Capitolo 4

Reti logiche

4.1 - Funzioni, espressioni e schemi logici
4.2 - Algebra di commutazione
4.3 - Famiglie logiche

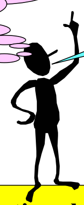
4.1 Funzioni, espressioni e schemi logici

Logica e Reti logiche

1. Tutti gli uomini sono mortali

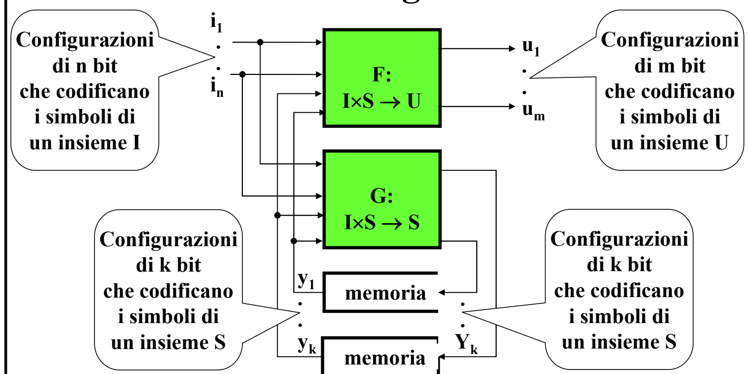
2. Socrate è un uomo

3. Socrate è mortale



<u>Rete logica</u> -Modello matematico che assume come *primitive* alcune semplici modalità di elaborazione di segnali binari e deduce da queste in modo rigoroso
• quale struttura soddisfa un dato comportamento,
• quale comportamento ha una data struttura.

Reti logiche

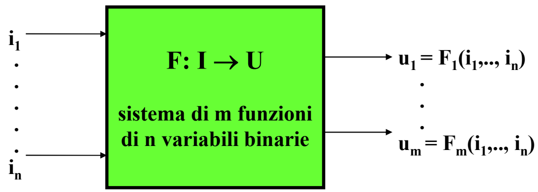


Rete logica sequenziale sincrona

Rete logica sequenziale asincrona

Rete logica combinatoria

Rete logica combinatoria

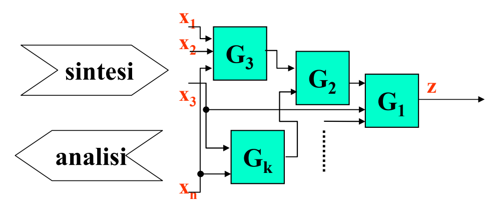


<u>Rete logica combinatoria</u> - I valori dei segnali d'uscita ... dai valori contemp... dei segnali d'ingresso.

bit
funzione
espressione

Struttura & Comportamento di una rete logica combinatoria

$x_1x_2x_3 \ldots x_n$	$z = F(x_1,.., x_n)$
0 0 0 0	0 *oppure* 1
1 0 0 0	0 *oppure* 1
0 1 0 0	0 *oppure* 1
1 1 0 0	0 *oppure* 1
0 0 1 0	0 *oppure* 1
⋮	⋮
0 1 1 1	0 *oppure* 1
1 1 1 1	0 *oppure* 1

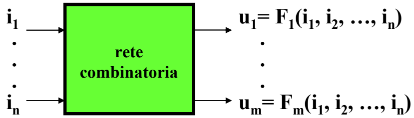


Analisi e Sintesi

- Operatore
- Regole di operazione
- Tabella della verità
- Porta logica
- Funzione
- Espressione
- Composizione di porte
- Schema fisico

Funzioni booleane

Funzioni di variabili binarie



Funzione completa di n variabili binarie $z = F(x_1, x_2, \ldots, x_n)$
Insieme di 2^n coppie ordinate $\{x, z \mid x \in B^n, z \in B\}$ formate da una configurazione di valori delle variabili indipendenti x_i e dal corrispondente valore della variabile dipendente z.

Il numero di distinte funzioni di n variabili binarie è finito.

$$\Phi(n) = 2^{2^n}$$

4 funzioni di 1 variabile,
16 funzioni di 2 variabili,
256 funzioni di 3 variabili,
65.536 funzioni di 4 variabili, ecc.

Funzione incompleta o non completamente specificata
Il dominio è un sottoinsieme di B^n

Esempio: BCD →7 segmenti

Tabelle della verità

Tabella della verità - Descrizione tabellare di una funzione di variabili binarie.

n+1 colonne

2^n righe

x_1 x_2 … x_n	$F(x_1, x_2, \ldots, x_n)$
0 0 0 ……..0	0 *oppure* 1 *oppure* -
1 0 0 ……..0	0 *oppure* 1 *oppure* -
0 1 0 ……..0	0 *oppure* 1 *oppure* -
1 1 0 ……..0	0 *oppure* 1 *oppure* -
0 0 1 ……..0	0 *oppure* 1 *oppure* -
⋮	⋮
0 1 1 ……..1	0 *oppure* 1 *oppure* -
1 1 1 ……..1	0 *oppure* 1 *oppure* -

Funzioni incomplete

Funzioni di una e di due variabili

x	f_0	f_3	f_1	f_2
0	0	1	0	1
1	0	1	1	0

4 funzioni di una variabile

f_0, f_3 : costanti 0 e 1
f_1: identità o buffer
f_2: not

x_0	x_1	f_0	f_{15}	f_3	f_5	f_{12}	f_{10}	f_1	f_{14}	f_7	f_8	f_9	f_6	f_{13}	f_2	f_{11}	f_4
0	0	0	1	0	0	1	1	0	1	0	1	1	0	1	0	1	0
0	1	0	1	0	1	1	0	0	1	1	0	0	1	1	0	0	1
1	0	0	1	1	0	0	1	0	1	1	0	0	1	0	1	1	0
1	1	0	1	1	1	0	0	1	0	1	0	1	0	1	0	1	0

16 funzioni di due variabili

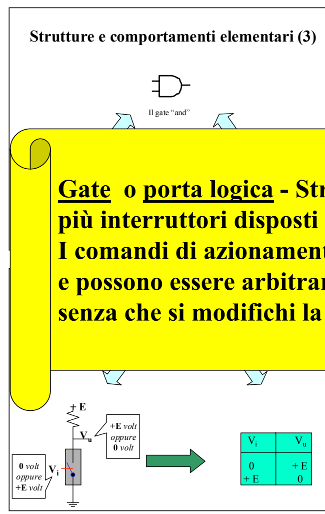
f_0, f_{15} : costanti 0 e 1
f_3 , f_5 : identità o buffer
f_{12} , f_{10} : not

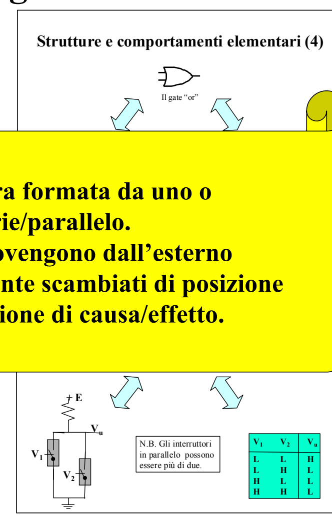
f_1 : and
f_{14} : nand
f_7 : or
f_8 : nor
f_9: equivalence
f_6: ex-or

f_{13}, f_2, f_{11}, f_8 : implicazioni

Dualità: f_{14}-f_8; f_9-f_6
Complemento: f_{14}-f_1; f_7-f_8; f_9-f_6

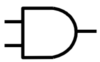
Porte logiche





Gate o porta logica - Struttura formata da uno o più interruttori disposti in serie/parallelo.
I comandi di azionamento provengono dall'esterno e possono essere arbitrariamente scambiati di posizione senza che si modifichi la relazione di causa/effetto.

Dualità tra “and” e “or”(1)



Il gate “and”

I1	I2	AB
0	0	0
1	0	0
0	1	0
1	1	1

Il gate “or”

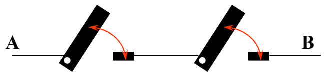
I1	I2	AB
0	0	0
1	0	1
0	1	1
1	1	1

Due differenti astrazioni!

{aperto = 0, chiuso = 1}

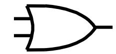
{aperto = 1, chiuso = 0}

Contatti in serie



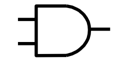
I1	I2	AB
aperto	aperto	aperto
aperto	chiuso	aperto
chiuso	aperto	aperto
chiuso	chiuso	chiuso

Dualità tra “and” e “or”(2)



Il gate “or”

I1	I2	AB
0	0	0
1	0	1
0	1	1
1	1	1



Il gate “and”

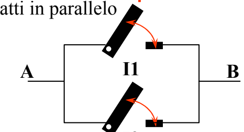
I1	I2	AB
0	0	0
1	0	0
0	1	0
1	1	1

Due differenti astrazioni!

