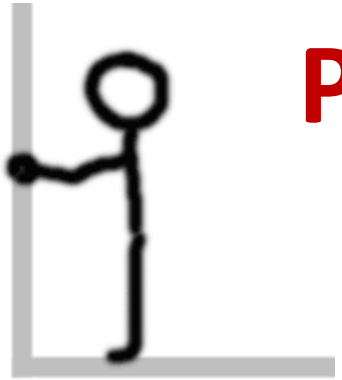
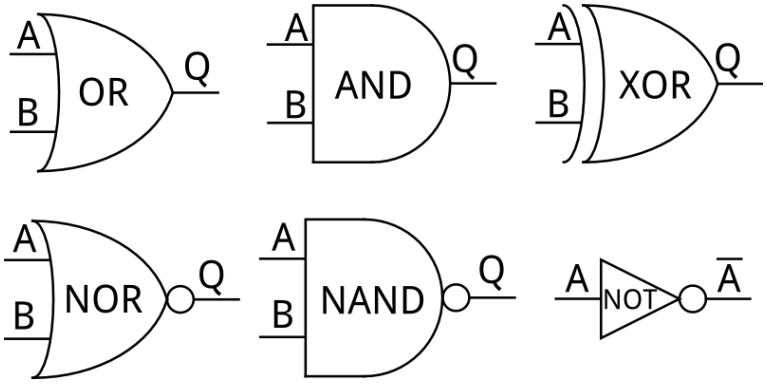
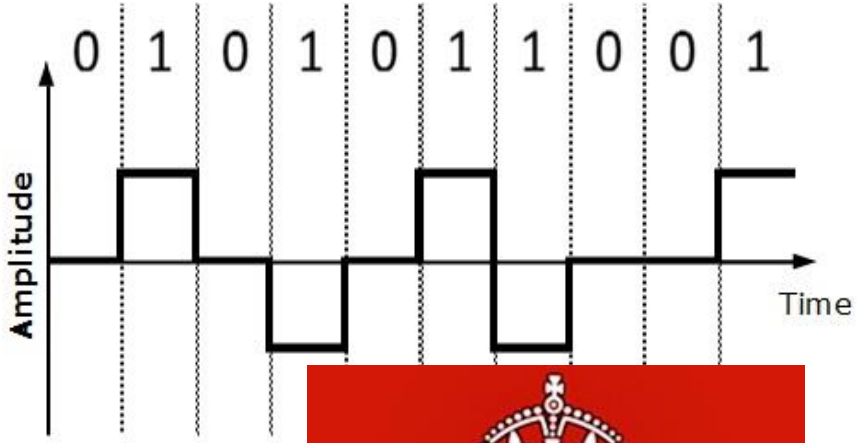


Proiectare Logica

Digital Logic Design

Curs 02





Date contact

- **Lector**: Cornel Mironel NICULAE
- **email**: **cornel.nicolae@unibuc.ro**
- **www**: cnic.ro/pl/
- Logic grid (simulator logic in HTML5): cnic.ro/pl/lg/
- **Carte de referinta:** B. Holdsworth and R.C. Woods, Digital Logic Design, 4th Edition, 1999

Recapitulare Curs 01

- Schimbarea bazei de numeratie
- Reprezentarea numerelor intregi si fractionale in baza 2 si alte baze uzuale de numeratie.
- Adunarea, scaderea si inmultirea in baza 2
- Conceptul de registru
- Complementul aritmetic;
- Reprezentarea in complement fata de 2 a numerelor intregi

Schimbarea bazei de numeratie 3

Numere intregi:

Numar (Cat anterior)	Impartitor (baza)	Rest
213	2	1
106	2	0
53	2	1
26	2	0
13	2	1
6	2	0
3	2	1
1	2	1

LSD

MSD

$$213_{10} = 11010101_2$$

Schimbarea bazei de numeratie 4

Numere fractionare: Baza 2

Numar (Cat anterior)	Partea intreaga
0.403	-
0.403*2=**0**.806	0
0.806*2=**1**.612	1
0.612*2=**1**.224	1
0.224*2=**0**.448	0
0.448*2=**0**.896	0
0.896*2=**1**.792	1
0.792*2=**1**.584	1
0.584*2=**1**.168	1

MSD

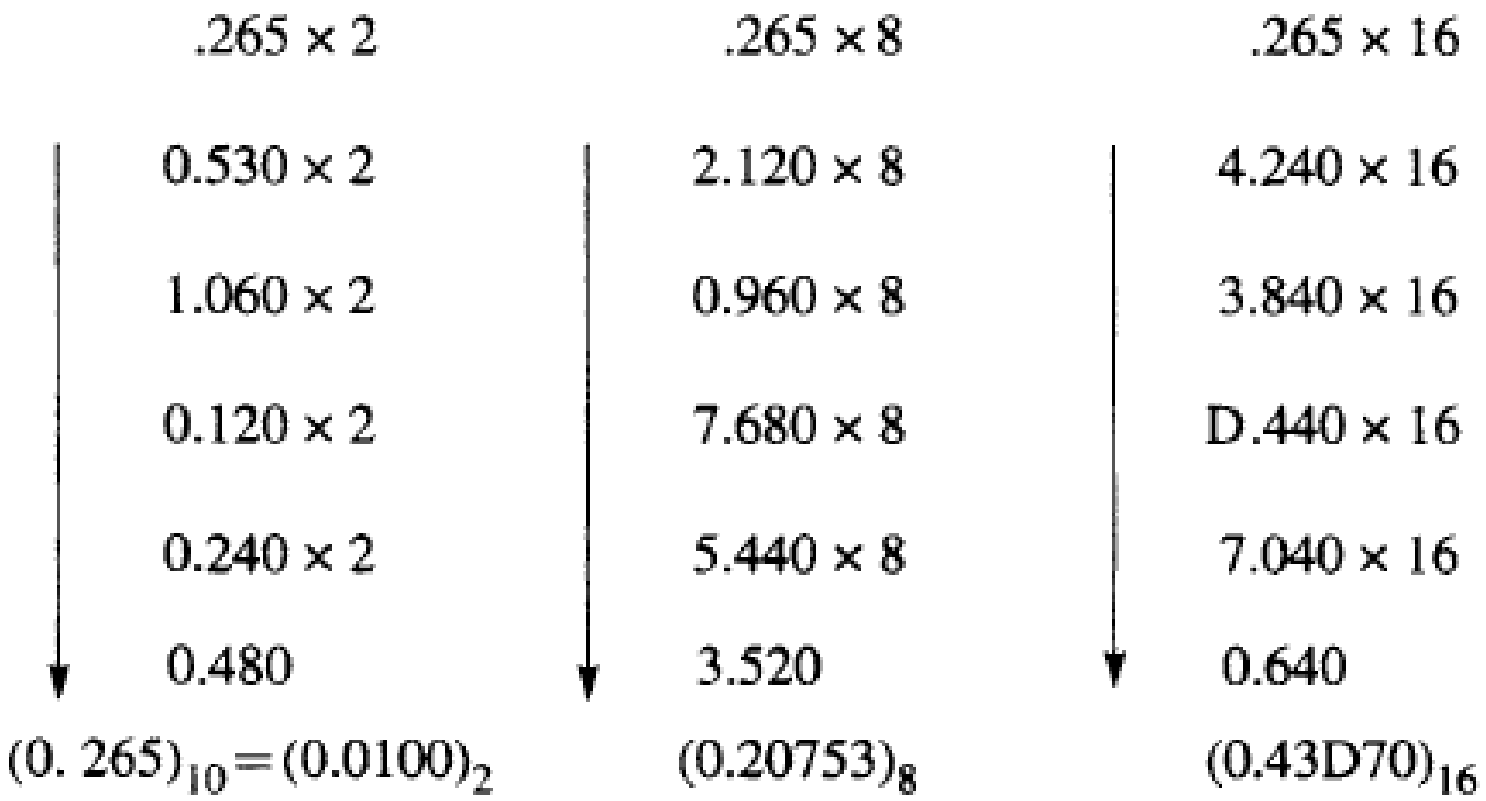
LSD

2^n	Aprox
-	
0*1/2	0
1*1/4	0.25
1*1/8	0.375
0*1/16	0.375
0*1/32	0.375
1*1/64	0.390625
1*1/128	0.3984375
1*1/256	0.40234375

$$0.403_{10} = 0.01000111\ldots_2$$

Schimbarea bazei de numeratie 5

Numere fractionare:



Bazele de numeratie 2, 8 si 16

In bazele de numeratie mai mari ca 10 se folosesc litere pentru desemnarea cifrelor mai mari de 10.

Octal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

HD	Binary	HD	Binary
0	000	8	1000
1	001	9	1001
2	010	A	1010
3	011	B	1011
4	100	C	1100
5	101	D	1101
6	110	E	1110
7	111	F	1111

Conversii binar <-->octal si invers

$$(110 \quad 001 \quad 011 \quad 100)_2$$
$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$
$$=(6 \quad 1 \quad 3 \quad 4)_8$$

$$(4 \quad 3 \quad 2 \quad .7)_8$$
$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$
$$=(100 \quad 011 \quad 010 \quad .111)_2$$

Octal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Conversii binar <--> hexazecimal

In bazele de numeratie mai mari ca 10 se folosesc litere pentru desemnarea cifrelor mai mari de 10.

$$(1011\ \ 1010\ \ 0011\ \ .0010)_2$$
$$=(B\ \ \ \ A\ \ \ \ 3\ \ \ \ .2)_{16}$$

$$(4\ \ \ \ F\ \ \ \ C\ \ \ \ 2)_{16}$$
$$=(0100\ \ 1111\ \ 1100\ \ 0010)_2$$

HD	Binary	HD	Binary
0	000	8	1000
1	001	9	1001
2	010	A	1010
3	011	B	1011
4	100	C	1100
5	101	D	1101
6	110	E	1110
7	111	F	1111

Scaderea Binara – rezultat pozitiv

	2^3	2^2	2^1	2^0	
	1	1	0	0	12
	0	0	1	1	−3
Diferenta	1	0	0	1	+9
Biti de imprumut		1	1		

Scaderea Binara – rezultat negativ

Daca apare bit de imprumut la cel mai semnificativ bit ➔ rezultatul este negativ.

