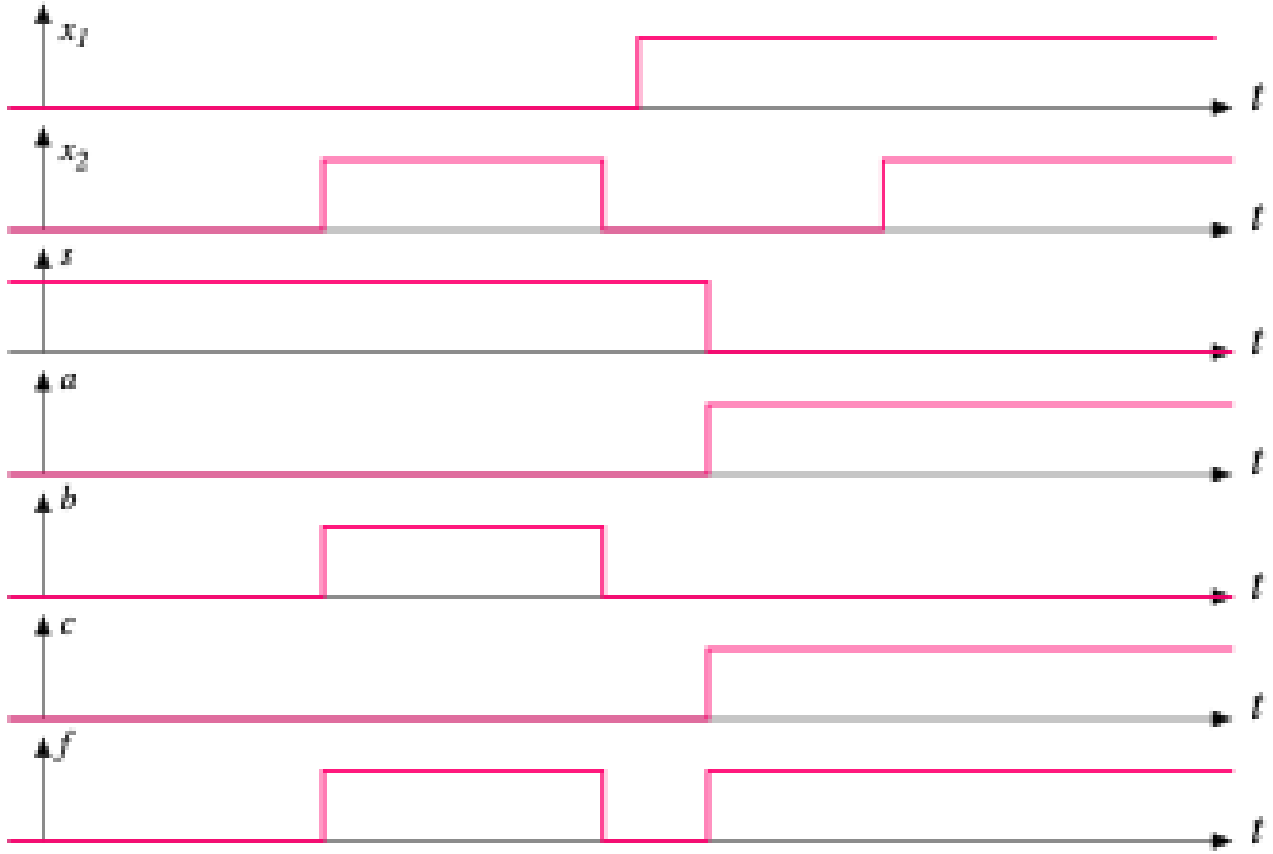
Tempi di salita e discesa devono essere brevi per ridurre l'incertezza

Esempio: multiplexer 2:1

- Un MUX 2:1 seleziona uno tra due ingressi e lo manda in uscita: $f = x_1 \cdot \overline{s} + x_2 \cdot s$