{aperto = 0, chiuso = 1}

{aperto = 1, chiuso = 0}

Contatti in parallelo



I1	I2	AB
aperto	aperto	aperto
aperto	chiuso	chiuso
chiuso	aperto	chiuso
chiuso	chiuso	chiuso

Operazioni logiche

Funzioni e operazioni

*** operatore**

Operazioni a un operando

F(x) = *(x)
F(x) = (x)*

Esempi:
radice
logaritmo
potenza
derivata
modulo

Operazioni a due operandi

F(x,y) = *(x,y)
F(x,y) = x* y

Esempi:
addizione
sottrazione
moltiplicazione
divisione
Min, Max

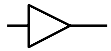
Identità : z = x

Regole:
0 = 0
1 = 1

Funzione:

x	z
0	0
1	1

Realizzazione:

x  z

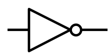
Complementazione : x’ , $\overline{x}$, ⌉x

Regole:
0’ = 1
1’ = 0

Funzione:

x	z
0	1
1	0

Realizzazione:

x  z

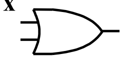
Somma logica: x + y , x ∨ y

Regole:
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 1

Funzione:

x	y	z
0	0	0
0	1	1
1	0	1
1	1	1

Realizzazione:

x
 z
y

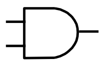
Prodotto logico: x . y , xy , x ∧ y

Regole:
0 . 0 = 0
0 . 1 = 0
1 . 0 = 0
1 . 1 = 1

Funzione:

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

Realizzazione:

x
 z
y

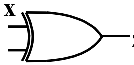
Somma modulo due: x ⊕ y

Regole:
0 ⊕ 0 = 0
0 ⊕ 1 = 1
1 ⊕ 0 = 1
1 ⊕ 1 = 0

Funzione:

x	y	z
0	0	0
0	1	1
1	0	1
1	1	0

Realizzazione:

x
 z
y

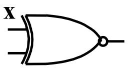
Equivalenza: x ≡ y

Regole:
0 ≡ 0 = 1
0 ≡ 1 = 0
1 ≡ 0 = 0
1 ≡ 1 = 1

Funzione:

x	y	z
0	0	1
0	1	0
1	0	0
1	1	1

Realizzazione:

x
 z
y

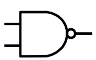
Nand (operazione di Shaffer): z = x ↑ y

Regole:
0 ↑ 0 = 1
0 ↑ 1 = 1
1 ↑ 0 = 1
1 ↑ 1 = 0

Funzione:

x	y	z
0	0	1
0	1	1
1	0	1
1	1	0

Realizzazione:

x
 z
y

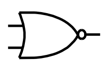
Nor (operazione di Pierce): z = x ↓ y

Regole:
0 ↓ 0 = 1
0 ↓ 1 = 0
1 ↓ 0 = 0
1 ↓ 1 = 0

Funzione:

x	y	z
0	0	1
0	1	0
1	0	0
1	1	0

Realizzazione:

x
 z
z

Operazioni e Espressioni

$f_1(x) = x$ $f_7(x,y) = x + y$ $f_1(x,y) = x \,.\, y$ $f_6(x,y) = x \oplus y$

$f_2(x) = x'$ $f_8(x,y) = x \downarrow y$ $f_{14}(x,y) = x \uparrow y$ $f_9(x,y) = x \equiv y$

<u>Espressione logica</u> - Stringa formata da costanti, bit, operatori logici e parentesi.

Esempi: $(x \oplus y) \oplus (z \oplus w)$ $a + (b.c)$

$(x \downarrow y) \downarrow 0$

Valutazione di una espressione

<u>Valutazione di una espressione di n variabili per una n-pla di valori</u>
1 - Si sostituisce ad ogni variabile il valore che le compete.
2 - Partendo dalle parentesi più interne si sostituisce ogni operazione con il suo risultato fino ad ottenere o la costante 0 o la costante 1.

***Esempio*: $E(a,b,c) = a+(b.c)$ per $a=0$, $b=1$, $c=0$**

$$= 0+(1.0)$$
$$= 0+0$$
$$= 0$$

<u>N° di valutazioni</u> - Una espressione di n variabili può essere valutata in 2^n modi diversi.

Espressioni e Funzioni

Le 2^n valutazioni di una espressione $E(x_1, x_2, \ldots, x_n)$ creano 2^n coppie x, z $\{x, z \mid x \in B^n, z \in B\}$

Esempio: $E(a,b,c) = a+(b.c)$

	a b c	E
$E(0,0,0) = 0+(0.0) = 0$	0 0 0	0
$E(0,0,1) = 0+(0.1) = 0$	0 0 1	0
$E(0,1,0) = 0+(1.0) = 0$	0 1 0	0
$E(0,1,1) = 0+(1.1) = 1$	0 1 1	1
$E(1,0,0) = 1+(0.0) = 1$	1 0 0	1
$E(1,0,1) = 1+(0.1) = 1$	1 0 1	1
$E(1,1,0) = 1+(1.0) = 1$	1 1 0	1
$E(1,1,1) = 1+(1.1) = 1$	1 1 1	1

T1) Ogni espressione descrive una e una sola funzione completa.

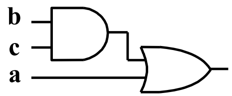
Espressioni e Schemi logici

T2) Ogni espressione descrive una struttura formata da gate connessi in serie e/o in parallelo.

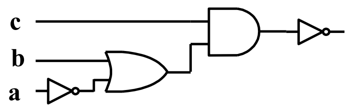
Per individuare lo schema descritto da una espressione:
1 - si parte dalle parentesi più interne e si traccia il simbolo del gate corrispondente all'operazione, collegandone gli ingressi ai segnali esterni;
2 - si procede in modo analogo con le altre coppie di parentesi, considerando via via come ingressi dei nuovi gate anche le uscite di quelli già tracciati.

Esempi

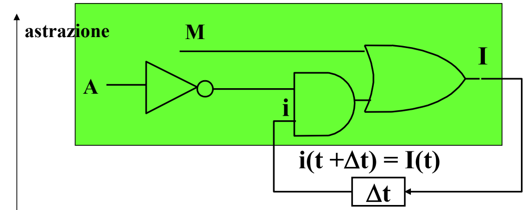
a+(b.c)



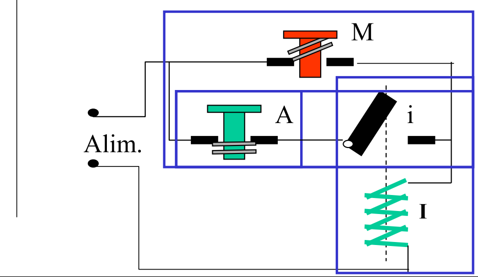
(((a)' + b) . c)'



N.B. - Lo schema logico di una espressione non può avere segnali in <u>retroazione</u> (l'uscita di ogni gate dipende da segnali d'ingresso e/o da uscite di gate disposti "a monte").

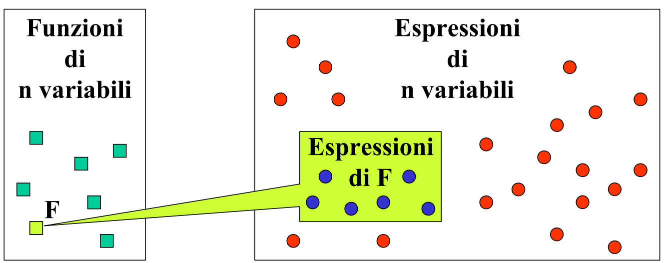


I = f(M,A,i) = M + A'.i



Equivalenza tra espressioni

<u>Espressioni equivalenti</u> - Due espressioni E_1, E_2 sono equivalenti, e si scrive $E_1 = E_2$, se e solo se descrivono la stessa funzione.



Proprietà

T3) proprietà commutativa (+, ., ↓, ↑, ⊕, ≡)

a * b = b * a

T4) proprietà associativa (+, ., ⊕)

(a * b) * c = a * (b * c) **= a * b * c**

T5) complementi:

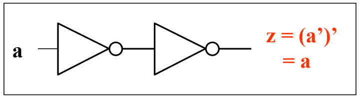
(x + y)' = x ↓ y

(x . y)' = x ↑ y

(x ≡ y)' = x ⊕ y

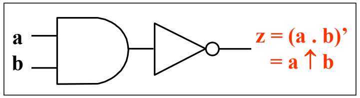
Insiemi di gate (1)

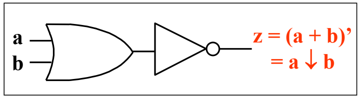
<u>Insieme AND, OR, NOT</u> - Disponendo opportunamente in serie/parallelo soltanto questi tre tipi di gate è possibile ottenere il comportamento di tutti gli altri.

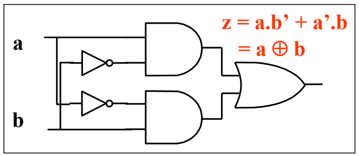


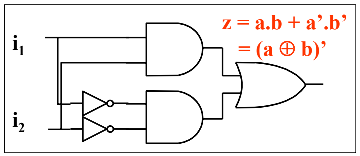
infatti:

a	a'	(a')'
0	1	0
1	0	1



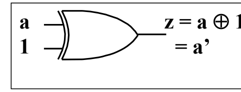






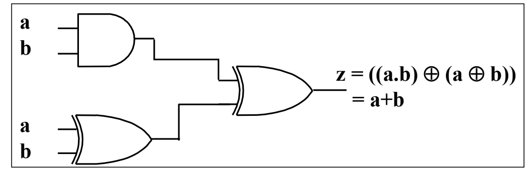
Insiemi di gate (2)

<u>Insieme EX-OR, AND</u> - Disponendo opportunamente in serie/parallelo soltanto questi due tipi di gate è possibile ottenere il comportamento di tutti gli altri.