	2^3	2^2	2^1	2^0	
	0	0	1	1	3
	1	1	0	0	–12
Diferenta	0	1	1	1	–9
Biti de imprumut	1	1			

Conceptul de registru

- In aplicatiile practice, de exemplu in calculatoare, avem limitari privind numarul de cifre alocate pentru un numar.
- De exemplu, putem alege sa reprezentam numere in baza 10 pe un spatiu ce poate memora MAXIM 4 cifre zecimale.
- Acest spatiu format din 4 locuri pentru cifrele din baza 10 reprezinta un **registru de 4 cifre zecimale**. Putem memora in acest registru numere intre 0000 si 9999. Numarul 3956 poate fi reprezentat in registru:

3	9	5	6

Complementul aritmetic 3

In baza 10:

Complementul fata de 9(10-1) pe un registru de 4 cifre**.**

8694 →numar initial

1305 →complementul fata de 9 (suma tuturor cifrelor face 9: cel mai mare numar din 4 cifre din baza 10)

Complement fata de 10

Se obtine din complementul fata de 9 la care se adauga o unitate.

Dorim sa calculam pe reg. de 3 cifre: 127-95=32; →

C(95)=904+1=905;905+127=1032→Trunchiat la 3 cifre

Complementul aritmetic 3

- In fiecare baza de numeratie exista 2 complementi:
  - Complementul fata de cea mai mare cifra (De ex. fata de 9 in baza 10)
  - Complementul fata de baza de numeratie (De ex. fata de 10 in baza 10
- In baza 2 vor exista 2 complementi:
  - Complementul fata de 1 si
  - Complementul fata de 2.
- **Ce complementi vor exista in baza 16?**

Reprezentarea binara a numerelor negative folosind C2

- Reprezentarea numerelor negative o constituie complementul fata de 2 a numarului pozitiv corespunzator.
  - De ex. folosind registrii de 8 biti 19_2=00010011_2
  - -19_{10}=C2(19)=11101101_2
  - Verificare:

```
  00010011+      +19
  11101101       -19
 ---------------------
 100000000         0
```

  - Bitul de semn al numerelor negative este 1
- In acesta reprezentare, un registru cu 8 biti poate memora numere intregi intre −128 si +127. Intr-adevar -128_{10}=10000000_2; $+127_{10}$ =01111111_2 ; -1_{10}= 11111111_2 . Verificarea se face prin adunarea:
  - 10000000_2+ -128_{10}
    01111111_2 $+127_{10}$
    11111111_2 -1_{10}

Reprezentare in C2

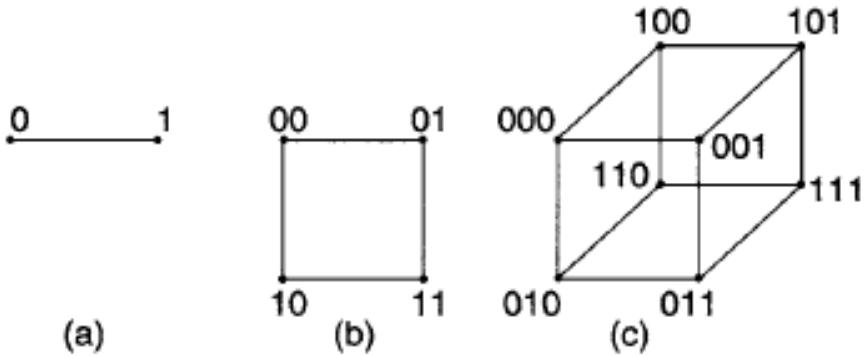
Pe **un registru de 4 biti** avem in aceasta reprezentare urmatoarele numere binare:

Nr zecimal	Nr. binar in C2	Nr zecimal	Nr. binar in C2
-1	1111	7	0111
-2	1110	6	0110
-3	1101	5	0101
-4	1100	4	0100
-5	1011	3	0011
-6	1010	2	0010
-7	1001	1	0001
-8	1000	0	0000

Curs 02

Distanta Hamming

- Distanta Hamming (distanta) dintre doua numere binare reprezentate pe acelasi numar de biti este egala cu **numarul de biti diferiti ai celor doua numere**. Ex d(101,011)=2.
- Exista reprezentari grafice numite **n-cuburi** in care graful asociat leaga noduri (in care sunt plasate numerele binare) situate **la distanta 1**: a) 1-cub, b) 2-cub, c) 3-cub



Distanta Hamming

- Distanta Hamming poate fi calculata daca se foloseste operatorul SUMA MODULO 2 (notata $\oplus$ = XOR=SAU EXCLUSIV), aplicata bitilor corespunzatori celor doua numere binare.

$$0 \oplus 0 = 0 \qquad 1 \oplus 0 = 1$$
$$0 \oplus 1 = 1 \qquad 1 \oplus 1 = 0$$

Ex. dist(101,110)=count(101$\oplus$110)=count(011)=2

Detectia si corectia erorilor

- in timpul unei transmisii digitale anumiti biti pot ajunge la destinatie schimbati.
- Ex. adaugarea unui bit suplimentar pentru verificarea paritatii:

Original Code	Modified Code	
	Even parity	Odd parity
000	0000	0001
001	0011	0010
010	0101	0100
011	0110	0111
100	1001	1000
101	1010	1011
110	1100	1101
111	1111	1110

Detectia si corectia erorilor 2

- Ex. verificarea paritatii (pare) a 4 numere binare de 4 biti atat pe orizontala cat si pe verticala

	X_1	X_2	X_3	X_4	p_h
	0	0	1	0	1
	1	0	1	0	0
	1	0	1	1	1
	1	1	0	0	0
p_v	1	1	1	1	0

a) Cod transmis corect

1	0	1	0	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
0	1	1	1	1

b) Cod transmis incorect: un bit a fost inversat

Codul Gray

- Exista mai multe moduri de a genera coduri Gray. Cea descrisa in continuare se numeste *binar reflectata*.

$g_2g_1g_0$ $b_2b_1b_0$

Reflected binary	Natural binary
000	000
001	001
011	010
010	011
110	100
111	101
101	110
100	111

De exemplu:

110=$b_2b_1b_0$ ➔ $g_2g_1g_0$=**101**

$$g_0 = b_0 \oplus b_1 = 0 \oplus 1 = 1$$
$$g_1 = b_1 \oplus b_2 = 1 \oplus 1 = 0$$
$$g_2 = b_2 \oplus b_3 = b_2 = 1$$

b_3 assumed to be 0

Codul ASCII

Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII
00	NUL	10	DLE	20	SP	30	0	40	@	50	P	60	`	70	p
01	SOH	11	DC_1	21	!	31	1	41	A	51	Q	61	a	71	q
02	STX	12	DC_2	22	"	32	2	42	B	52	R	62	b	72	r
03	ETX	13	DC_3	23	£(#)	33	3	43	C	53	S	63	c	73	s
04	EOT	14	DC_4	24	$	34	4	44	D	54	T	64	d	74	t
05	ENQ	15	NAK	25	%	35	5	45	E	55	U	65	e	75	u
06	ACK	16	SYN	26	&	36	6	46	F	56	V	66	f	76	v
07	BEL	17	ETB	27	'	37	7	47	G	57	W	67	g	77	w
08	BS	18	CAN	28	(	38	8	48	H	58	X	68	h	78	x
09	HT	19	EM	29	)	39	9	49	I	59	Y	69	i	79	y
0A	LF	1A	SUB	2A	*	3A	:	4A	J	5A	Z	6A	j	7A	z
0B	VT	1B	ESC	2B	+	3B	;	4B	K	5B	[	6B	k	7B	{
0C	FF	1C	FS	2C	,	3C	<	4C	L	5C	\	6C	l	7C	|
0D	CR	1D	GS	2D	-	3D	=	4D	M	5D	]	6D	m	7D	}
0E	SO	1E	RS	2E	.	3E	>	4E	N	5E	^	6E	n	7E	~
0F	SI	1F	US	2F	/	3F	?	4F	O	5F	_	6F	o	7F	DEL

Codul ASCII

Code HD	Symbol	Function
00	NUL	All the 0's
01	SOH	Indicates start of header field
02	STX	Indicates start of text
03	ETX	Indicates end of text
04	EOT	Termination of transmission
05	ENQ	Enquire if terminal is on
06	ACK	Informs Tx of receipt of error free data
10	DLE	Data link escape
15	NACK	Informs Tx of receipt of data containing errors
16	SYN	Establishes bit and character synchronism
17	ETB	Indicates the end of block of data

Tema de antrenament

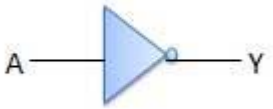
- Problemele 1.1-1.12 de la pagina 26 din cursul de referinta : B. Holdsworth and R.C. Woods, *Digital Logic Design*, 4th Edition, 1999

Algebra Booleana

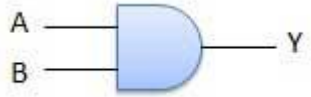
*I will **take** an **umbrella** with me if it is **raining** or the weather **forecast** is **bad**.*

Raining	Bad Forecast	Umbrella
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