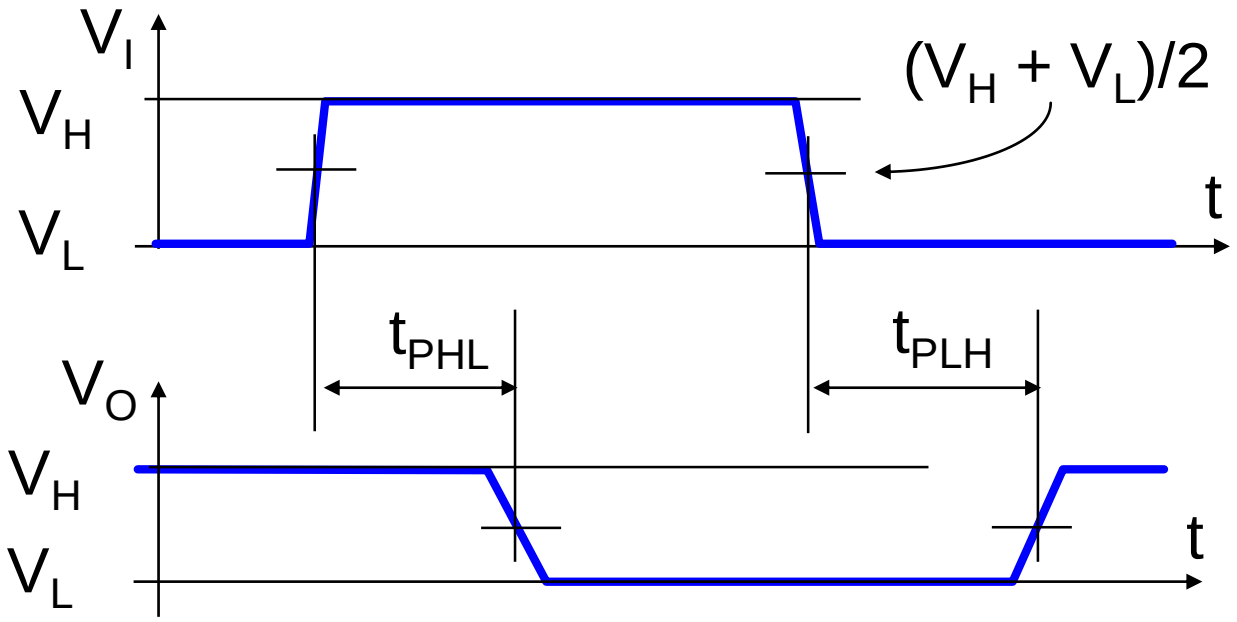


s	f
0	x_1
1	x_2



Ritardi di propagazione

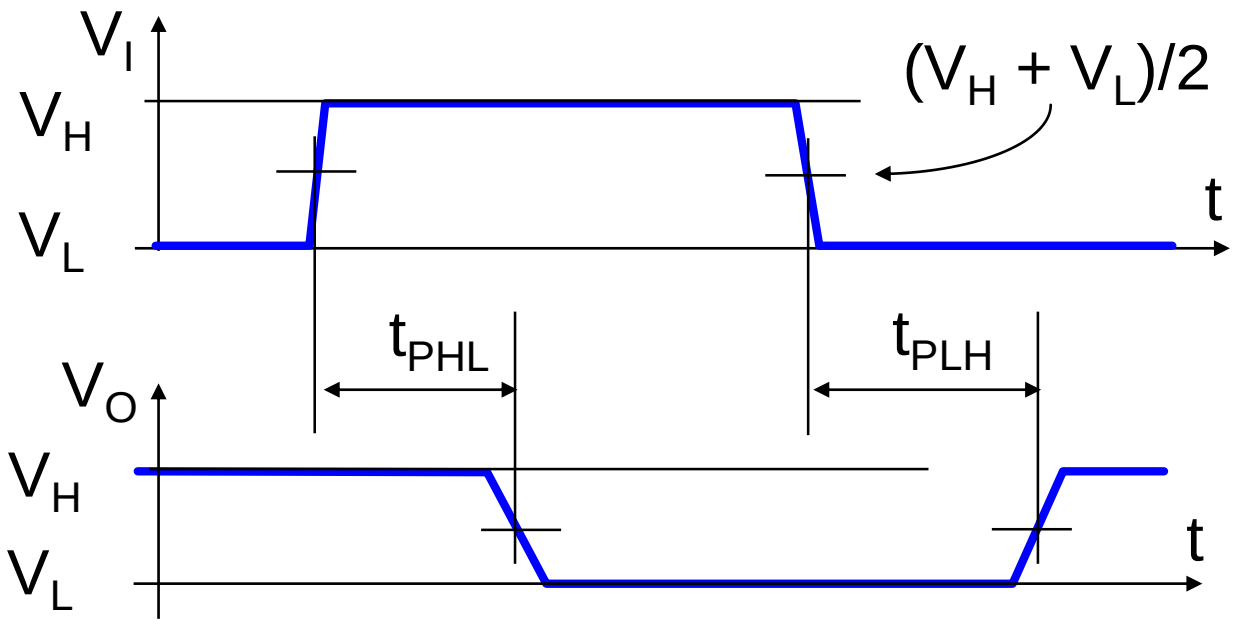
- Le uscite delle porte non cambiano istantaneamente
- I valori logici impiegano tempo a propagarsi tra ingressi e uscite
- Esempio: Inverter



- t_{PHL}: ritardo di propagazione con ingresso che sale (high, H) e uscita che scende (low, L)
- t_{PLH}: ritardo di propagazione con ingresso che scende (L) e uscita che sale (H)