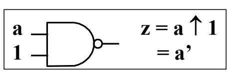
infatti

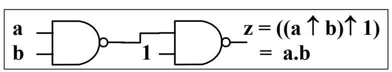
a	a ⊕ 1
0	1
1	0

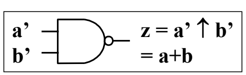


Insiemi di gate (3)

<u>NAND</u> - Disponendo opportunamente in serie/parallelo solo questo tipo di gate è possibile ottenere il comportamento di tutti gli altri.







Dimostrazione per induzione perfetta →

a b	a+b	a'.b'	a'↑b'
0 0	0	1	0
0 1	1	0	1
1 0	1	0	1
1 1	1	0	1

Calcolo delle proposizioni

Assegnata una qualsiasi funzione di variabili binarie, è possibile descriverla con una espressione contenente solo le operazioni eseguite dai gate?

<u>Proposizione</u> - Frase o "vera" o "falsa", formata da affermazioni o "vere" o "false" unite dai connettivi o, e, non.

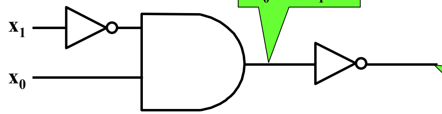
Sia b la proposizione "il bit b vale 1".

La frase "F(x,y) vale 1 se o x vale 1 o y vale 1"
- **descrive la funzione "or"**
- **è equivalente alla proposizione "o x o y"**
  (*vera* per 01,10,11 e *falsa* per 00)
- **è equivalente all'espressione x + y**

"vero"→ 1
"falso"→ 0
"e"→ .
"o"→ +
"non"→ '

Sintesi di una delle implicazioni

1 se e solo se x_0=1 e x_1=0



non 1 se e solo se x_0=1 e x_1=0

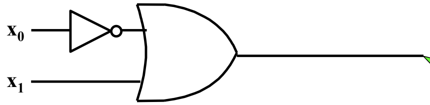
x_0 x_1	f_{13}
0 0	1
0 1	1
1 0	0
1 1	1

$(x_0 . x_1')'$

non (x_0 e non x_1)

non x_0 o x_1

$x_0' + x_1$



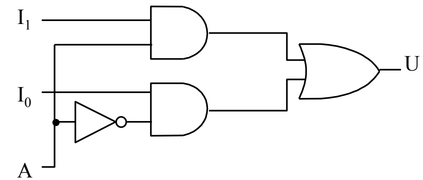
1 se e solo se o x_0=0 o x_1=1

Sintesi di un SELETTORE a due vie

A	I_0	I_1	U
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

o non A e I_0 o A e I_1

$A' . I_0 + A . I_1$



Algebre binarie

Algebra binaria - Sistema matematico formato da un insieme di operatori definiti assiomaticamente ed atti a descrivere con una espressione ogni funzione di variabili binarie

Calcolo delle proposizioni
{*vero*, *falso*} {*e*, *o*, *non*}
tre operatori

G. Boole (1854)

↓

Algebra di commutazione
{0, 1} {+, . , '}
tre operatori

C. Shannon (1938)

Algebra del nand
{0, 1} {↑}
un operatore

Algebra del nor
{0, 1} {↓}
un operatore

Algebra lineare
{0, 1} { ⊕ , .}
due operatori

Esercitazione N.5

1 – Valutazione di espressioni

2 – Uso dei diagrammi di Venn

3 – Individuazione di uno schema

4.2
Algebra di commutazione

Algebra di commutazione

1) **Costanti**: 0, 1

2) **Operazioni**:

somma logica (+) prodotto logico (.) complementazione (')

3) **Postulati**:

0 + 0 = 0	0 . 0 = 0	0' = 1
1 + 0 = 1	1 . 0 = 0	1' = 0
0 + 1 = 1	0 . 1 = 0	
1 + 1 = 1	1 . 1 = 1	

4) **Variabili**: simboli sostituibili o con 0 o con 1

Espressioni

5) **Espressione** - Stringa finita di costanti, variabili, operatori e parentesi, formata in accordo con le seguenti regole:
1) 0 e 1 sono espressioni
2) una variabile è una espressione
3) se A è un'espressione, lo è anche (A')
4) se A, B sono espressioni, lo sono anche (A+B), (A.B)

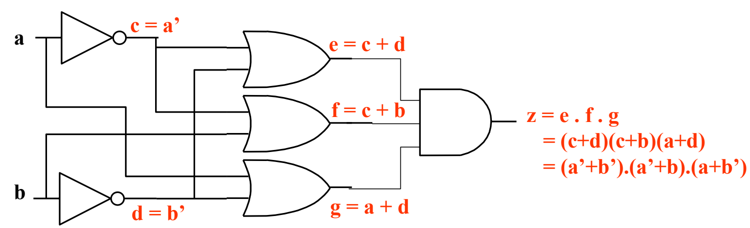
Esempi:

a+(b.c) a + bc

a'.b (a+b)' a'b + 0 + ab'

N.B - L'operazione di prodotto è prioritaria rispetto alla somma e non è obbligatorio racchiuderla tra parentesi. La notazione AB indica A.B

Schemi logici e Espressioni



Teoremi di equivalenza

Equivalenze notevoli

Proprietà della somma e del prodotto logico:

E1) *commutativa*	x + y	=	y + x
	x . y	=	y . x
E2) *associativa*	(x + y) + z	=	x + y + z
	(x . y) . z	=	x . y. z
E3) *distributiva*	(x . y) + (x . z)	=	x . (y + z)
	(x + y) . (x + z)	=	x + (y . z)
E4) *idempotenza*	x + x	=	x
	x . x	=	x
E5) *identità*	x + 0	=	x
	x . 1	=	x
E6) *limite*	x + 1	=	1
	x . 0	=	0

Equivalenze notevoli

Proprietà della complementazione:

E7) *involuzione*	(x ') ' = x
E8) *limitazione*	x + x ' = 1
	x . x ' = 0
E9) *combinazione*	xy + xy' = x
	(x+y).(x+y') = x
E10) *I^a legge di De Morgan*	(x + y) ' = x ' . y '
II^a legge di De Morgan	(x . y) ' = x ' + y '
E11) *consenso*	xy + x'z + yz = xy + x'z
	(x+y).(x'+z).(y+z) = (x+y).(x'+z)

Espressioni di funzioni incomplete

ENCODER a 3 ingressi

x_2	x_1	x_0	z_1	z_0
0	0	0	0	0
1	0	0	1	1
0	1	0	1	0
0	0	1	0	1

N.B. le altre configurazioni sono per ipotesi impossibili

x_2	x_1	x_0	z_1	z_0
0	0	0	0	0
1	0	0	1	1
0	1	0	1	0
0	0	1	0	1
1	1	0	-	-
1	0	1	-	-
0	1	1	-	-
1	1	1	-	-

Espressioni di funzioni incomplete

Espressioni equivalenti di funzioni incomplete - Espressioni che forniscono eguale valutazione limitatamente al dominio di una funzione incompleta sono dette equivalenti.

Espressioni per l'ENCODER:

$z_1 = x_2\, x_1' x_0' + x_2'\, x_1\, x_0'$
$z_0 = x_2\, x_1' x_0' + x_2'\, x_1' x_0$

$u_1 = x_2 + x_1$
$u_0 = x_2 + x_0$

x_2	x_1	x_0	z_1	u_1	z_0	u_0
0	0	0	0	0	0	0
1	0	0	1	1	1	1
0	1	0	1	1	0	0
0	0	1	0	0	1	1
1	1	0	0	1	0	1
1	0	1	0	1	0	1
0	1	1	0	1	0	1
1	1	1	0	1	0	1

Espressioni canoniche

Espressioni canoniche

T6) *Espressione canonica SP (Somma di Prodotti)*

Iª forma canonica - Ogni funzione di n variabili è descritta da una somma di tanti prodotti logici quante sono le configurazioni per cui vale 1. In ciascun prodotto, o mintermine, appare ogni variabile, in forma vera se nella configurazione corrispondente vale 1, in forma complementata se vale 0.

T7) *Espressione canonica PS (Prodotto di Somme)*

IIª forma canonica - Ogni funzione di n variabili è descritta da un prodotto di tante somme logiche quante sono le configurazioni per cui vale 0. In ciascuna somma, o maxtermine, appare ogni variabile, in forma vera se nella configurazione corrispondente vale 0, in forma complementata se vale 1.