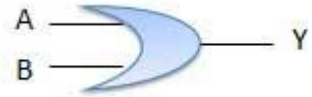
NOT → Y=A'

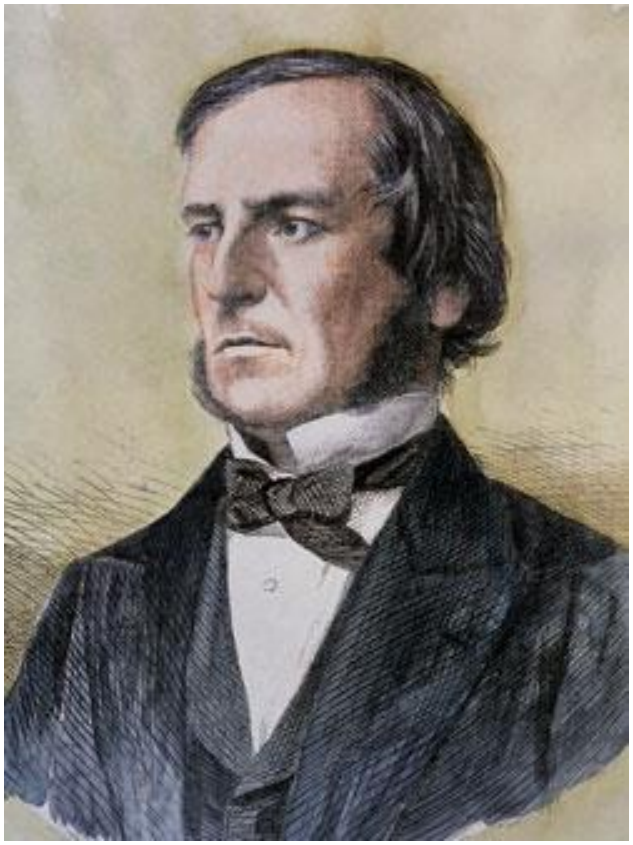


AND → Y=A·B



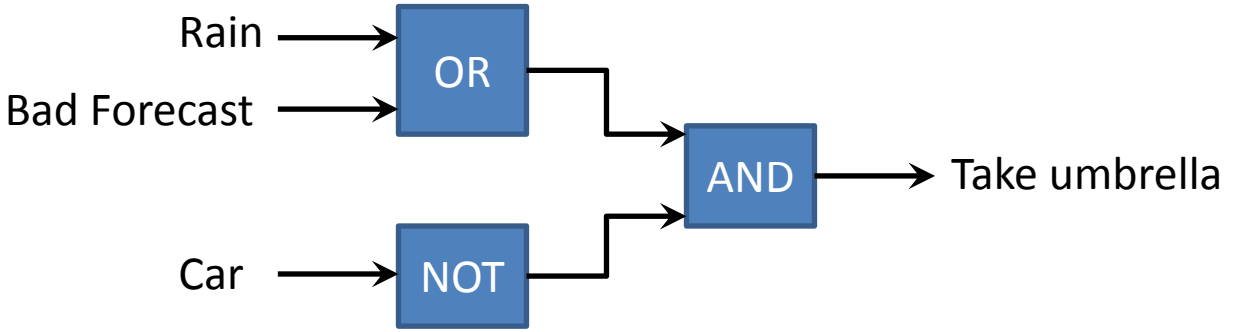
OR → Y=A+B





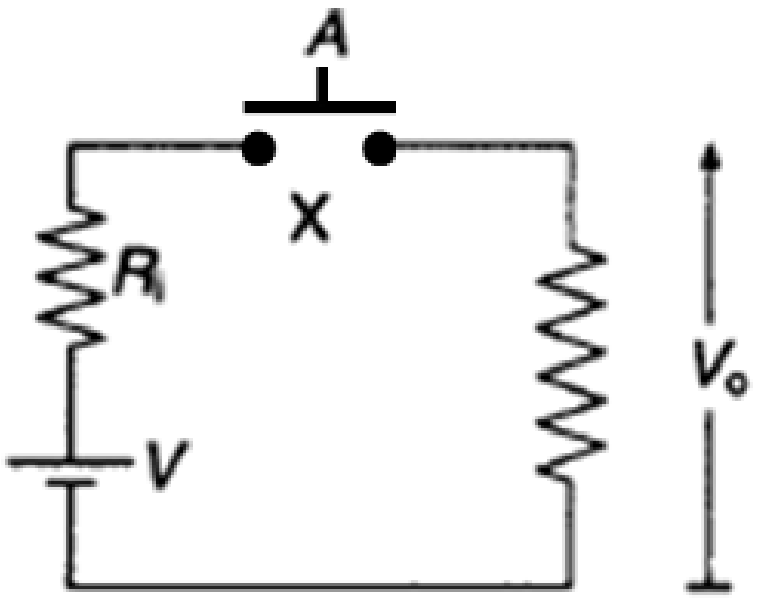
George Boole (1815-1864)

*If I do not take the **car** then I will take the **umbrella** if it is **raining** or the weather **forecast** is **bad.***