Ritardi di propagazione

- Le uscite delle porte non cambiano istantaneamente
- I valori logici impiegano tempo a propagarsi tra ingressi e uscite
- Esempio: Inverter

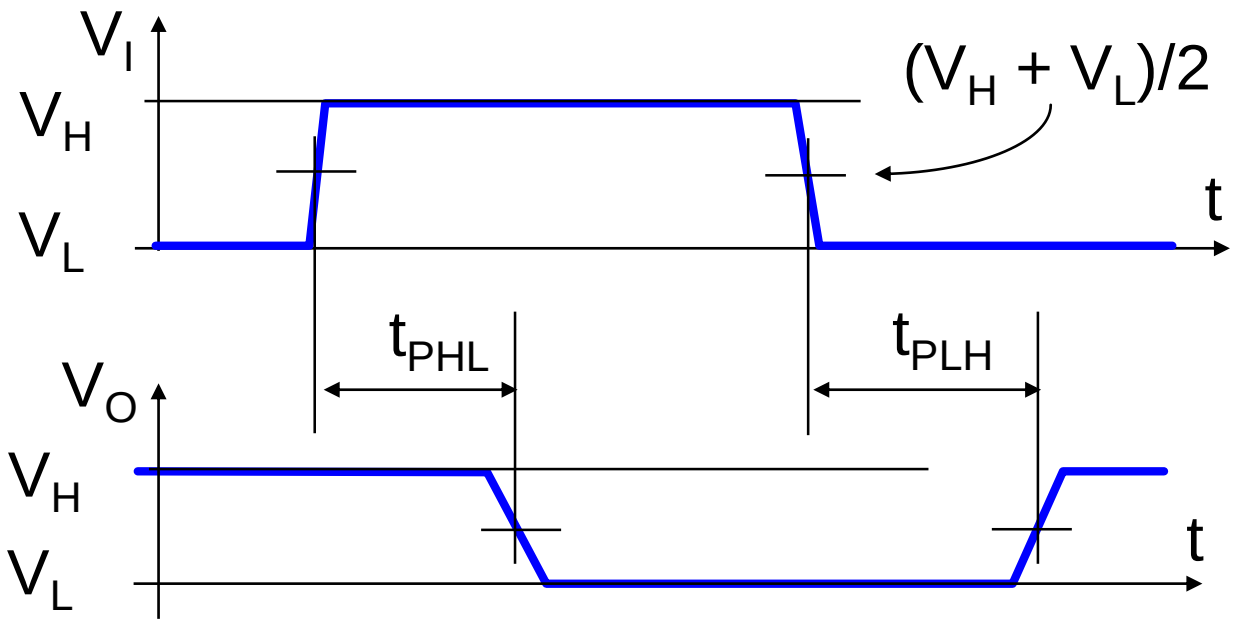


Ritardi misurati da $V_I = (V_H + V_L)/2$
a $V_O = (V_H + V_L)/2$

- t_{PHL}: ritardo di propagazione con ingresso che sale (high, H) e uscita che scende (low, L)
- t_{PLH}: ritardo di propagazione con ingresso che scende (L) e uscita che sale (H)

Ritardi di propagazione

- Le uscite delle porte non cambiano istantaneamente
- I valori logici impiegano tempo a propagarsi tra ingressi e uscite
- Esempio: Inverter



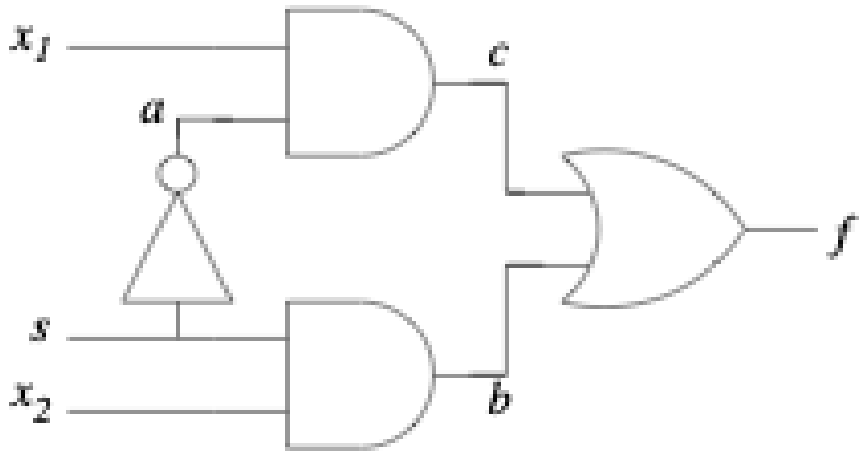
Ritardi misurati da $V_I = (V_H + V_L)/2$ a $V_O = (V_H + V_L)/2$

- t_{PHL}: ritardo di propagazione con ingresso che sale (high, H) e uscita che scende (low, L)
- t_{PLH}: ritardo di propagazione con ingresso che scende (L) e uscita che sale (H)