Espressioni canoniche della funzione "a implica b"

a b	a→b
0 0	1
0 1	1
1 0	0
1 1	1

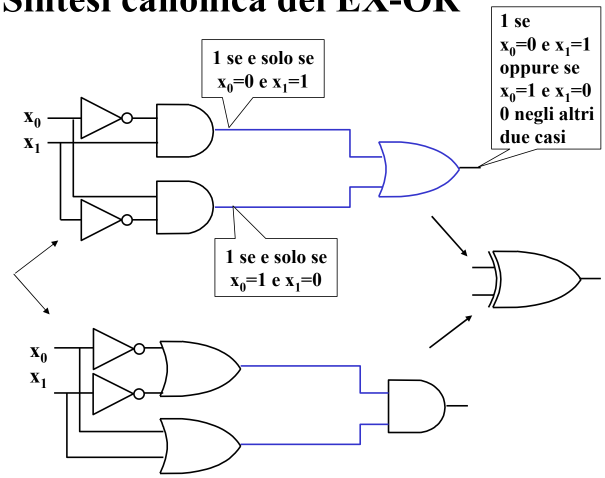
IIª forma canonica:
F(a,b) = a' + b

Iª forma canonica:
F(a,b) = a' . b' + a' . b + a . b

Verifica della equivalenza per manipolazione algebrica:
F(a,b) = a' . b' + a' . b + a . b
= a' . (b' + b) + a . b — E3
= a'.1 + a . b — E8
= a' + a . b — E5
= a' + a . b + a' . b — una parte è inclusa nel tutto
= a' + b — E3, E8, E5

Sintesi canonica del EX-OR

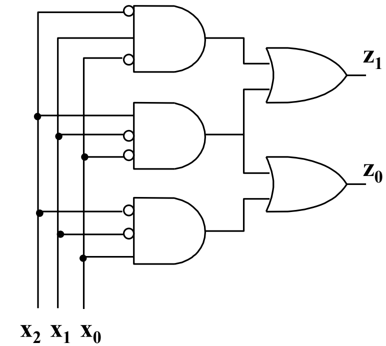
x_0	x_1	$x_0 \oplus x_1$
0	0	0
0	1	1
1	0	1
1	1	0



Sintesi di un ENCODER a tre ingressi

x_2	x_1	x_0	z_1	z_0
0	0	0	0	0
1	0	0	1	1
0	1	0	1	0
0	0	1	0	1

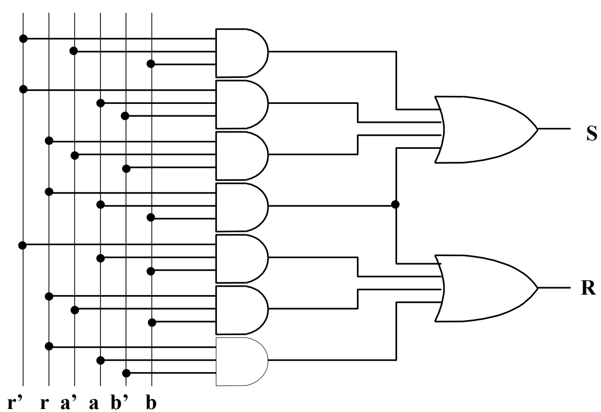
N.B. le altre configurazioni sono per ipotesi impossibili



$$z_1 = x_2\ x_1'x_0' + x_2'\ x_1\ x_0'$$
$$z_0 = x_2\ x_1'x_0' + x_2'\ x_1'x_0$$

Sintesi canonica del Full Adder

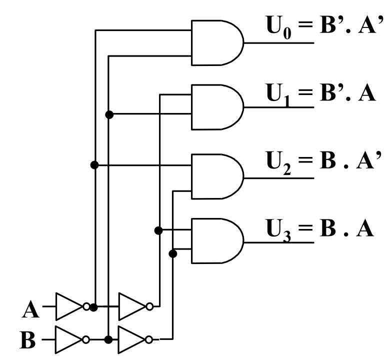
$$S = r'.\,a'.\,b + r'.\,a.\,b' + r.\,a'.\,b' + r.\,a.\,b$$
$$R = r'.\,a.\,b + r.\,a'.\,b + r.\,a.\,b' + r.\,a.\,b$$



Sintesi del trascodificatore da binario a 1 su N

Esempio: Trascodifica 2:4

B	A	U_0	U_1	U_2	U_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



Manipolazione algebrica

A I_0 I_1	U
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

Iª forma canonica:

$U = A' . I_0 . I_1' + A' . I_0 . I_1 + A . I_0' . I_1 + A . I_0 . I_1$

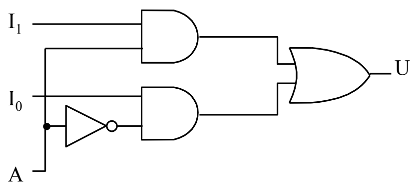
4 AND a 3 ingressi e 1 OR a 4 ingressi

forme equivalenti ottenute per "manipolazione":

$= A' . I_0 . (I_1' + I_1) + A . (I_0' + I_0) . I_1$

$= A' . I_0 . 1 + A . 1 . I_1$

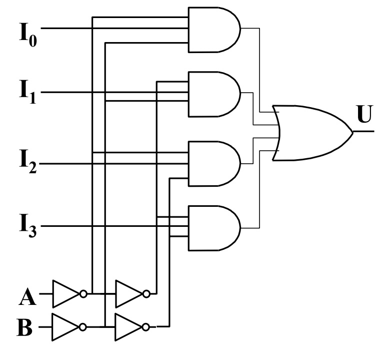
$= A' . I_0 + A . I_1$



Il Selettore a quattro vie

MUX a 4 vie (espressione SP)

$U = A_1' . A_0' . I_0 + A_1' . A_0 . I_1 +$
$+ A_1 . A_0' . I_2 + A_1 . A_0 . I_3$



Notazioni simboliche

Notazioni simboliche per le espressioni canoniche

i	r a b	R	S
0	0 0 0	0	0
1	0 0 1	0	1
2	0 1 0	0	1
3	0 1 1	1	0
4	1 0 0	0	1
5	1 0 1	1	0
6	1 1 0	1	0
7	1 1 1	1	1

$S(r,a,b) = \Sigma_3\, m\,(1,2,4,7)$

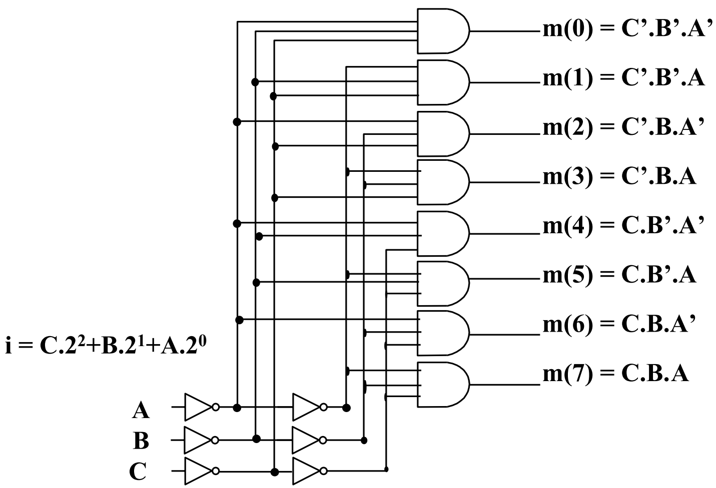
$S(r,a,b) = \Pi_3\, M\,(0,3,5,6)$

$R(r,a,b) = \Sigma_3\, m\,(3,5,6,7)$

$R(r,a,b) = \Pi_3\, M\,(0,1,2,4)$

- **m(i) :** mintermine di n bit che assume il valore 1 solo per la n-pla di valori delle variabili corrispondente all'indice i
- **M(i) :** maxtermine di n bit che assume il valore 0 solo per la n-pla di valori delle variabili corrispondente all'indice i

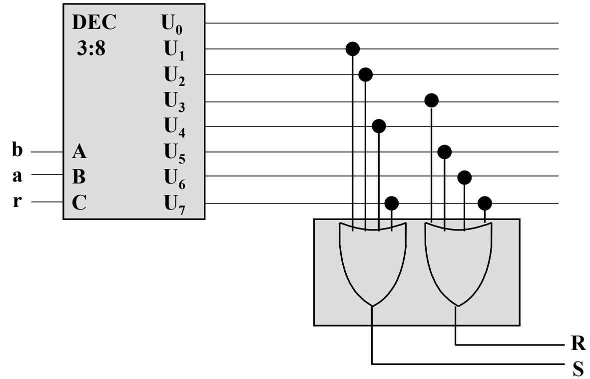
Decoder 3:8



Sintesi del Full Adder con Decoder e Or

$S = \Sigma_3\, m\,(1,2,4,7)$

$R = \Sigma_3\, m\,(3,5,6,7)$



Espressioni generali

Teoremi di espansione (o di Shannon)

T8) $E(x_1,x_2,..,x_{n-1},x_n) = x_n'.E(x_1,x_2,..,x_{n-1},0) + x_n.E(x_1,x_2,..,x_{n-1},1)$

T9) $E(x_1,x_2,..x_{n-1},x_n) = (x_n+E(x_1,x_2,..,x_{n-1},0)).(x_n'+E(x_1,x_2,..,x_{n-1},1))$

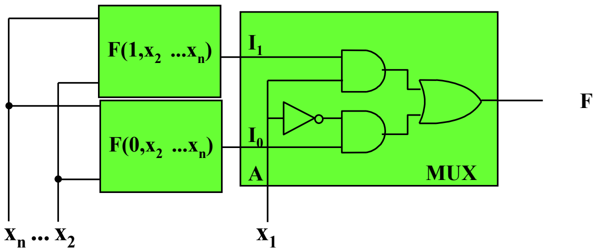
Esempio:

$E= x_1+x_2\,x_3'$

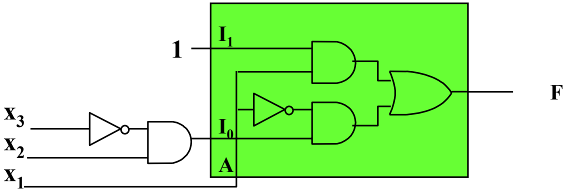
$= x_1'.(0+x_2\,x_3')+x_1.(1+x_2\,x_3')$

$= (x_1+(0+x_2\,x_3')).(x_1'+(1+x_2\,x_3'))$

Mux e teoremi di espansione



Esempio : $x_1+x_2\,x_3' = x_1'\,(0+\ x_2\,x_3') + x_1\,(1+\ x_2\,x_3')$



Applicazione iterata dei teoremi di espansione

$E(x_1x_2\,x_3) = x_1+x_2\,x_3'$

$= x_1'(0+x_2x_3')+x_1(1+x_2x_3')$

$= x_1'x_2'(0+0.x_3')+x_1'x_2(0+1.x_3')+ x_1x_2'(1+0.x_3')+ x_1x_2(1+1.x_3')$

$= x_1'x_2'\,x_3'(0+0.0)\ +$ m(0).E(0) +

$x_1'x_2'\,x_3\,(0+0.1)\ +$ m(1).E(1) +

$x_1'x_2\,\,x_3'(0+1.0)\ +$ m(2).E(2) +

$x_1'x_2\,\,x_3\,(0+1.1)\ +$ m(3).E(3) +

$x_1\,x_2'\,x_3'(1+0.0)\ +$ m(4).E(4) +

$x_1\,x_2'\,x_3\,(1+0.1)\ +$ m(5).E(5) +

$x_1\,x_2\,\,x_3'(1+1.0)\ +$ m(6).E(6) +

$x_1\,x_2\,\,x_3\,(1+1.1)$ m(7).E(7)

Espressioni generali

T10 e T11)- Ogni funzione è descritta da una espressione in cui compaiono o tutti i mintermini o tutti i maxtermini:

$$F(x_1,x_2,\ldots x_i,\ldots x_n) = \sum_{i=0}^{2^n-1} m(i)\cdot F(i) \qquad \text{(SP)}$$

$$F(x_1,x_2,\ldots x_i,\ldots x_n) = \prod_{i=0}^{2^n-1} (\,M(i) + F(i)) \qquad \text{(PS)}$$

Caso SP

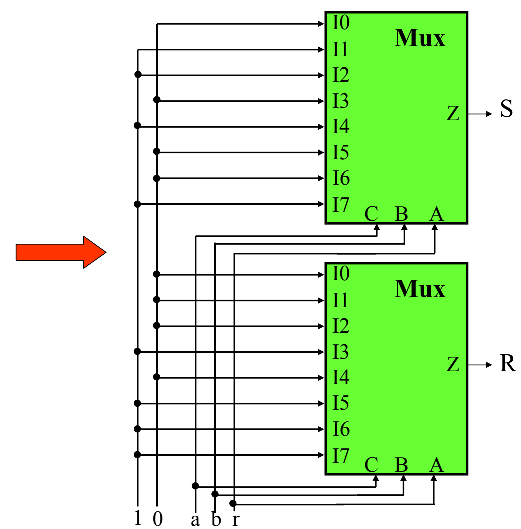
m(i) : mintermine di n bit
F(i): valore dalla funzione per la n-pla di valori delle variabili per cui m(i)=1

Caso PS

M(i) : maxtermine di n bit
F(i): valore dalla funzione per la n-pla di valori delle variabili per cui M(i)=0

Sintesi di un full-adder con MUX

a	b	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



Esercitazione N.6

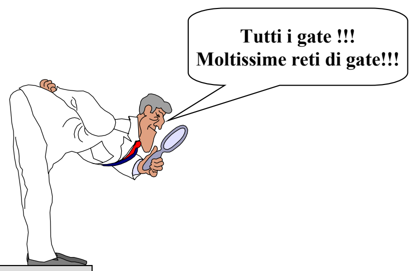
1 – Individuazione dei mintermini

2 – Uso dei multiplexer

3 – Analisi di uno schema

4.3
Famiglie logiche

Famiglie di circuiti logici integrati



Livello logico

Livello fisico

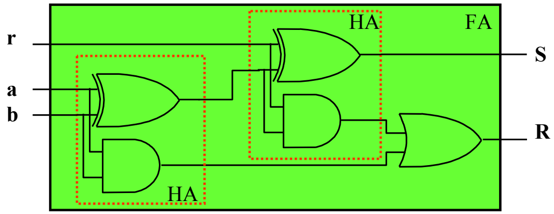
Full Adder con AND, OR e EX-OR

$S = r' . a' . b + r' . a . b' + r . a' . b' + r . a . b$
$R = r' . a . b + r . a' . b + r . a . b' + r . a . b$

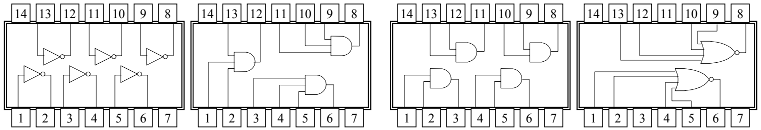
manipolazione algebrica:

$S = r' . (a' . b + a . b') + r . (a' . b' + a . b)$
$S = r' . (a \oplus b) + r . (a \oplus b)'$
$S = r \oplus (a \oplus b)$

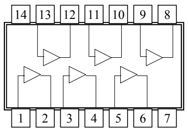
$R = (r' + r) . a . b + r . (a' . b + a . b')$
$R = a . b + r . (a \oplus b)$



Famiglie di gate (TTL SSI -1968/74)



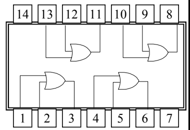
SN7404 SN7411 SN7408 SN7423



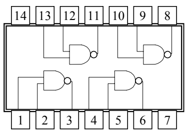
SN7407

Famiglia di circuiti logici:

- alimentazione e consumo
- segnali e soglie
- fan-out (n° max. di ingressi collegabili all'uscita)
- velocità di commutazione

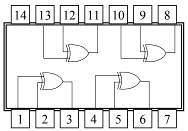


SN7432



SN7400

....

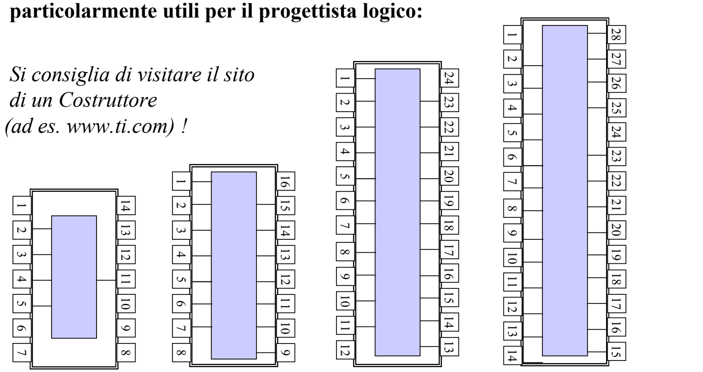


SN7498

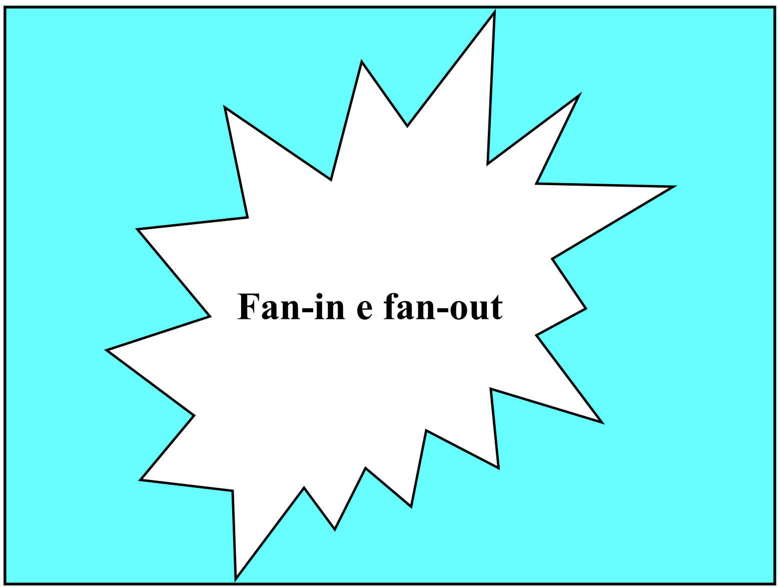
Circuiti combinatori MSI e LSI

Sono disponibili come parti elementari anche reti di gate particolarmente utili per il progettista logico:

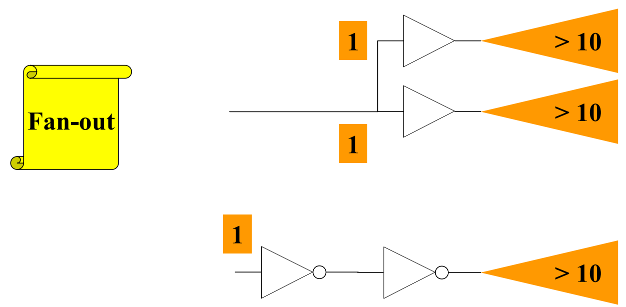
Si consiglia di visitare il sito di un Costruttore (ad es. www.ti.com) !