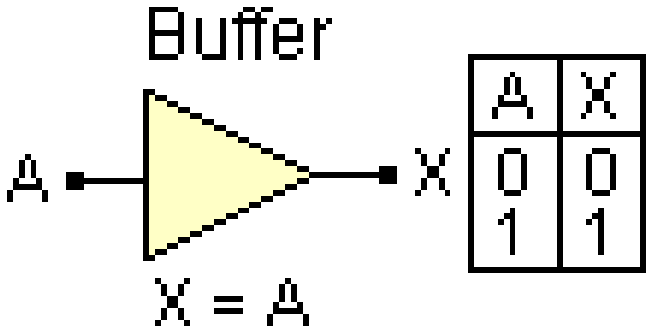


Circuite logice cu intrerupatoare

Repetorul logic: *Buffer, Driver*



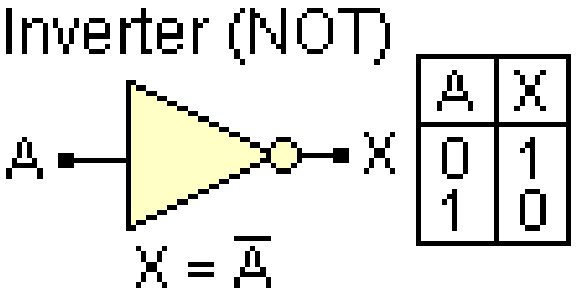
A	f
0	0
1	1



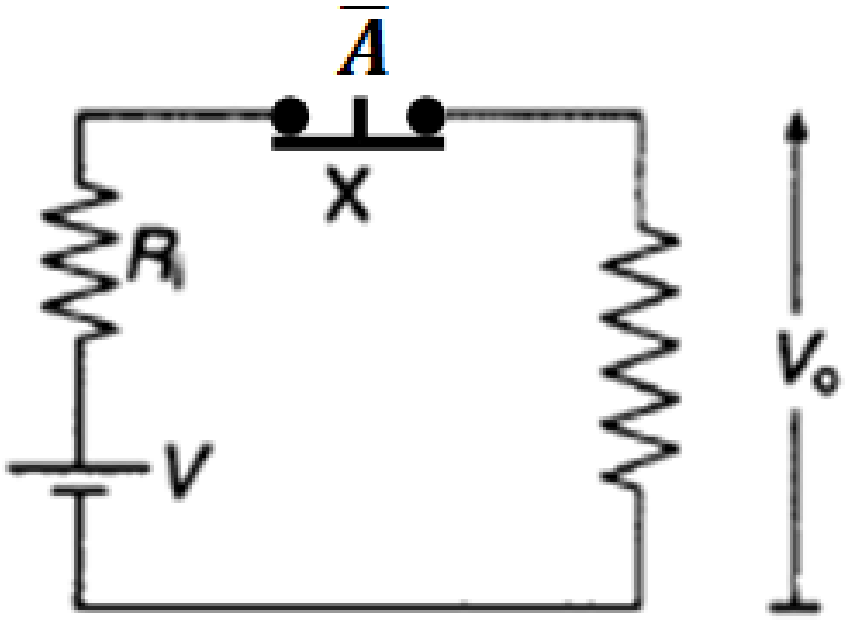
A	X
0	0
1	1

Circuite logice cu intrerupatoare

Inversorul logic: *NOT*

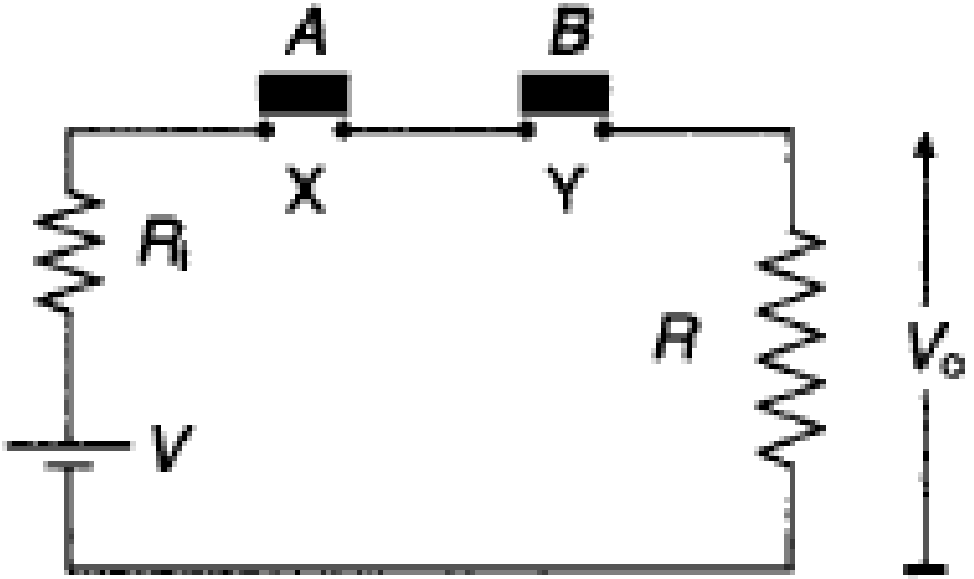


A	X
0	1
1	0



A	f
0	1
1	0

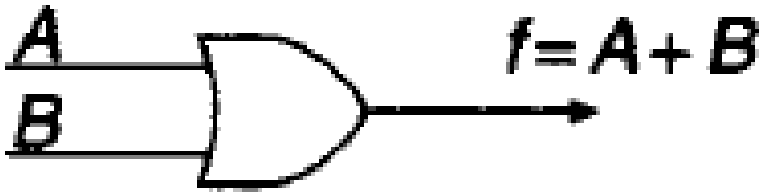
Functia SI (AND) cu intrerupatoare

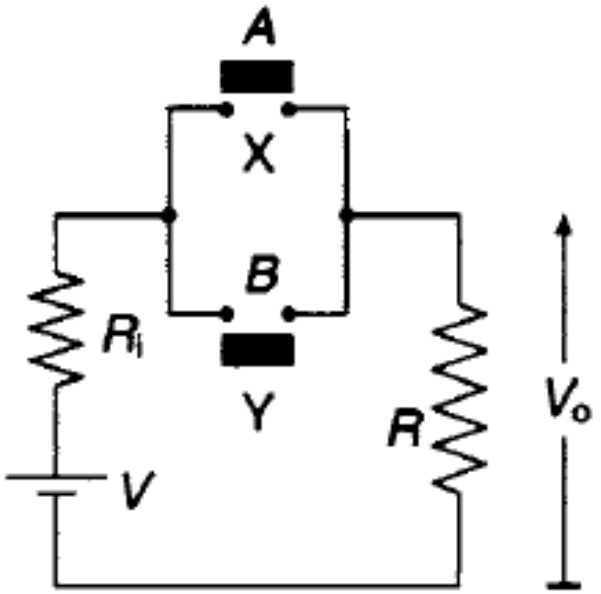


A	B	f
0	0	0
0	1	0
1	0	0
1	1	1

0.0 = 0
0.1 = 0
1.0 = 0
1.1 = 1

Functia SAU (OR) cu intrerupatoare

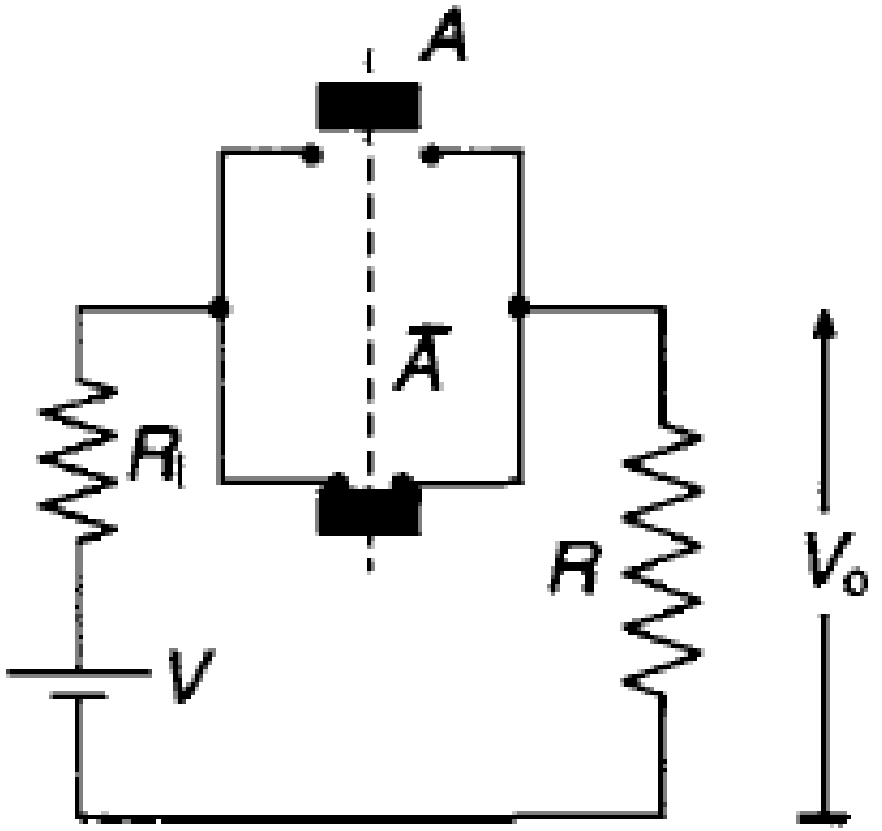




A	B	f
0	0	0
0	1	1
1	0	1
1	1	1

0 + 0 = 0	0 + 0 = 0
0 + 1 = 1	0 + 1 = 1
1 + 0 = 1	1 + 0 = 1
1 + 1 = 1	1 + 1 = 0 Carry 1
Boolean addition	Binary addition

Teorema complementarii



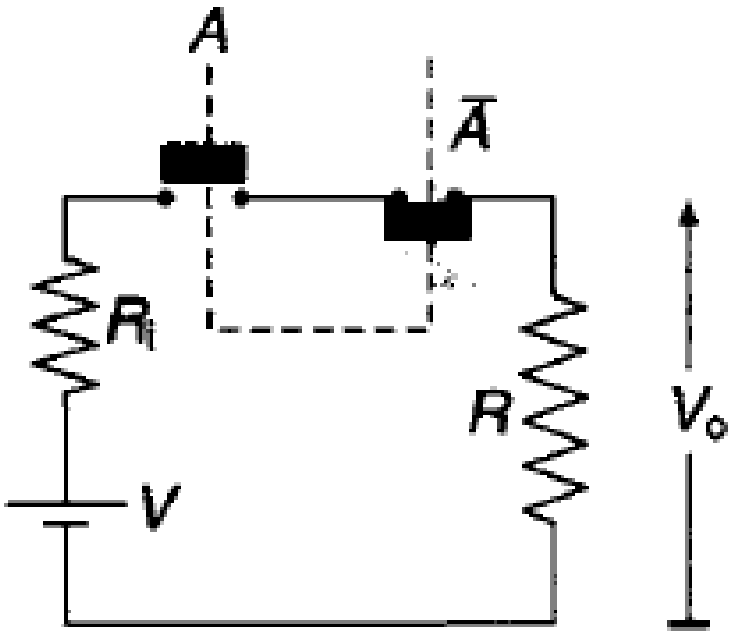
$$f = A + \bar{A}$$

A	$\bar{A}$	f
0	1	1
1	0	1

$$A + \bar{A} = 1$$

Duala T. complementarii

$$f = A \cdot \bar{A}$$



A	$\bar{A}$	f
0	1	0
1	0	0

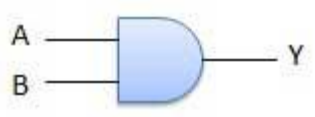
$$A \cdot \bar{A} = 0$$

AND – OR – NOT

Notatii, tabele de adevar si precedenta operatorilor.

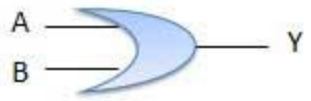
AND ·

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



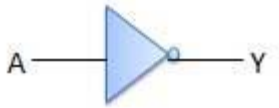
OR +

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



NOT ′

A	R
0	1
1	0



Operator	Symbol	Precedence
NOT	′	Highest
AND	·	Middle
OR	+	Lowest

Functii booleene

- Forma disjunctiva FD:
  - $f(A, B, C, D) = AB\bar{C} + CD + \bar{B}$

    Suma de produse

- Forma conjunctiva FC:
  - $g(A, B, C, D) = (A + B + C)(\bar{C}+\bar{D})(A + B)$

    Produs de sume

Functii booleene

- Forma canonica disjunctiva FCD:
  - $f(A,B,C) = a_0A'B'C' + a_1A'B'C + a_2A'BC' + \cdots + a_7ABC = \sum_{i=0}^{2^3-1} a_i m_i$, unde m_i se numeste mintermenul *i* iar $a_i \in \{0,1\}$

    Suma de produse

- Forma canonica conjunctiva FCC:
  - *g(A, B, C)* $=(b_0 + A + B + C) \cdot (b_1 + A + B + C') + \cdots + (b_7 + A' + B' + C') = \prod_{i=1}^{2^3-1}(b_i + M_i)$, unde M_i se numeste maxtermenul *i* iar $b_i \in \{0,1\}$

    Produs de sume

Aici sunt folosite identitatile:

$\boldsymbol{A \cdot 0 = 0}$

$\boldsymbol{A \cdot 1 = A}$: 1 = elem. neutru fata de " $\cdot$ "

$\boldsymbol{A + 0 = A}$: 0 = elem. neutru fata de " $+$ "

$\boldsymbol{A + 1 = 1}$

Tabele de adevar - mintermeni

- Forma analitica

Mintermeni

  - $f(A,B,C) = \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C =$
    $m_1 + m_2 + m_3 + m_5$

- Tabelul de adevar

Mintermen	Notatie	A	B	C	f
$\bar{A}\bar{B}\bar{C}$	m_0	0	0	0	0
$\bar{A}\bar{B}C$	m_1	0	0	1	1
$\bar{A}B\bar{C}$	m_2	0	1	0	1
$\bar{A}BC$	m_3	0	1	1	1
$A\bar{B}\bar{C}$	m_4	1	0	0	0
$A\bar{B}C$	m_5	1	0	1	1
$AB\bar{C}$	m_6	1	1	0	0
ABC	m_7	1	1	1	0

Unde trebuie sa ajungem?