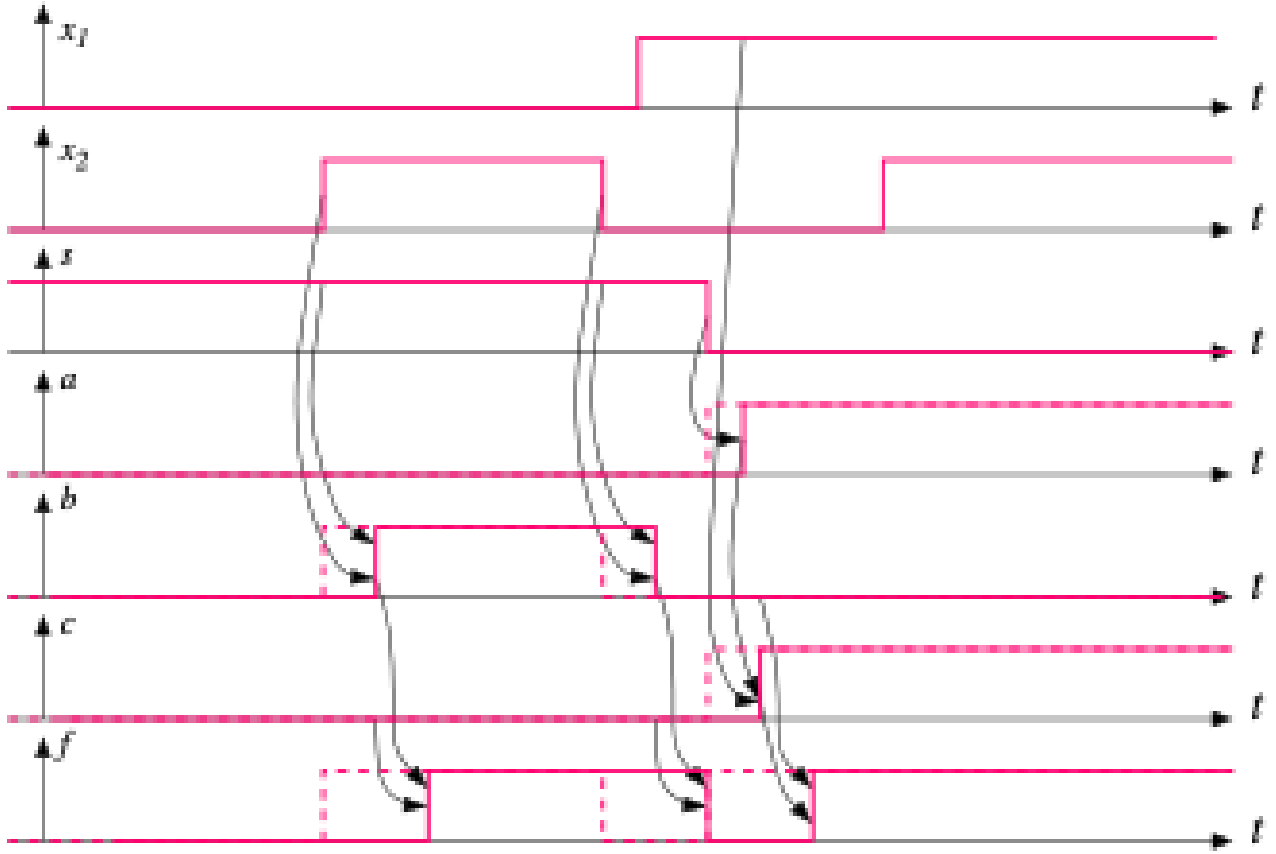
In base al tipo di porta possono anche essere presenti t_{PLL} e t_{PHH}

Esempio: multiplexer 2:1 con ritardi

- Un MUX 2:1 seleziona uno tra due ingressi e lo manda in uscita: $f = x_1 \cdot \overline{s} + x_2 \cdot s$