Full adder **Decoder Multiplexer** **Aritmetica Registro** **Trascodifica Buffer** **… µP RAM**



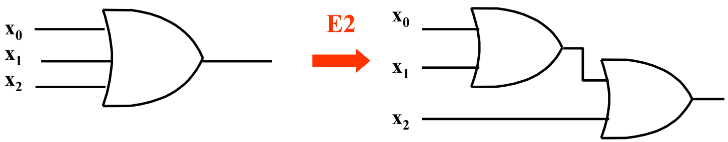
Buffer e Not



And e Or: proprietà associativa

Fan-in

Gate con un massimo di otto ingressi



Parità con EX-OR (1)

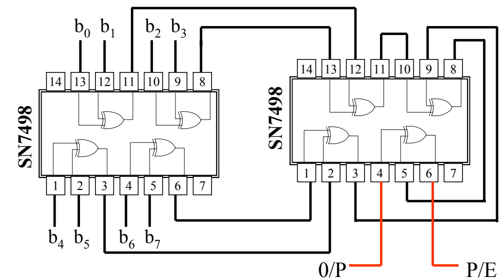
Fan-in

$$P = b_0 \oplus b_1 \oplus b_2 \oplus b_3 \oplus .. \oplus b_7$$

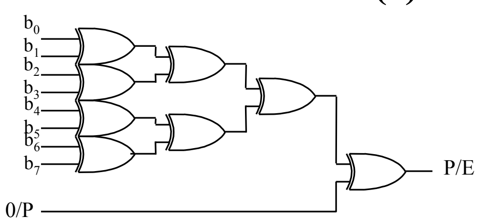
N.B. L'operazione di somma modulo due è associativa

$$P = ((b_0 \oplus b_1) \oplus (b_2 \oplus b_3)) \oplus ((.. \oplus b_7))$$

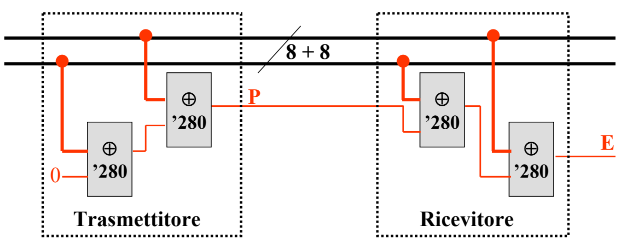
$$E = P \oplus (((b_0 \oplus b_1) \oplus (b_2 \oplus b_3)) \oplus ((.. \oplus b_7)))$$



Parità con EX-OR (2)

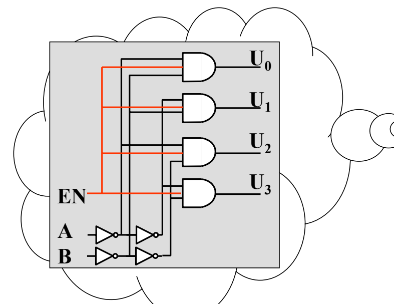


Generazione parità e rilevazione errori singoli su dati da due byte:

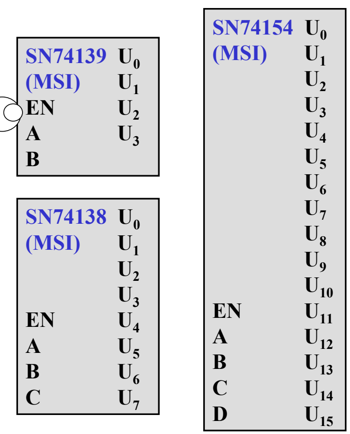


Il circuito integrato DECODER

<u>Decoder</u> o <u>Rete di decodifica</u> - Rete logica combinatoria che realizza i 2^n distinti mintermini di n variabili (*n = 2,3,4*)

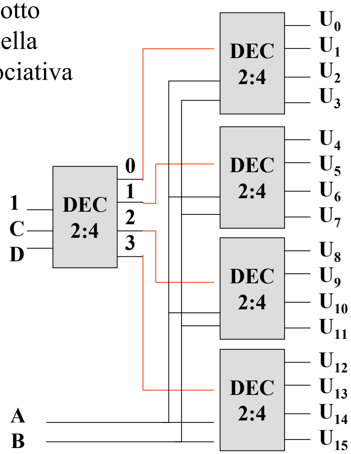


Quando EN=1, vale 1 l'uscita il cui pedice, in decimale, corrisponde al numero binario in ingresso (A bit di minor peso)



Composizione modulare di un Decoder 4:16

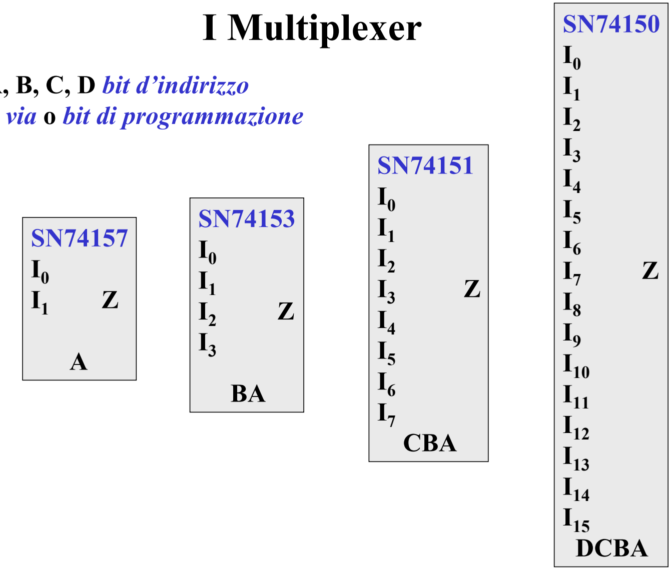
N.B. - Il prodotto logico gode della proprietà associativa