Minimizarea expresiilor booleene

Forme canonice: exemplu

- Fie functia *f* definita de tabelul:

i	A	B	C	*f*	mintermeni (m_i)	Maxtermeni (M_i)
0	0	0	0	**1**	**A'B'C'**	A+B+C
1	0	0	1	**0**	A'B'C	**A+B+C'**
2	0	1	0	**0**	A'BC'	**A+B'+C**
3	0	1	1	**1**	**A'BC**	A+B'+C'
4	1	0	0	**1**	**AB'C'**	A'+B+C
5	1	0	1	**0**	AB'C	**A'+B+C'**
6	1	1	0	**1**	**ABC'**	A'+B'+C
7	1	1	1	**1**	**ABC**	A'+B'+C'

FCD: $f = A'B'C' + A'BC + AB'C' + ABC' + ABC = \sum m(0,3,4,6,7)$

FCC: $f = (A + B + C')(A + B' + C)(A' + B + C') = \prod M(1,2,5)$

Teoreme Simple:

$(A')' = A$: Dubla negatie == afirmatie

$A \cdot A' = 0$: O propozitie SI negatia sa ➔FALSA

$A + A' = 1$: O propozitie SAU negatia sa ➔ADEV.

Asociativitate, Comutativitate si Distributivitate

Asociativitate:

$$(A \cdot B) \cdot C = A \cdot (B \cdot C) = A \cdot B \cdot C$$

$$(A + B) + C = A + (B + C) = A + B + C$$

Comutativitate:

$$A \cdot B = B \cdot A$$

$$A + B = B + A$$

Distributivitate:

$$A \cdot (B + C) = A \cdot B + A \cdot C$$

$A + (B \cdot C) = (A + B) \cdot (A + C)$ → Ciudata

Reguli de simplificare

- **Idempotenta:**

$A \cdot A = A$

$A + A = A$

- **In care apar constantele booleene:**

$A \cdot 0 = 0$

$A \cdot 1 = A$: 1 = elem. neutru fata de " $\cdot$ "

$A + 0 = A$: 0 = elem. neutru fata de " $+$ "

$A + 1 = 1$

- **Grupul cel mai larg de reguli de simplificare:**

$A + A \cdot B = A$ si duala sa $A \cdot (A + B) = A$

$A + A \cdot B' = A$ si duala sa $A \cdot (A + B') = A$

Principiul dualitatii – Dualitatea De Morgan

Daca intr-o identitate booleana interschimbam constantele

$$0 \Leftrightarrow 1$$

si operatorii

" · " $\Leftrightarrow$ "+"

atunci obtinem o noua identitate, numita identitatea duala.

Exemple:

$A \cdot 0 = 0$ si duala sa $A + 1 = 1$

Teorememe De Morgan

Un exemplu remarcabil al principiului dualitatii il reprezinta teoremele De Morgan

$$(A \cdot B)' = A' + B'$$
$$(A + B)' = A'B'$$

Nota: In locul operatorului **NOT** notat cu (') – single quote (de ex. A') se mai foloseste simbolul $\neg$, adica $\neg A$. La fel de uzitata este si notatia cu bara deasupra: $\overline{A}$.

Termen de consens

- **Teorema de consens:**

Fie functia $f = A \cdot C + B \cdot C'$. Un termen de consens este format din cele doua variabile diferite de C, adica $A \cdot B$. Functia formata prin adaugarea acestui termen nu difera de cea initiala pentru orice valoare parametrilor: Adica $A \cdot C + B \cdot C' \equiv A \cdot C + B \cdot C' + A \cdot B$

- **Prin definitie un termen de consens este un termen a carei prezenta nu schimba valoarea functiei.**

$$C + ABC' + AB = C + ABC'$$

Scopul introducerii acestor termeni de consens este simplificarea. Expresia de mai sus se simplifica:

$$C + ABC' = C + AB$$

Demonstrarea identitatilor

$A+AB=A$

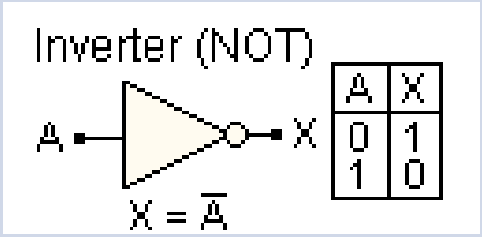
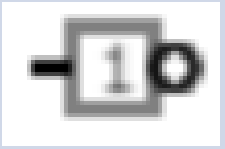
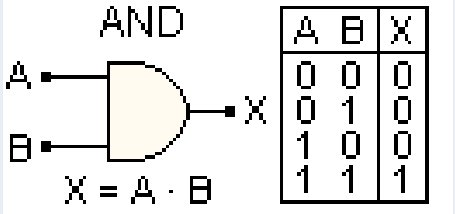
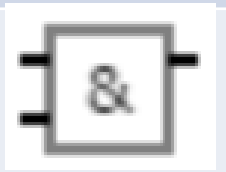
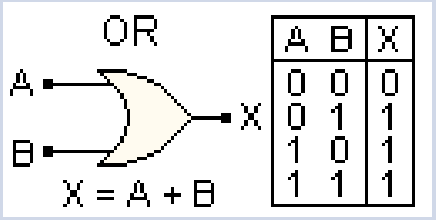
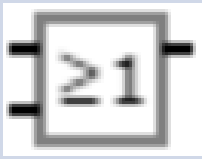
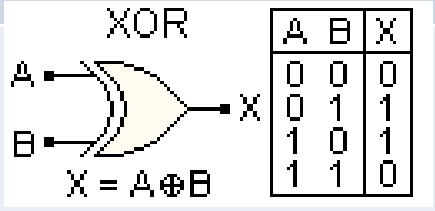
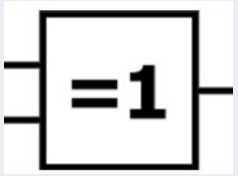
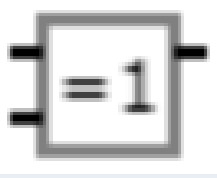
- **Se poate face folosind teoremele**

$$\begin{aligned} A + AB &= A \cdot 1 + AB \\ &= A(B + \bar{B}) + AB \\ &= AB + A\bar{B} + AB \\ &= AB + A\bar{B} \\ &= A(B + \bar{B}) \\ &= A \end{aligned}$$