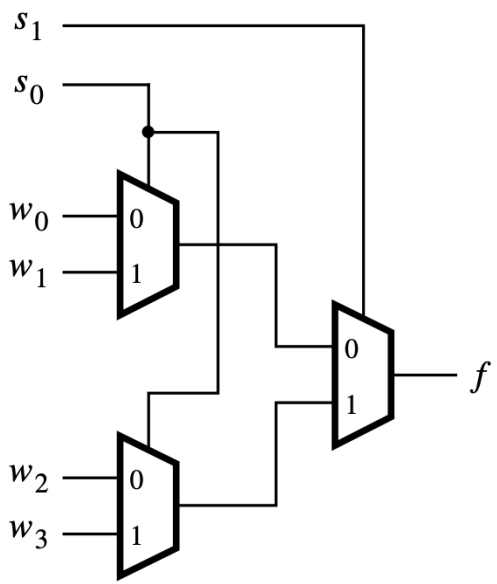


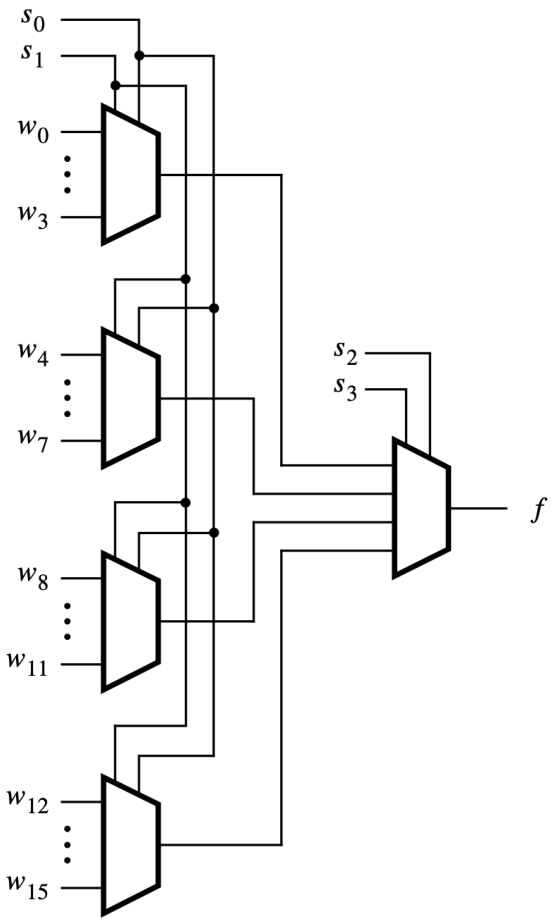
s	f
0	x_1
1	x_2

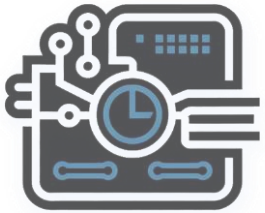


Circuiti logici combinatori

- Gli esempi visti finora sono circuiti **combinatori**
- **In un circuito combinatorio, ad ogni istante t le uscite dipendono solo dal valore logico degli ingressi in quell'istante**
- Idealmente, trascurando i ritardi:
  - $y(t) = f(x_1(t), x_2(t), \ldots, x_n(t))$
- Se mettiamo insieme più circuiti combinatori creando un grafo **aciclico**, otteniamo sempre un circuito combinatorio
- Esempio: multiplexer 4:1 e 16:1 "gerarchici"





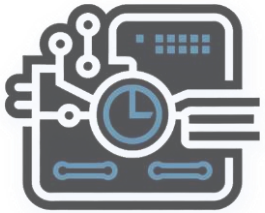


Richiami di algebra di Boole

- Per l'analisi e la sintesi di circuiti logici occorre conoscere bene l'algebra booleana

Assiomi:

- $0 \cdot 0 = 0$
- $1+1 = 1$
- $1 \cdot 1 = 1$
- $0+0 = 0$
- $0 \cdot 1 = 1 \cdot 0 = 0$
- $1+0 = 0+1 = 1$
- $x = 0 => \overline{x} = 1$
- $x = 1 => \overline{x} = 0$



Richiami di algebra di Boole

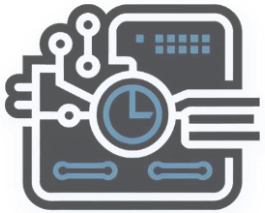
- Per l'analisi e la sintesi di circuiti logici occorre conoscere bene l'algebra booleana

Assiomi:

- $0 \cdot 0 = 0$
- $1+1 = 1$
- $1 \cdot 1 = 1$
- $0+0 = 0$
- $0 \cdot 1 = 1 \cdot 0 = 0$
- $1+0 = 0+1 = 1$
- $x = 0 => \overline{x} = 1$
- $x = 1 => \overline{x} = 0$

Teoremi (una variabile)

- $x \cdot 0 = 0$
- $x+1 = 1$
- $x \cdot 1 = x$
- $x+0 = x$
- $x \cdot x = x$
- $x+x = x$
- $x \cdot \overline{x} = 0$
- $x+\overline{x} = 1$
- $x = x$



Richiami di algebra di Boole

- Per l'analisi e la sintesi di circuiti logici occorre conoscere bene l'algebra booleana

Assiomi:

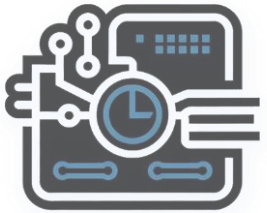
- $0 \cdot 0 = 0$
- $1+1 = 1$
- $1 \cdot 1 = 1$
- $0+0 = 0$
- $0 \cdot 1 = 1 \cdot 0 = 0$
- $1+0 = 0+1 = 1$
- $x = 0 \Rightarrow \overline{x} = 1$
- $x = 1 \Rightarrow \overline{x} = 0$