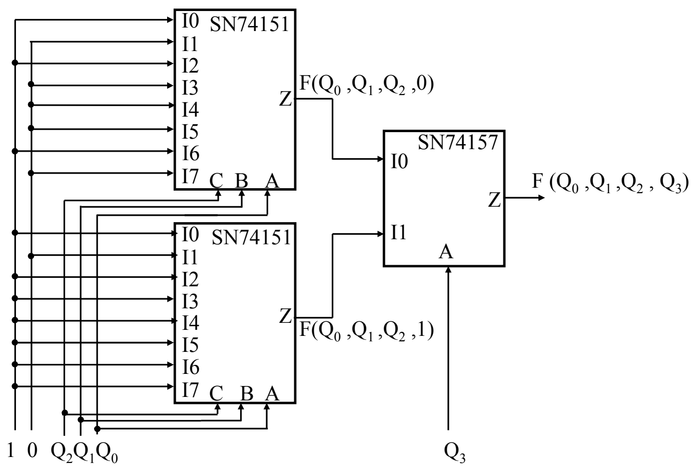
I Multiplexer

A, B, C, D *bit d'indirizzo*

I_i *via* **o** *bit di programmazione*



Sintesi a MUX di funzioni di 4 variabili



Sintesi con MUX a n-1 bit d'indirizzo

Q_3 \ $Q_2Q_1Q_0$	000	001	010	011	100	101	110	111
0	1	0	1	0	0	0	1	0
1	1	0	0	1	1	1	0	1

$F(0,0,0,Q_3) = 1$

$F(0,0,1,Q_3) = 0$

$F(0,1,0,Q_3) = Q_3'$

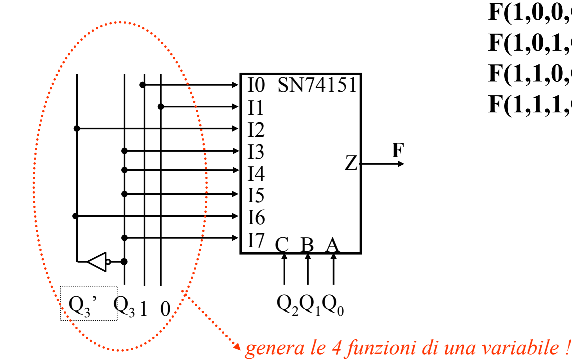
$F(0,1,1,Q_3) = Q_3$

$F(1,0,0,Q_3) = Q_3$

$F(1,0,1,Q_3) = Q_3$

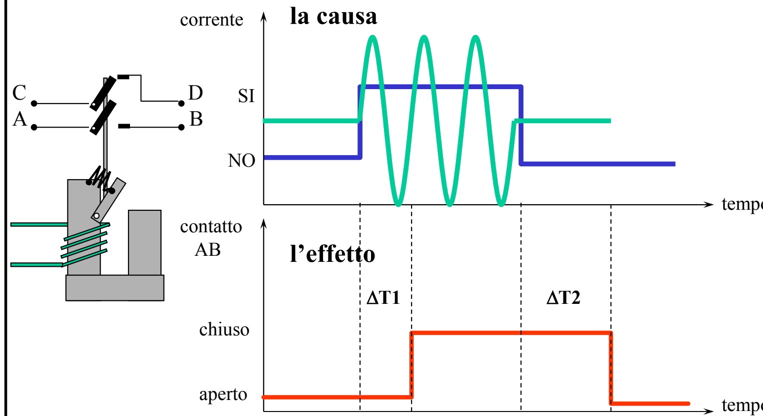
$F(1,1,0,Q_3) = Q_3'$

$F(1,1,1,Q_3) = Q_3$

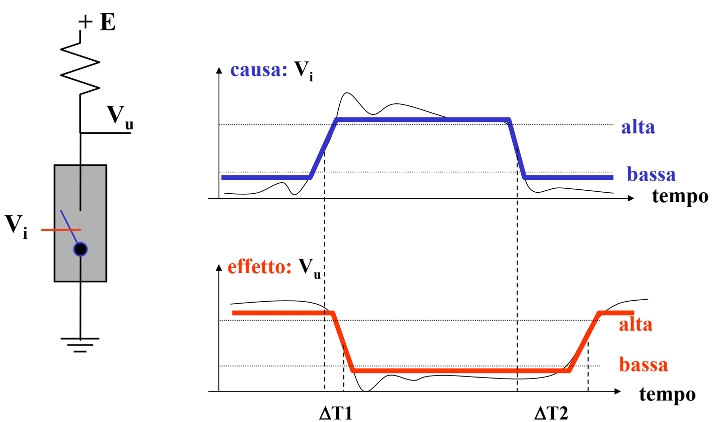


Tempo di propagazione

Tempo di propagazione: il fenomeno del ritardo nel relè



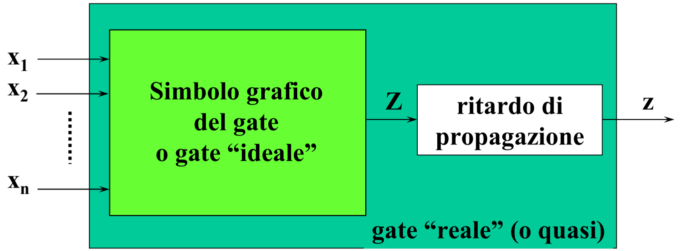
Velocità di commutazione: il ritardo del Not elettronico



Il ritardo sui fronti

- Il <u>ritardo sui fronti di salita</u> (τ_{LH}) <u>e di discesa</u> (τ_{HL}) è presente in ogni tipo di gate e varia in modo notevole da dispositivo a dispositivo.
- A causa della marcata differenza dei due valori, la durata di una situazione H o L in ingresso ad un gate è diversa dalla corrispondente situazione in uscita.
- A causa della "inerzia" del gate, un segnale di ingresso "impulsivo" e "troppo stretto" può <u>non</u> essere avvertito in uscita.

Un modello più realistico per il gate



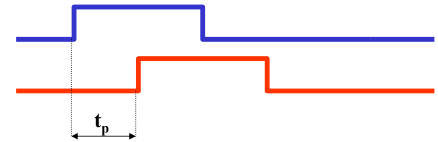
$Z = F(x_1, x_2, .., x_n)$
$z(t) = Z(t\text{-}t_p)$

N.B. - I Costruttori di famiglie logiche forniscono i valori *minimo*, *nominale* e *massimo* di t_p

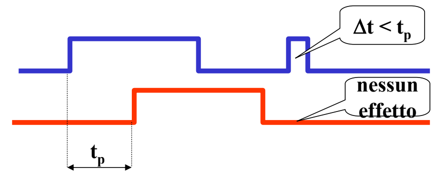
Il ritardo di propagazione

ritardo di propagazione: $t_p = \max(\tau_{LH}, \tau_{HL})$

- **Ritardo puro**



- **Ritardo inerziale**

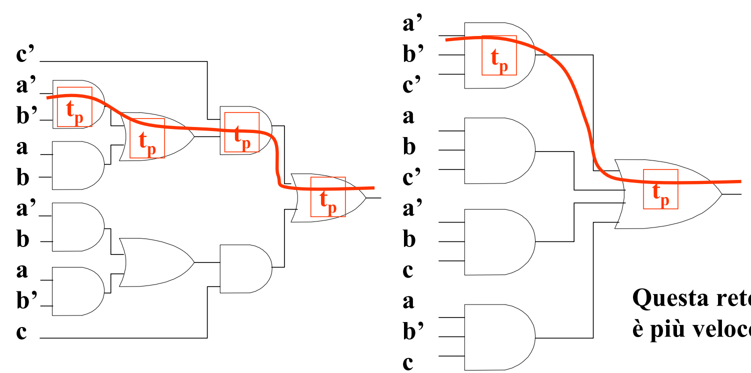


Il modello del ritardo inerziale è il più vicino alla realtà.
Il ritardo puro (o matematico) è però più facile da simulare.

Comportamento in transitorio

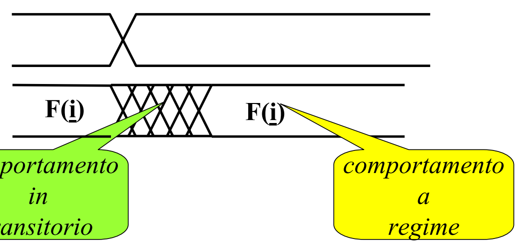
Velocità e lunghezza dei percorsi

(a’.b’+a.b).c’+(a’.b+a.b’).c = a’.b’.c’ +a.b.c’+a’.b. c +a.b’.c

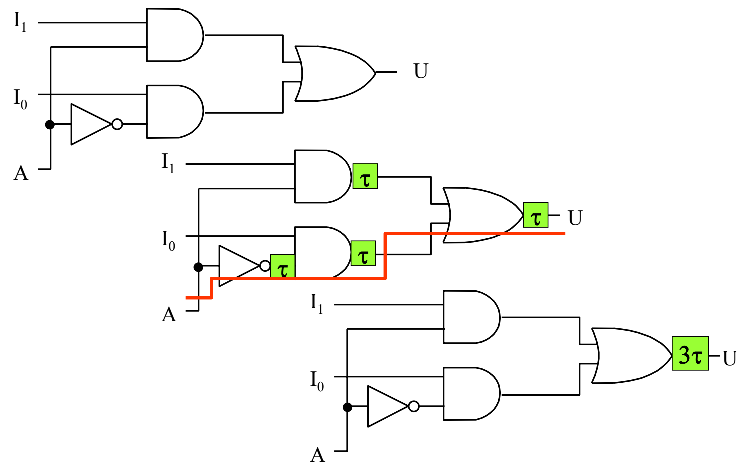


Comportamento *a regime* e *in transitorio* dei circuiti combinatori

I nuovi valori dei segnali di ingresso di una rete combinatoria devono propagarsi all'interno della struttura prima di riuscire ad imporre al segnale d'uscita il valore che ad essi deve corrispondere. Ciò determina un comportamento <u>in transitorio</u>, che in generale sarà diverso da quello <u>a regime</u>.



Stima della durata del transitorio (metodo del *caso peggiore*)



Ritardi dei MUX

SN54150, SN54151A, SN74150, SN74151A
DATA SELECTORS/MULTIPLEXERS

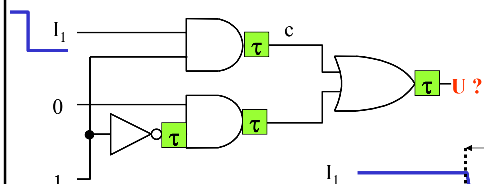
switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER[¶]	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	'150 MIN	'150 TYP	'150 MAX	'151A MIN	'151A TYP	'151A MAX	UNIT
t_{PLH}	A, B, or C (4 levels)	Y	C_L = 15 pF, R_L = 400 Ω, See Note 4					25	38	ns
t_{PHL}								25	38	
t_{PLH}	A, B, C, or D (3 levels)	W			23	[illegible]		17	26	ns
t_{PHL}					22	33		19	30	
t_{PLH}	Strobe $\overline{G}$	Y						21	33	ns
t_{PHL}								22	33	
t_{PLH}	Strobe $\overline{G}$	W			15.5	24		14	21	ns
t_{PHL}					21	30		15	23	
t_{PLH}	D0 thru D7	Y						13	20	ns
t_{PHL}								18	27	
t_{PLH}	E0 thru E15, or D0 thru D7	W			[illegible]	[illegible]		8	14	ns
t_{PHL}					[illegible]	[illegible]		8	14	

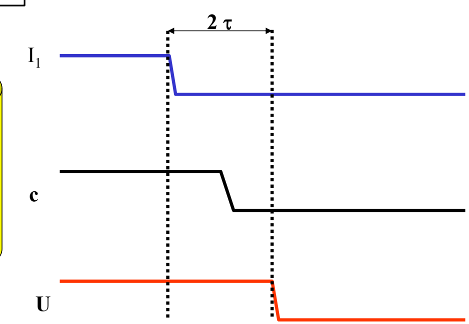
[¶] t_{PLH} = propagation delay time, low-to-high-level output
t_{PHL} = propagation delay time, high-to-low-level output
NOTE 4: Load circuits and voltage waveforms are shown in Section 1.