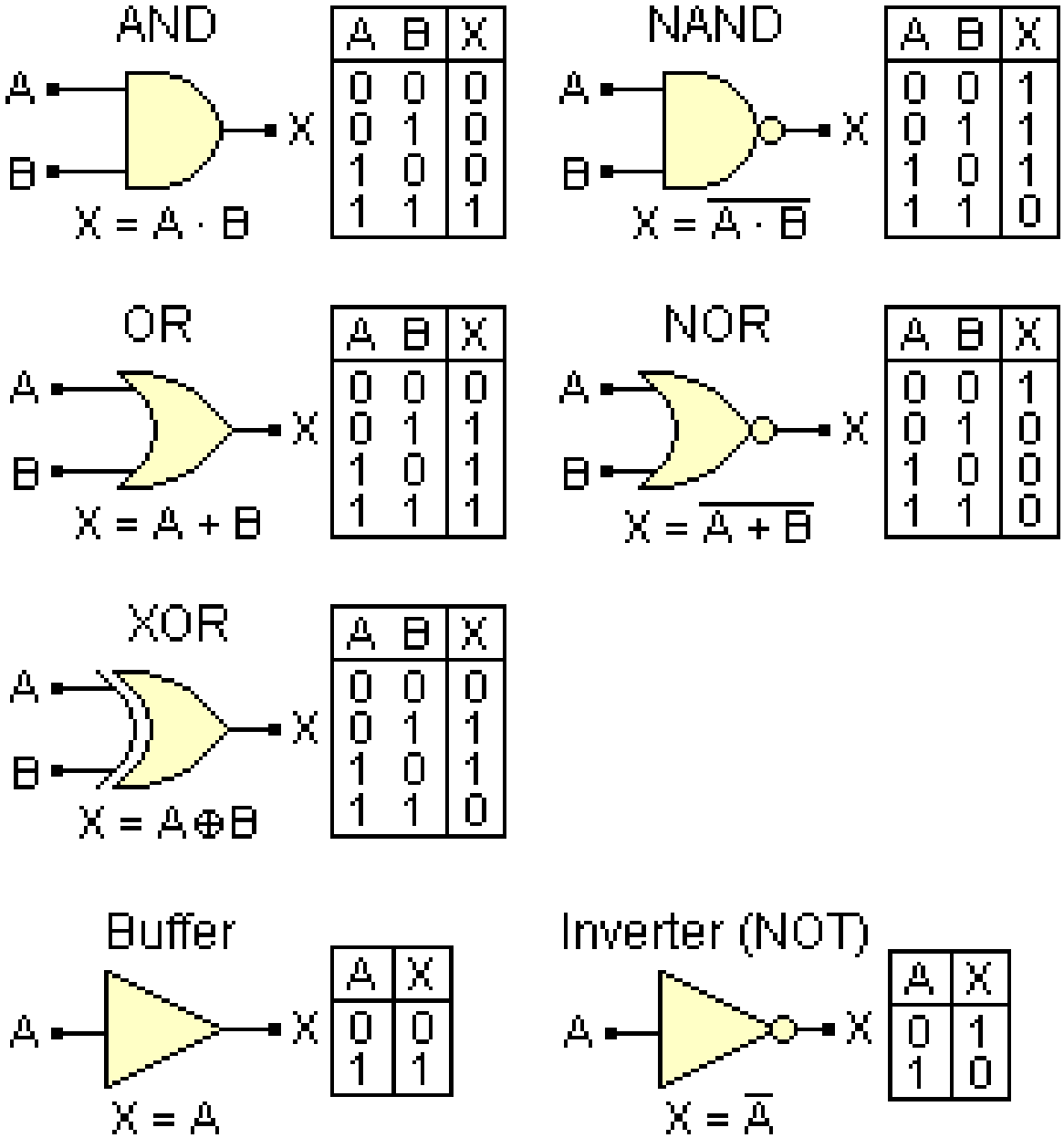
- **sau tabela de adevar**

A	B	AB	A+AB
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

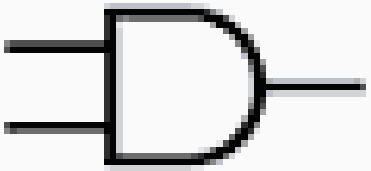
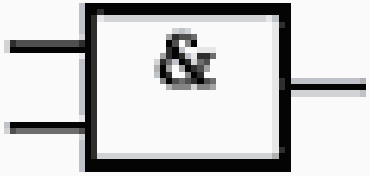
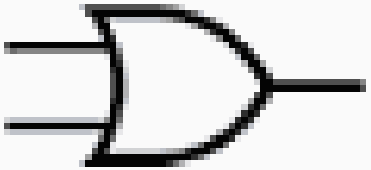
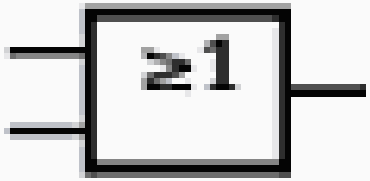
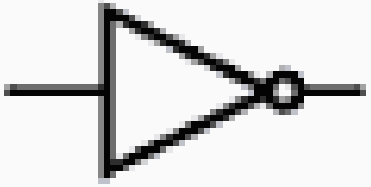
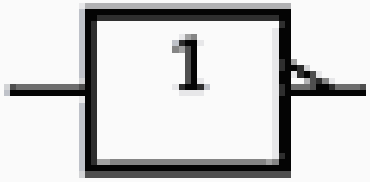
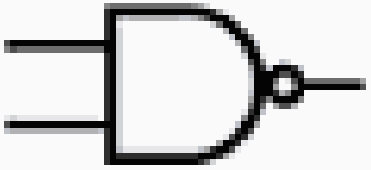
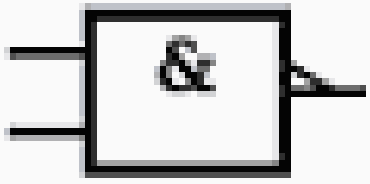
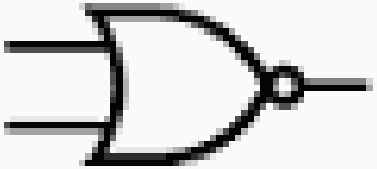
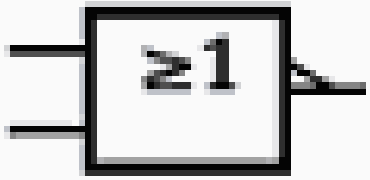
Operatii logice: simboluri si notatii

Operatie	Tip Op	Notatie	Simbol clasic	Simbol IEEE	wronex
NOT	unar	$\bar{A}$, A' sau $\neg A$			
AND	binar	$A \cdot B$			
OR	binar	$A + B$			
XOR	binar	$A \oplus B$			