Teoremi (una variabile)

- $x \cdot 0 = 0$
- $x+1 = 1$
- $x \cdot 1 = x$
- $x+0 = x$
- $x \cdot x = x$
- $x+x = x$
- $x \cdot \overline{x} = 0$
- $x+\overline{x} = 1$
- $x = x$

Proprietà (due variabili)

$x \cdot y = y \cdot x$	commutativa
$x+y = y+x$	//
$x \cdot (y \cdot z) = (x \cdot y) \cdot z$	associativa
$x+(y+z) = (x+y)+z$	//
$x \cdot (y+z) = x \cdot y + x \cdot z$	distributiva
$x+y \cdot z = (x+y) \cdot (x+z)$	fattorizzazione
$x + x \cdot y = x$	assorbimento (1)
$x \cdot (x+y) = x$	//
$x + \overline{x} \cdot y = x+y$	assorbimento (2)
$x \cdot (\overline{x}+y) = x \cdot y$	//
$x \cdot y + x \cdot \overline{y} = x$	adiacenza
$(x+y) \cdot (x+\overline{y}) = x$	//



Richiami di algebra di Boole

- Per l'analisi e la sintesi di circuiti logici occorre conoscere bene l'algebra booleana

Assiomi:

- $0 \cdot 0 = 0$
- $1+1 = 1$
- $1 \cdot 1 = 1$
- $0+0 = 0$
- $0 \cdot 1 = 1 \cdot 0 = 0$
- $1+0 = 0+1 = 1$
- $x = 0 \Rightarrow \overline{x} = 1$
- $x = 1 \Rightarrow \overline{x} = 0$

Teoremi (una variabile)

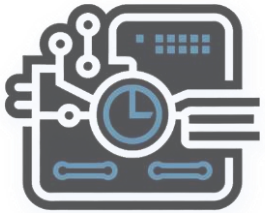
- $x \cdot 0 = 0$
- $x+1 = 1$
- $x \cdot 1 = x$
- $x+0 = x$
- $x \cdot x = x$
- $x+x = x$
- $x \cdot \overline{x} = 0$
- $x+\overline{x} = 1$
- $x = x$

Proprietà (due variabili)

$x \cdot y = y \cdot x$	commutativa
$x+y = y+x$	//
$x \cdot (y \cdot z) = (x \cdot y) \cdot z$	*associativa*
$x+(y+z) = (x+y)+z$	//
$x \cdot (y+z) = x \cdot y + x \cdot z$	*distributiva*
$x+y \cdot z = (x+y) \cdot (x+z)$	*fattorizzazione*
$x + x \cdot y = x$	*assorbimento (1)*
$x \cdot (x+y) = x$	//
$x + \overline{x} \cdot y = x+y$	*assorbimento (2)*
$x \cdot (\overline{x}+y) = x \cdot y$	//
$x \cdot y + x \cdot \overline{y} = x$	*adiacenza*
$(x+y) \cdot (x+\overline{y}) = x$	//

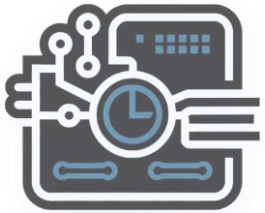
Proprietà (tre variabili)

$x \cdot y + y \cdot z + \overline{x} \cdot z = x \cdot y + \overline{x} \cdot z$	*consenso*
$(x+y) \cdot (y+z) \cdot (\overline{x}+z) = (x+y) \cdot (\overline{x}+z)$	//



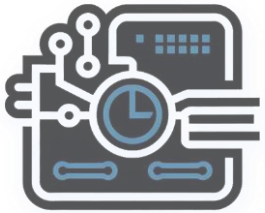
Esempi di dimostrazione

- Fattorizzazione: dimostrare che $(x+y) \cdot (x+z) = x + y \cdot z$



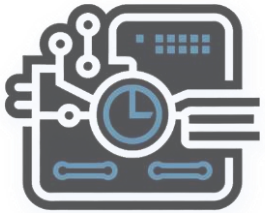
Esempi di dimostrazione

- Fattorizzazione: dimostrare che $(x+y) \cdot (x+z) = x + y \cdot z$
  - $(x+y) \cdot (x+z) = x \cdot x + x \cdot z + x \cdot y + y \cdot z = x + x \cdot z + x \cdot y + y \cdot z$ (idempotenza)