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

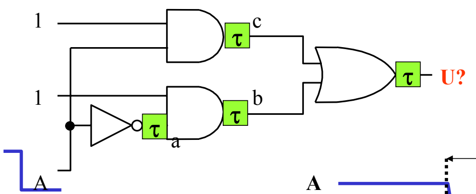
Tipi di transitorio: il ritardo

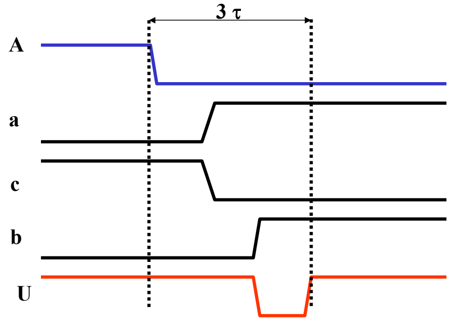


<u>Tipo "ritardo"</u> - L'uscita mantiene il vecchio valore per tutto il transitorio



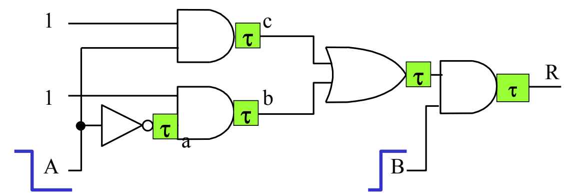
Tipi di transitorio: l'alea statica

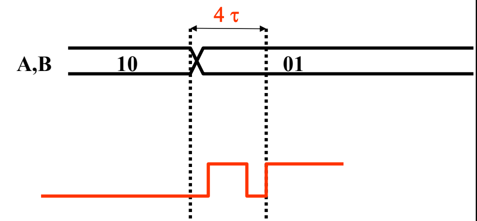




<u>Tipo "alea statica"</u> - L'uscita, che dovrebbe rimanere costante, assume temporaneamente l'altro valore.

Tipi di transitorio: l'alea dinamica



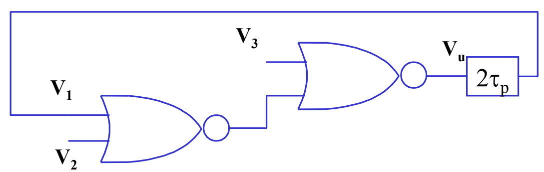


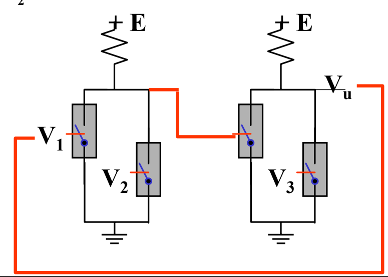
<u>Tipo "alea dinamica"</u> - L'uscita varia più volte prima di assestarsi sul nuovo valore.

Retroazione

Rete ideale, ritardo e retroazione

$V_u = V_3 \downarrow (V_1 \downarrow V_2) \qquad V_1 (t + 2\tau_p) = V_u (t)$





Descrizione della retroazione

Livello fisico

Livello logico

Segnale in retroazione

Variabile dipendente **Y**

Variabile indipendente **y**

Latch SR a NOR

$Q = R \downarrow (q \downarrow S)$
$Q = (R + (q + S)')'$
$Q = R' . (q + S)$

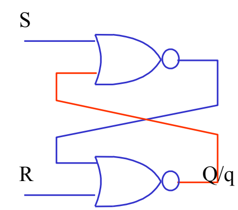
S , R

q	00	01	11	10
0	0	0	-	1
1	1	0	-	1

Q

$V_2 = V_3 = 1$ vietato !
Pongo $S = V_2$
$R = V_3$
$Q = V_u$

$V_u = V_3 \downarrow (V_1 \downarrow V_2)$



Latch SR a NAND

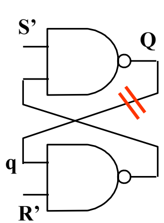
S'R'	Q
01	1
10	0
11	q

S R

q	00	01	11	10
0	0,1	0,1	1,1	1,1
1	1,0	0,1	1,1	1,0

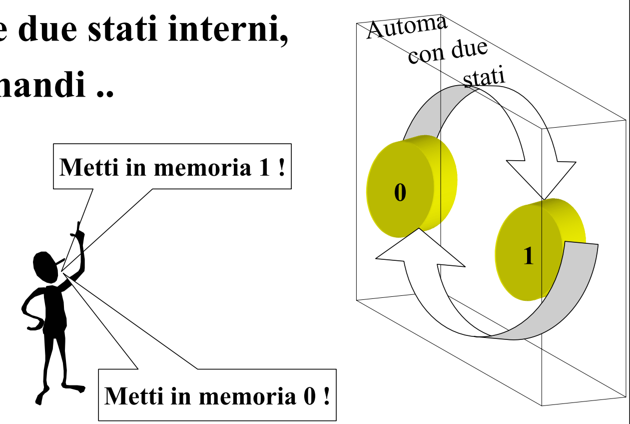
Q

analisi

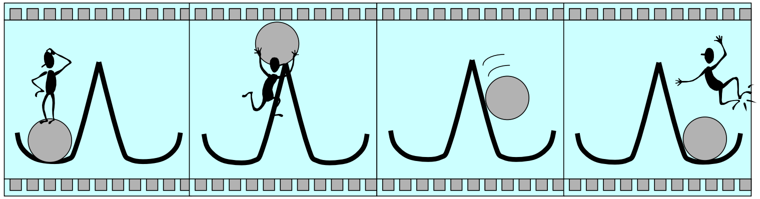


$Q = S' \uparrow (q \uparrow R')$
$Q = (S' . (q . R')')'$
$Q = S + q.R'$

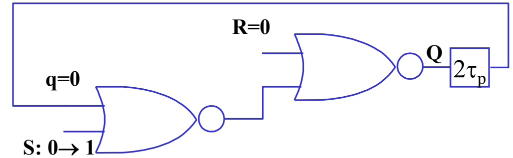
La memorizzazione di un bit richiede due stati interni, due comandi ..

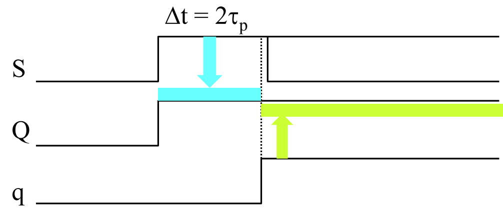


… e un po' di fatica !



Cosa occorre per scrivere un 1 (o uno 0)?





Dopo Δt dal fronte di salita di S, q passa da 0 a 1, condizione che si mantiene anche se S torna a 0

La durata minima di un comando di set/reset è spesso indicata con la denominazione di *tempo di set-up* del latch

SN54279, SN54LS279A, SN74279, SN74LS279A
QUADRUPLE $\overline{S}$-$\overline{R}$ LATCHES

SDLS093 – DECEMBER 1983 – REVISED MARCH 1988

recommended operating conditions

		SN54279			SN74279			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.8			0.8	V
I_{OH}	High-level output current			– 0.8			– 0.8	mA
I_{OL}	Low-level output current			16			16	mA
t_w	Pulse duration, low	20			20			ns
T_A	Operating free-air temperature	– 55		125	0		70	°C

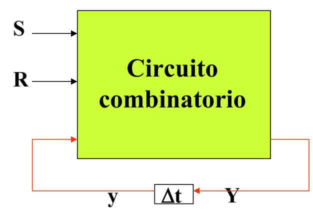
switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 3)

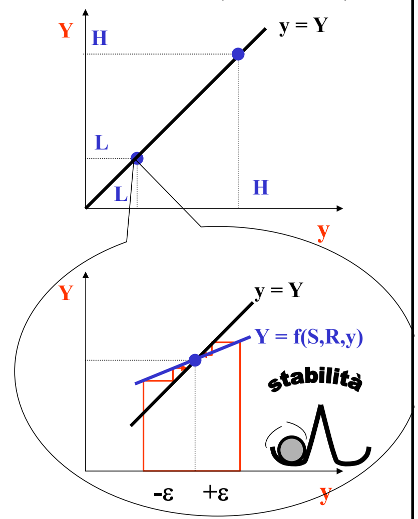
PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	$\overline{S}$	Q	R_L = 400 Ω, C_L = 15 pF		12	22	ns
t_{PHL}					9	15	
t_{PHL}	$\overline{R}$	Q			15	27	ns

NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

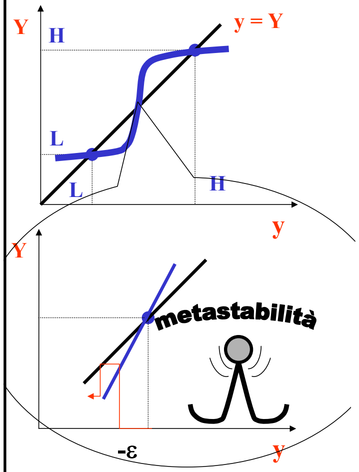
Cosa occorre per mantenere un 1 (o uno 0)?





Sull'anello di retroazione si deve poter mantenere la situazione di regime:
y = Y

Y=f(S,R,y): caratteristica in “catena chiusa”



Due tratti di “saturazione” (pendenza minore di 1) connessi da un tratto con “alto guadagno” (pendenza maggiore di 1): 3 intersezioni !

Per chiudere la retroazione occorre una amplificazione del segnale ed un comando “energico”.
Se l’impulso di set/reset ha durata inferiore al tempo di set-up il latch può andare in metastabilità.
Valore attuale?? E futuro ??