Operatii logice: simboluri clasice



Operatii logice: simboluri clasice si IEEE

	Distinctive	Rectangular
AND		
OR		
NOT		
NAND		
NOR		

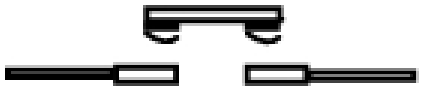
Gridul logic

- link: cnic.ro/pl/lg/

Tipuri ce intrerupatoare

SPST

Single-Pole
Single-Throw
Single-Break

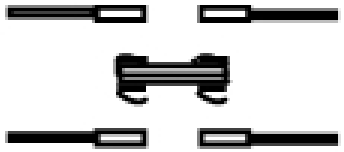


SPST

Single-Pole
Single-Throw
Double-Break

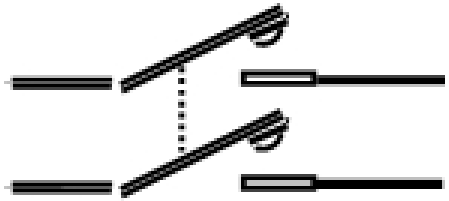
SPDT

Single-Pole
Double-Throw
Single-Break



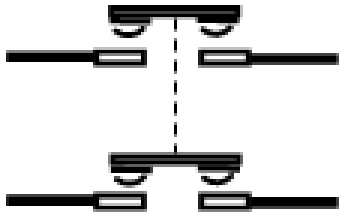
SPDT

Single-Pole
Double-Throw
Double-Break



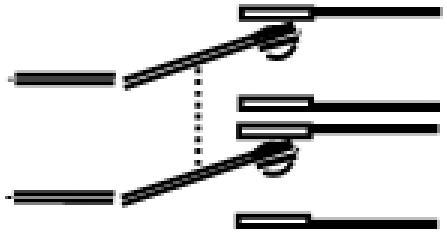
DPST

Double-Pole
Single-Throw
Single-Break



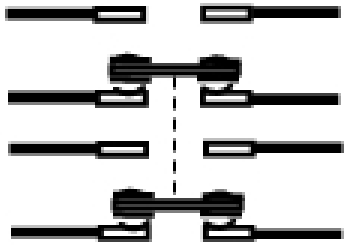
DPST

Double-Pole
Single-Throw
Double-Break



DPDT

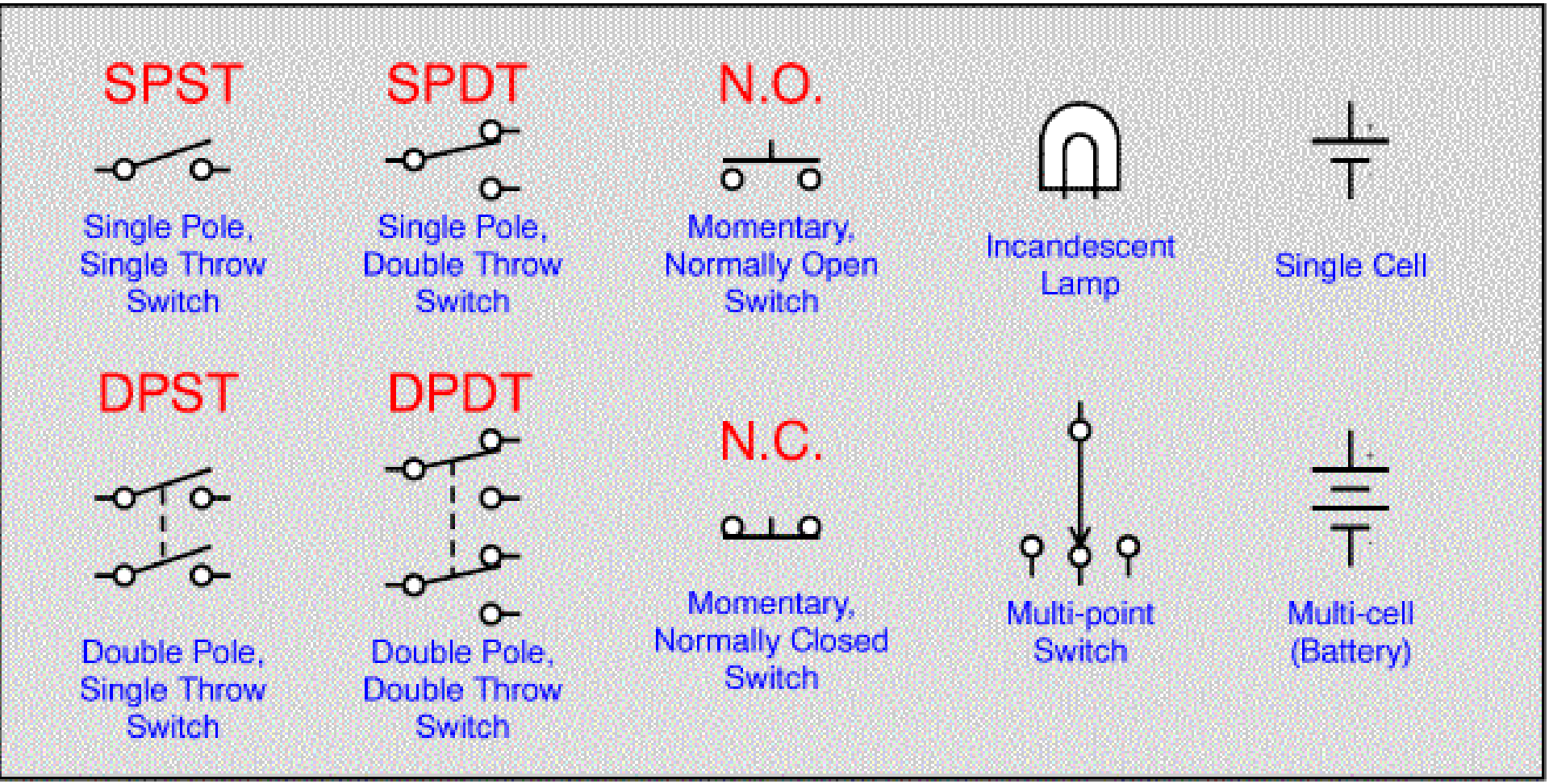
Double-Pole
Double-Throw
Single-Break



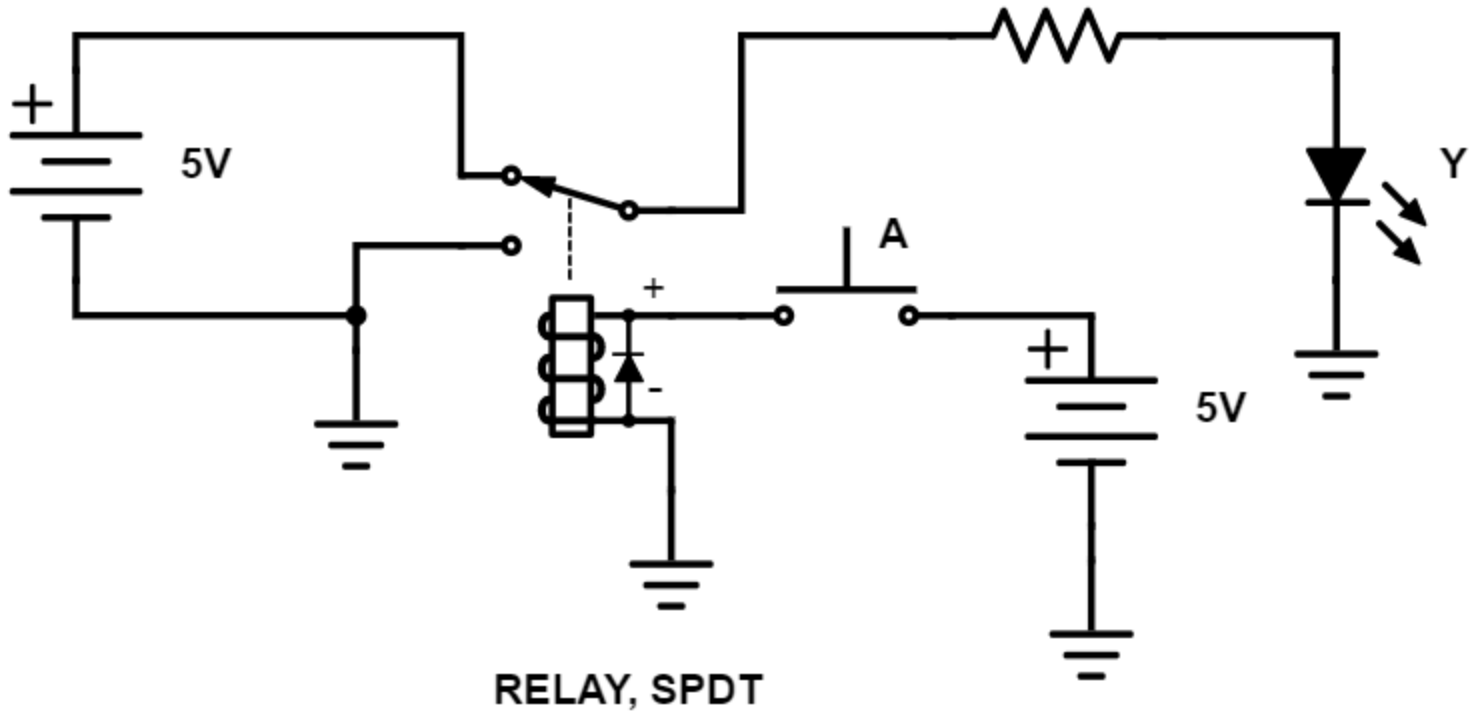
DPDT

Double-Pole
Double-Throw
Double-Break

Tipuri ce intrerupatoare

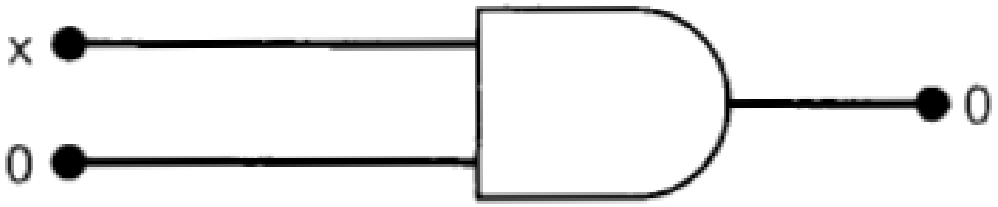


Inversorul logic cu releu SPDT

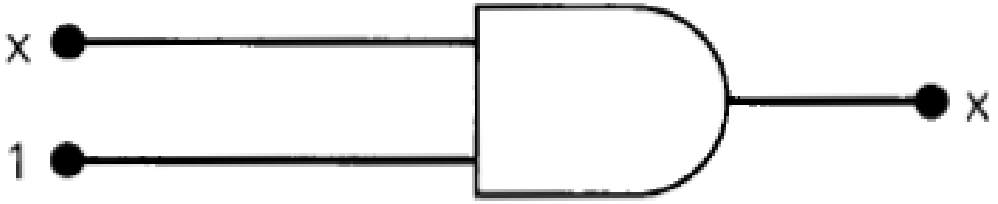


Teoreme cu o singura variabila (1)

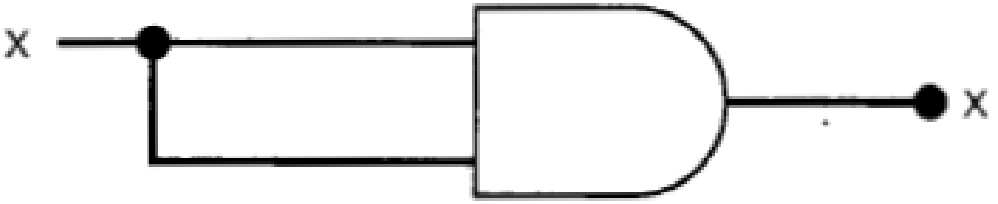
(1) $x \cdot 0 = 0$



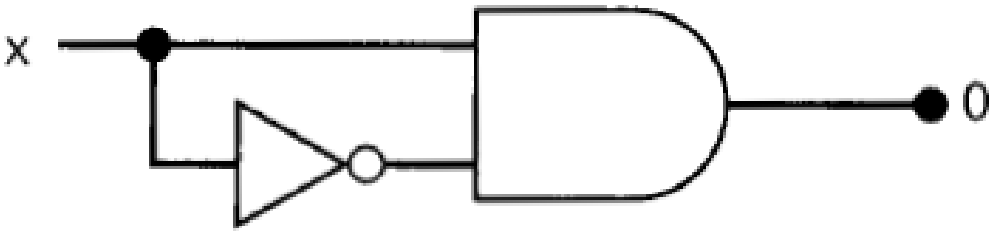
(2) $x \cdot 1 = x$



(3) $x \cdot x = x$

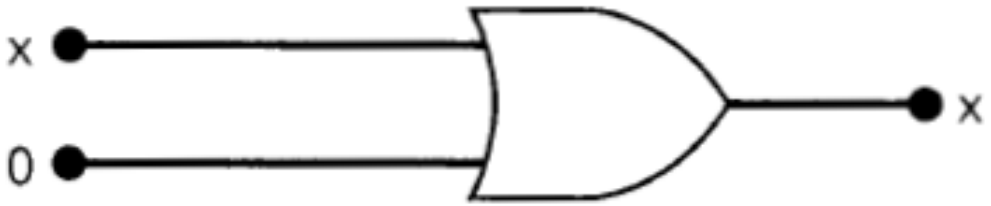


(4) $x \cdot \overline{x} = 0$

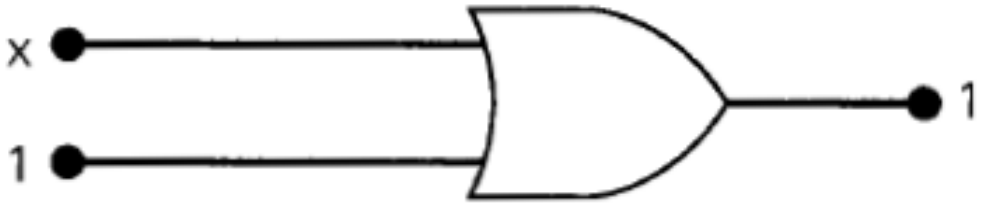


Teoreme cu o singura variabila (2)