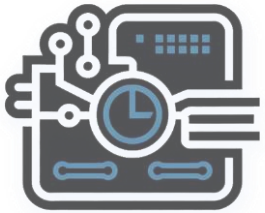
Esempi di dimostrazione

- Fattorizzazione: dimostrare che $(x+y) \cdot (x+z) = x + y \cdot z$
  - $(x+y) \cdot (x+z) = x \cdot x + x \cdot z + x \cdot y + y \cdot z = x + x \cdot z + x \cdot y + y \cdot z$ (idempotenza)
  - $= x \cdot (1+z) + x \cdot y + y \cdot z$ ($x = x \cdot 1$ e prop. associativa)



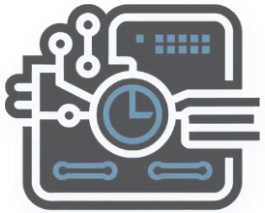
Esempi di dimostrazione

- Fattorizzazione: dimostrare che $(x+y) \cdot (x+z) = x + y \cdot z$
  - $(x+y) \cdot (x+z) = x \cdot x + x \cdot z + x \cdot y + y \cdot z = x + x \cdot z + x \cdot y + y \cdot z$ (idempotenza)
  - $= x \cdot (1+z) + x \cdot y + y \cdot z$ ($x = x \cdot 1$ e prop. associativa)
  - $= x + x \cdot y + y \cdot z$ ($1 + z = 1$)



Esempi di dimostrazione

- Fattorizzazione: dimostrare che $(x+y) \cdot (x+z) = x + y \cdot z$
  - $(x+y) \cdot (x+z) = x \cdot x + x \cdot z + x \cdot y + y \cdot z = x + x \cdot z + x \cdot y + y \cdot z$ (idempotenza)
  - $= x \cdot (1+z) + x \cdot y + y \cdot z$ ($x = x \cdot 1$ e prop. associativa)
  - $= x + x \cdot y + y \cdot z$ ($1 + z = 1$)
  - $= x \cdot (1+y) + y \cdot z$ ($x = x \cdot 1$ e prop. associativa)



Esempi di dimostrazione

- Fattorizzazione: dimostrare che $(x+y) \cdot (x+z) = x + y \cdot z$
  - $(x+y) \cdot (x+z) = x \cdot x + x \cdot z + x \cdot y + y \cdot z = x + x \cdot z + x \cdot y + y \cdot z$ (idempotenza)
  - $= x \cdot (1+z) + x \cdot y + y \cdot z$ ($x = x \cdot 1$ e prop. associativa)
  - $= x + x \cdot y + y \cdot z$ ($1 + z = 1$)
  - $= x \cdot (1+y) + y \cdot z$ ($x = x \cdot 1$ e prop. associativa)
  - $= x + y \cdot z$ ($1 + y = 1$)

Esempi di dimostrazione

- Fattorizzazione: dimostrare che $(x+y) \cdot (x+z) = x + y \cdot z$
  - $(x+y) \cdot (x+z) = x \cdot x + x \cdot z + x \cdot y + y \cdot z = x + x \cdot z + x \cdot y + y \cdot z$ (idempotenza)
  - $= x \cdot (1+z) + x \cdot y + y \cdot z$ ($x = x \cdot 1$ e prop. associativa)
  - $= x + x \cdot y + y \cdot z$ ($1 + z = 1$)
  - $= x \cdot (1+y) + y \cdot z$ ($x = x \cdot 1$ e prop. associativa)
  - $= x + y \cdot z$ ($1 + y = 1$)
- Con il principio di induzione completa, ossia verificando tutte le combinazioni di x, y, z

x	y	z	$x + y$	$x + z$	$(x + y) \cdot (x+z)$	$y \cdot z$	$x + y \cdot z$
0	0	0	0	0	**0**	0	**0**
0	0	1	0	1	**0**	0	**0**
0	1	0	1	0	**0**	0	**0**
0	1	1	1	1	**1**	1	**1**
1	0	0	1	1	**1**	0	**1**
1	0	1	1	1	**1**	0	**1**
1	1	0	1	1	**1**	0	**1**
1	1	1	1	1	**1**	1	**1**

Teorema di De Morgan

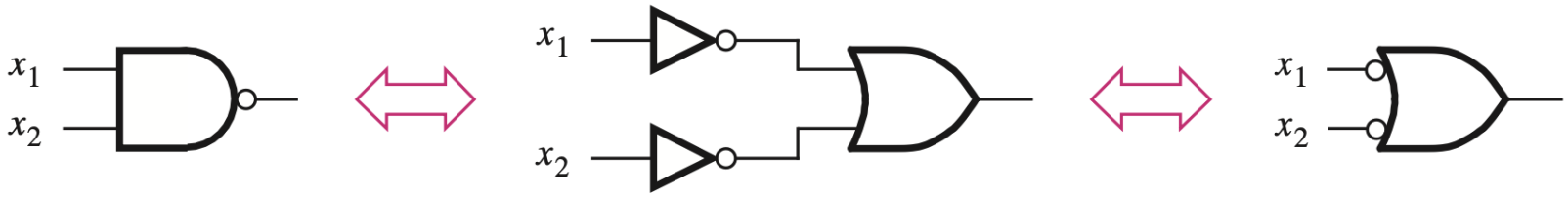
- $\overline{x \cdot y} = \bar{x} + \bar{y}$
- Prova tramite induzione completa

x	y	$x \cdot y$	$\overline{x \cdot y}$	$\bar{x}$	$\bar{y}$	$\bar{x} + \bar{y}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0
		LHS		RHS		