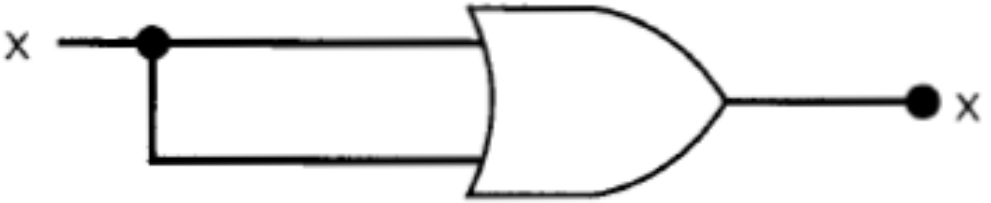
(5) $x + 0 = x$



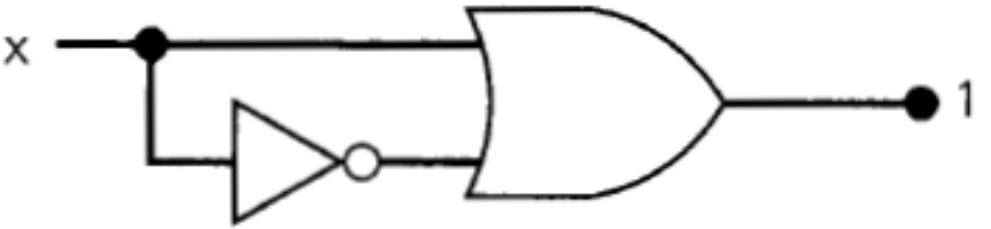
(6) $x + 1 = 1$



(7) $x + x = x$



(8) $x + \overline{x} = 1$



Teoreme cu mai multe variabile

(9) $x + y = y + x$

(10) $x \cdot y = y \cdot x$

(11) $x + (y + z) = (x + y) + z = x + y + z$

(12) $x(yz) = (xy)z = xyz$

(13a) $x(y + z) = xy + xz$

(13b) $(w + x)(y + z) = wy + xy + wz + xz$

(14) $x + xy = x$

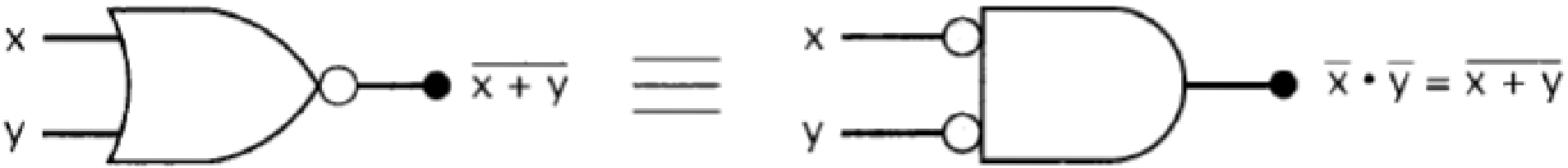
(15a) $x + \overline{x}y = x + y$

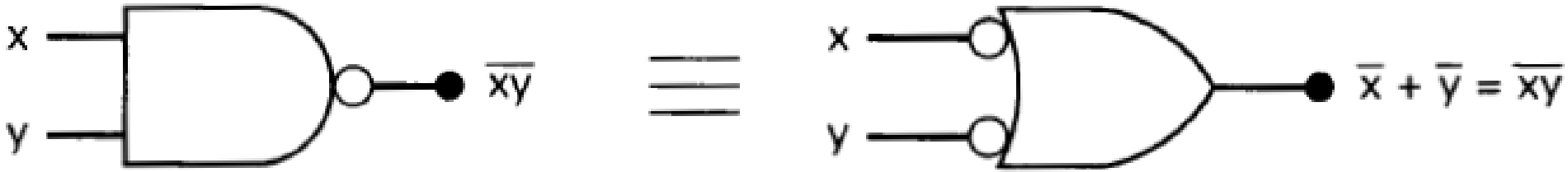
(15b) $\overline{x} + xy = \overline{x} + y$

Teoremele DeMorgan

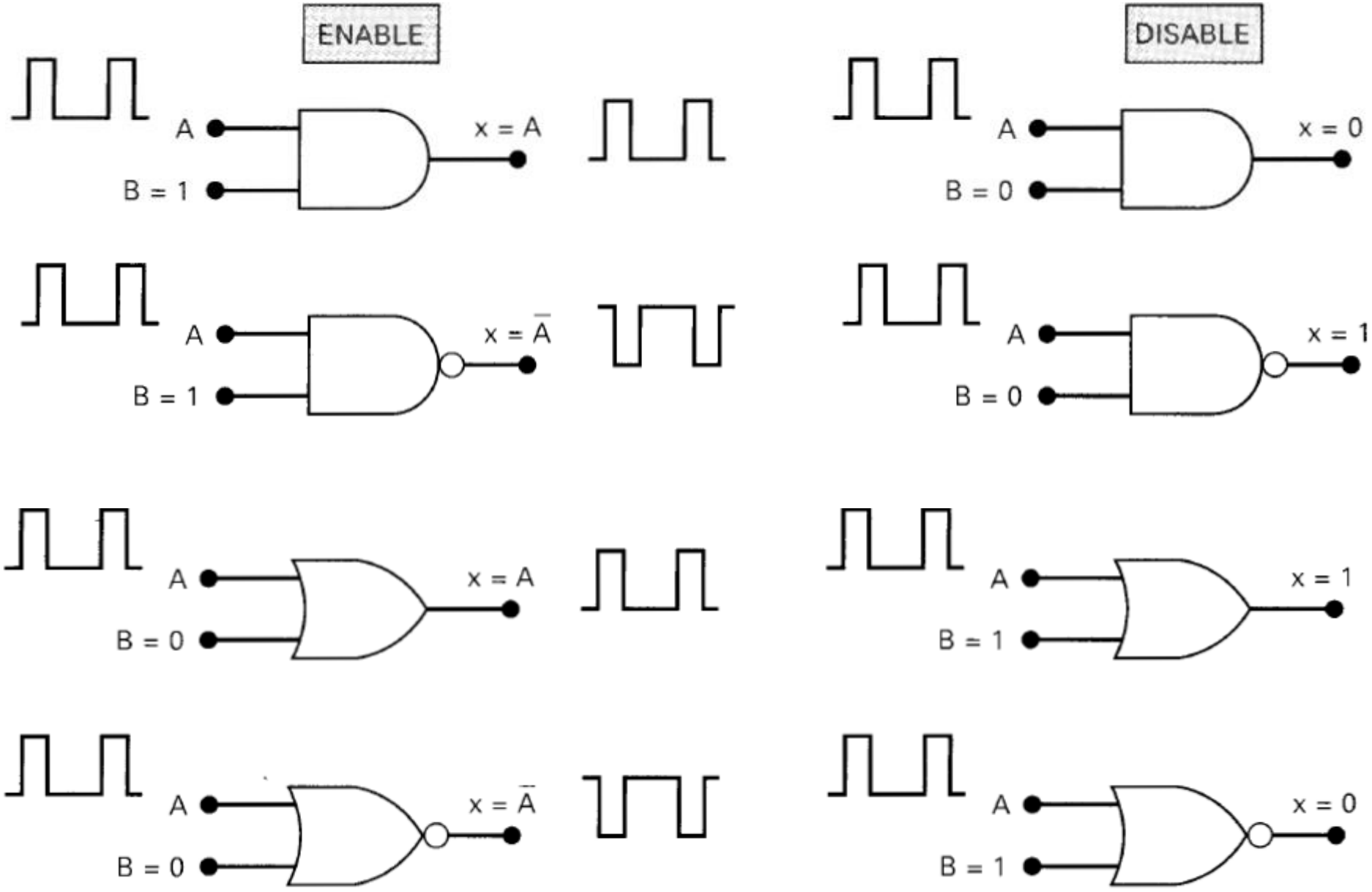
(16) **NOR** $\overline{(x + y)} = \overline{x} \cdot \overline{y}$

(17) **NAND** $\overline{(x \cdot y)} \equiv \overline{x} + \overline{y}$

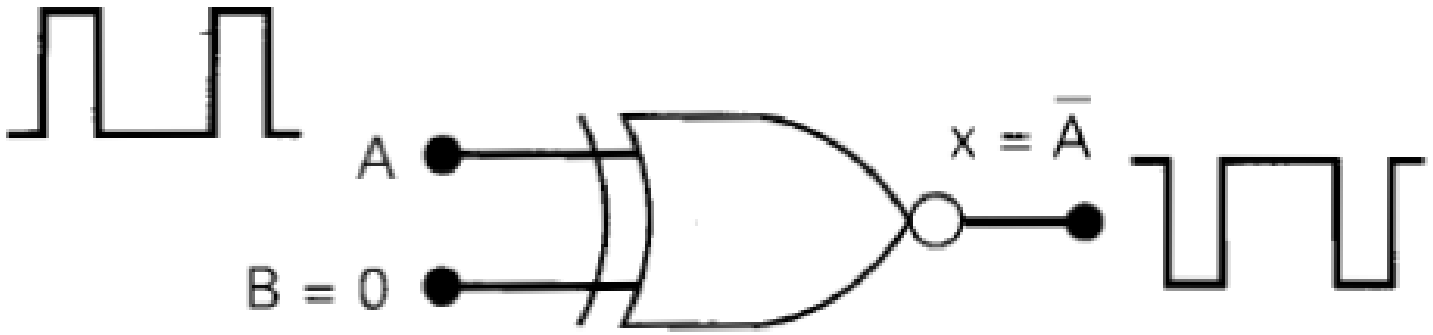


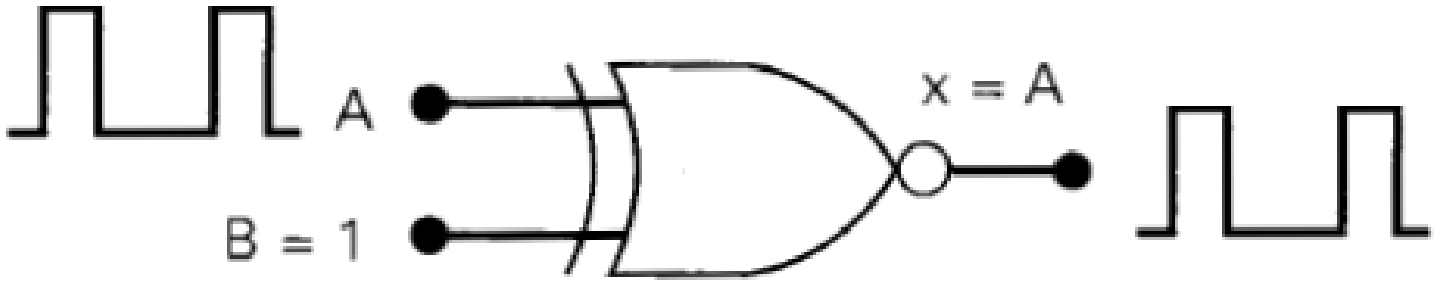


ENABLE/DISABLE Circuits



ENABLE/DISABLE Circuits – XNOR case





Circuitul de dirijare a pulsului– pulse-steering circuit