Teorema di De Morgan

- $\overline{x \cdot y} = \bar{x} + \bar{y}$
- Prova tramite induzione completa

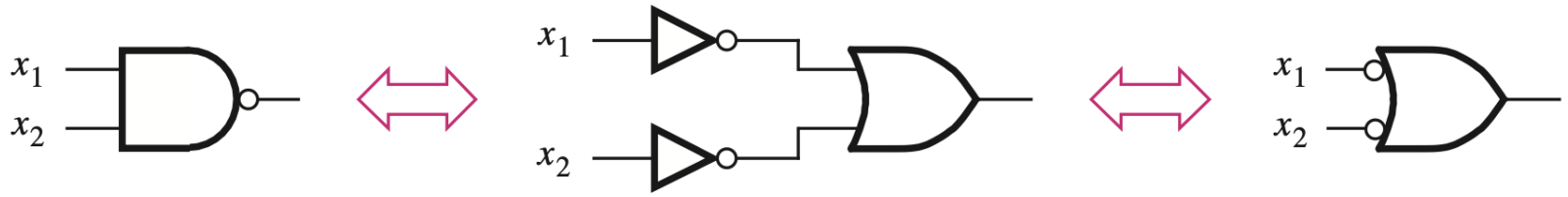
x	y	$x \cdot y$	$\overline{x \cdot y}$	$\bar{x}$	$\bar{y}$	$\bar{x} + \bar{y}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0
		LHS		RHS		



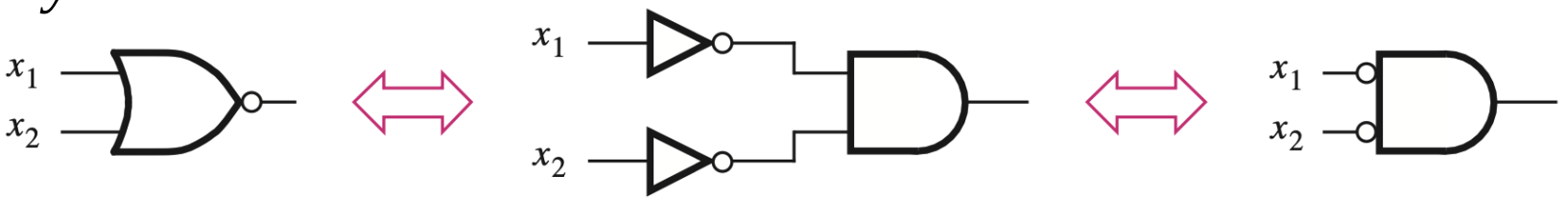
Teorema di De Morgan

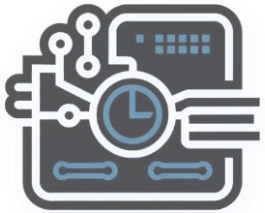
- $\overline{x \cdot y} = \bar{x} + \bar{y}$
- Prova tramite induzione completa

x	y	$x \cdot y$	$\overline{x \cdot y}$	$\bar{x}$	$\bar{y}$	$\bar{x} + \bar{y}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0
		LHS		RHS		



- $\overline{x + y} = \bar{x} \cdot \bar{y}$





Conclusioni

- Valori e segnali logici come astrazione di livelli di tensione e loro variazioni
- Le porte logiche implementano funzioni logiche elementari
  - Per distinguere il valore logico usano una soglia di tensione (V_T), la quale può variare in un range (tra V_{IL} e V_{IH})
  - In condizioni statiche, per avere un valore logico corretto occorre una tensione abbastanza lontana dalla soglia ($V_L < V_{IL}$, $V_H > V_{IH}$), in modo da avere maggiore robustezza al rumore
  - In condizioni dinamiche, occorre che le tensioni stiano per un tempo limitato nel range di variazione della soglia => tempi di salita e discesa brevi
- Le porte logiche impiegano tempo a propagare i valori logici verso l'uscita
  - Ritardi di propagazione
- Circuiti logici combinano porte logiche per eseguire funzioni logiche complesse
  - Circuiti combinatori, uscita dipende dagli ingressi in un dato istante
  - Circuiti sequenziali, uscita dipende dalla "storia", hanno memoria